Statistical Learning Library in DROP

v3.93 2 December 2018

Probabilistic Bounds

Motivation

1. Design of Probabilistic Bounds: The primary objective of estimating bounds is to really bound the "bad metric", and it usually takes the form

$$a\mathbb{I}_{x\geq a} \leq x$$

or

$$a\mathbb{I}_{x>a} < x$$

Here x is the "bad metric", and a is the bounds (or cut-off) on the "bad metric".

2. Bounds Setting Techniques:
    a. Markov Inequality
    b. Jensen's Convexity Inequality
    c. Cauchy's Joint Bounding Inequality
    d. Tail Probability Bounds Approaches (in several chapters below)
    e. Sample Independence Inequality
    f. Local Taylor Expansion Inequality
    g. Local Extrema (mini-max) Inequality
    h. Cross Function (e.g., Hypothesis) Union Bounding Inequality
    i. Symmetrization Inequality

Tail Probability Bounds Estimation - Survey

1. Martingale Methods: Milman and Schechtman (1986) and McDiarmid (1989, 1998).
2. Information-Theoretic Methods: Ahlswede, Gacs, and Korner (1976), Marton (1986, 1996a, 1996b), Dembo (1997), Massart (1998), and Rio (2001).
3. Talagrand's Induction Methods: Talagrand (1995, 1996a, 1996b), Panchenko (2001), McDiarmid (2002), Panchenko (2002, 2003), and Luczak and McDiarmid (2003).
4. Entropy Methods: This is based on logarithmic Sobolev inequalities, see Ledoux (1996, 1997), Bobkov and Ledoux (1997), Boucheron, Lugosi, and Massart (2000), Massart (2000), Rio (2001), Bousquet (2002), Boucheron, Lugosi, and Massart (2003), and Lugosi (2009).
5. Random Graph Theory: Other problem specific methods in random graph theory are contained in Janson, Luczak, and Rucinski (2000).

Basic Probability Inequalities

1. Jensen's Inequality for Convex Functions:

$$f(\mathbb{E}[X]) \leq \mathbb{E}[f(X)]$$

2. Maximized Function Space Inequality:

$$E\left[\underset{f \in \Im}{sup}(\ldots)\right] \geq \underset{f \in \Im}{sup} E[\ldots]$$

since the former automatically filters out and retains only the measure space maxima.

3. Markov Inequality: Applies to the situation where x, a are strictly non-decreasing. Starting from the "bad metric" bound, and taking expectations on both sides, we get

$$P(x \geq a) \leq \frac{E[x]}{a}$$

or

$$P(x > a) < \frac{E[x]}{a}$$

This bounds the upper probability of a "bad outcome".

4. Chebyshev Inequality: Here we set

$$E[x] \to \{x - E[x]\}^2$$

Then

$$P(x \geq a) \leq \frac{E[x]}{a} \to P(\{x - E[x]\}^2 \geq a^2) \leq \frac{E[\{x - E[x]\}^2]}{a^2} \to P(\{x - E[x]\} \geq a) \leq \frac{Var[x]}{a^2}$$

This is the Chebyshev's inequality.

5. Generalized Moment Bounds: Set

$$E[x] \to \{x - E[x]\}^m$$

Using the treatment above, we get

$$P(\{x - E[x]\} \geq a) \leq \frac{Moment_{2k}(x)}{a^{2k}}; k = 1,2, \dots, \infty$$

Here

$$Moment_{2k}(x) = E[\{x - E[x]\}^{2k}]$$

6. Chernoff Bound: Here

$$E[x] \to e^x$$

so

$$P(x \geq a) = P(e^x \geq e^a) \leq e^{-a}E[e^x]$$

In practice, an additional coefficient t that allows for optimization is inserted, so

$$P(x \geq a) \leq e^{-at}E[e^{xt}]$$

for

$$a, t > 0$$

7. Chernoff vs. Moment Bounds: While both Chernoff bounds and generalized moment bounds allow for customization and/or optimization using the t parameter and the k parameter respectively, the range/width of moment bounding possible often produces more tight bounds than exponential/Chernoff bounds.

Cauchy-Schwartz Inequality

1. Statement: The Cauchy-Schwartz inequality states that if random variables X and Y have finite second moments –

$$\mathbb{E}[X^2] < \infty$$

and

$$\mathbb{E}[Y^2] < \infty$$

then

$$|\mathbb{E}[XY]| \leq \sqrt{\mathbb{E}[X^2]\mathbb{E}[Y^2]}$$

a. Consider 2 sequences of the random variables of X and Y - $(x_1, \dots, x_n)$ and $(y_1, \dots, y_n)$. Now,

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\left(x_i y_j - x_j y_i\right)^2 \geq 0$$

for obvious reasons. Expanding, we get

$$\sum_{i=1}^{n}\sum_{j=1}^{n} {x_i}^2 {y_j}^2 + \sum_{i=1}^{n}\sum_{j=1}^{n} {x_j}^2 {y_i}^2 - 2\sum_{i=1}^{n}\sum_{j=1}^{n} x_i x_j y_i y_j$$

$$\Rightarrow 2\left(\sum_{i=1}^{n} {x_i}^2\right)\left(\sum_{j=1}^{n} {y_j}^2\right) - 2\left(\sum_{i=1}^{n} x_i y_i\right)\left(\sum_{j=1}^{n} x_j y_j\right) \geq 0$$

Since

$$\sum_{i=1}^{n} {x_i}^2 = \mathbb{E}[X^2]$$

and

$$\sum_{i=1}^{n} {y_i}^2 = \mathbb{E}[Y^2]$$

and

$$\sum_{i=1}^{n} x_i y_i = \mathbb{E}[XY]$$

the result follows.

2. Chebyshev-Cantelli Inequality: Let

$$t \geq 0$$

Then

$$\mathbb{P}[X - \mathbb{E}[X] \geq t] \leq \frac{Var[X]}{Var[X] + t^2}$$

a. Proof => Re-casting

$$X - \mathbb{E}[X] \to X$$

$Var[X]$ does not alter. For all t

$$t = \mathbb{E}[t - X] \leq \mathbb{E}[(t - X)\mathbb{I}_{X<t}]$$

where $\mathbb{I}$ denotes the indicator function. Thus, for

$$t \geq 0$$

from the Cauchy-Schwartz inequality

$$t^2 \leq \mathbb{E}[(t - X)^2]\mathbb{E}\left[\mathbb{I}_{X<t}{}^2\right] = \mathbb{E}[(t - X)^2]\mathbb{P}(X < t) = [Var(X) + t^2]\mathbb{P}(X < t)$$

that is

$$\mathbb{P}(X < t) \geq \frac{t^2}{Var(X) + t^2} \Longrightarrow \mathbb{P}(X \geq t) \leq \frac{Var(X)}{Var(X) + t^2}$$

Association Inequalities

1. Motivation: These deal with inequalities dealing with non-increasing and non-decreasing sequences. Chebyshev inequality (Hall, Littlewood, and Polya (1952)) is a simple univariate inequality – Dubdashi and Ranjan (1998) show several instances where association properties may be used to derive concentration inequalities.
2. Chebyshev's Association Inequality: Let f and g both be non-decreasing (or non-increasing) real-valued functions defined on the real line. If X is a real-valued random variable, then

$$\mathbb{E}[f(X)g(X)] \geq \mathbb{E}[f(X)]\mathbb{E}[g(X)]$$

If f is non-increasing and g is non-decreasing (or vice-versa), then

$$\mathbb{E}[f(X)g(X)] \leq \mathbb{E}[f(X)]\mathbb{E}[g(X)]$$

   a. Proof => Let the random variable Y be distributed as X and be independent of it. If f and g are non-decreasing, then

$$[f(X) - f(Y)][g(X) - g(Y)] \geq 0$$

   so that

$$\mathbb{E}\{[f(X) - f(Y)][g(X) - g(Y)]\} \geq 0$$

   Expand this expectation to obtain the first inequality. The proof of the second inequality is similar.
3. Multi-variate Association Inequality: An important generalization of the Chebyshev's association inequality is the following. A real-valued function f defined on $\mathbb{R}^n$ is said to be

non-decreasing (non-increasing) in each variable while keeping all the other variables fixed at any value.

4. Harris' Inequality: Let

$$f: \mathbb{R}^n \to \mathbb{R}$$

be a non-decreasing function. Let $X_1, .., X_n$ be independent real-valued random variables and define the random variable

$$X = (X_1, .., X_n)$$

taking values in $\mathbb{R}^n$. Then

$$\mathbb{E}[f(X)g(X)] \geq \mathbb{E}[f(X)]\mathbb{E}[g(X)]$$

Similarly, if f is non-increasing and g is non-decreasing, then

$$\mathbb{E}[f(X)g(X)] \leq \mathbb{E}[f(X)]\mathbb{E}[g(X)]$$

a. Proof Step #1 => As before, it suffices to prove the first inequality. We proceed by induction. For

$$n = 1$$

the statement is just Chebyshev's association inequality.

b. Proof Step #2 => Now suppose that the statement is true for

$$m < n$$

Then

$$\mathbb{E}[f(X)g(X)] = \mathbb{E}\{\mathbb{E}[f(X)g(X)|X_1,..,X_{n-1}]\}$$
$$\geq \mathbb{E}\{\mathbb{E}[f(X)|X_1,..,X_{n-1}]\mathbb{E}[g(X)|X_1,..,X_{n-1}]\}$$

because, given $X_1,..,X_{n-1}$, both f and g are non-decreasing functions of the n^{th} variable. Now it follows by independence that the functions

$$f', g' : \mathbb{R}^{n-1} \to \mathbb{R}^n$$

defined by

$$f'(x_1, \dots, x_{n-1}) = \mathbb{E}[f(X)|X_1 = x_1,..,X_{n-1} = x_{n-1}]$$

and

$$g'(x_1, \dots, x_{n-1}) = \mathbb{E}[g(X)|X_1 = x_1,..,X_{n-1} = x_{n-1}]$$

are non-decreasing functions, so by the induction hypothesis

$$\mathbb{E}\left[f'^{(X_1,..,X_{n-1})}g'^{(X_1,..,X_{n-1})}\right] \geq \mathbb{E}[f'(X_1,..,X_{n-1})]\mathbb{E}[g'(X_1,..,X_{n-1})]$$
$$= \mathbb{E}[f(X)]\mathbb{E}[g(X)]$$

Moment, Gaussian, and Exponential Bounds

1. Moment Bounds vs. Chernoff Bounds: The moment bounds for tail probabilities are always better Chernoff bounds for tail probabilities. More precisely, let X be a non-negative random variable and set

$$t > 0$$

The best moment bound for tail probability is

$$\min_{q} \mathbb{E}[X^q]t^{-q}$$

where the minimum is taken over all positive integers q. The best Chernoff bound is

$$\inf_{s>0} \mathbb{E}\left[e^{s(X-t)}\right]$$

It can be shown that

$$\min_{q} \mathbb{E}[X^q]t^{-q} \leq \inf_{s>0} \mathbb{E}\left[e^{s(X-t)}\right]$$

2. First/Second Moments on Integer-valued Distributions: If X is a non-negative integer-valued random variable then it is obvious that

$$\mathbb{P}[X \neq 0] \leq \mathbb{E}[X]$$

Further, it can be shown that

$$\mathbb{P}[X \neq 0] \leq \frac{Var(X)}{Var(X) + \{\mathbb{E}[X]\}^2}$$

3. Sub-Gaussian Bounds: We say that a random variable X has a sub-Gaussian distribution if there exists a constant

$$c > 0$$

such that for all

$$s > 0$$

$$\mathbb{E}[e^{sX}] \leq e^{cs^2}$$

It can be shown that there exists a universal constant K such that if X is sub-Gaussian, then for every possible integer q,

$$\{\mathbb{E}[X_+{}^q]\}^{\frac{1}{q}} \leq K\sqrt{cq}$$

4. Sub-Gaussian Bounds - Converse: Let X be a random variable such that there exists a constant $c > 0$ such that

$$\{\mathbb{E}[X_+{}^q]\}^{\frac{1}{q}} \leq \sqrt{cq} \tag{1.19}$$

for every positive integer q. Then X is a sub-Gaussian – more precisely, for every $s > 0$,

$$\mathbb{E}[e^{sX}] \leq \sqrt{2}e^{\frac{1}{6}}e^{\frac{ces^2}{2}} \tag{1.20}$$

5. Sub-Exponential Bounds: We say that a random variable has a sub-exponential distribution if there exists a constant $c > 0$ such that for all $0 < s < \frac{1}{c}$,

$$\mathbb{E}[e^{sX}] \leq \frac{1}{1-cs} \tag{1.21}$$

If X is sub-exponential, then for every positive integer q,

$$\{\mathbb{E}[X_+{}^q]\}^{\frac{1}{q}} \leq \frac{4c}{e}q \tag{1.22}$$

6. Sub-Exponential Bounds - Converse: Let X be a random variable such that there exists a constant

$$c > 0$$

such that

$$\{\mathbb{E}[X_+{}^q]\}^{\frac{1}{q}} \leq cq$$

for every positive integer q. Then X is sub exponential – more precisely, for any

$$0 < s < \frac{1}{c}$$

$$\mathbb{E}[e^{sX}] \leq \frac{1}{1 - cs}$$

Bounding Sum of Independent Random Variables

1. Motivation – Sum of Independent Random Variables: Chebyshev's inequality and independence immediately imply

$$\mathbb{P}\{|S_n - \mathbb{E}[S_n]| \geq \varepsilon\} \leq \frac{\mathbb{E}[|S_n - \mathbb{E}[S_n]|^2]}{\varepsilon} = \frac{\sum_{i=1}^{n} \mathbb{E}[|X_i - \mathbb{E}[X_i]|^2]}{\varepsilon}$$

Writing

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n} \mathbb{E}[|X_i - \mathbb{E}[X_i]|^2]$$

we get

$$\mathbb{P}\left\{\left|\frac{1}{n}\sum_{i=1}^{n} X_i - \mathbb{E}[X_i]\right| \geq \varepsilon\right\} \leq \frac{\sigma^2}{n\varepsilon^2}$$

This is referred to as the ***Weak Law of Large Numbers***.

2. Inadequacy of the Weak Law: Recall that, under some additional regularity conditions, the central limit theorem states that

$$\mathbb{P}\left\{\sqrt{\frac{n}{\sigma^2}}\left|\frac{1}{n}\sum_{i=1}^{n} X_i - \mathbb{E}[X_i]\right| \geq \varepsilon\right\} \longrightarrow 1 - \Phi(\varepsilon) \leq \frac{1}{\sqrt{2\pi}}\frac{e^{-\frac{\varepsilon^2}{2}}}{\varepsilon}$$

from which, within a certain range of parameters, we expect something like

$$\mathbb{P}\left\{\frac{1}{n}\sum_{i=1}^{n} X_i - \mathbb{E}[X_i] \geq \varepsilon\right\} \approx e^{-\frac{n\varepsilon^2}{2\sigma^2}}$$

3. General Bounding Technique:
    a. Independence of individual variables => The first step is to exploit the independence of the random variables to extract a bound on the sum. Here the exponential individual bound is convenient, since that reduces to a joint product.
    b. Bounding individual Errors => Use Markov bounds in conjunction with moment/exponential bounds to bound the individual empirical errors.
    c. Upper Bounding Individual Variable => Find additional methods for upper-bounding each individual random variable. Transforming them using a monotone convex function make this convenient (Hoeffding does this).
4. Literature: Hoeffding's inequality appears in Hoeffding (1963). Ledoux and Talagrand (1992) contain the proof of the contraction principle. Proof of an equivalent inequality for binomial random variables was provided by Chernoff (1952) and Okamoto (1958).

Non-Moment Based Bounding - Hoeffding Bound

1. Principle of Hoeffding's Bounding:
    a. Hoeffding's inequality bounds the individual variable exponential probability in terms of another.
    b. This bound, along with "exponential Chernoff scaling", results in a convex optimization problem that may be optimized for an optimal "free parameter".
2. Hoeffding's Lemma: Suppose X is a real variable with zero mean such that

$$Prob(X \in [a, b]) = 1$$

Then

$$E[e^{sX}] \leq e^{\frac{1}{8}s^2(a-b)^2}$$

Note that if

$$a \neq 0$$

and

$$b \neq 0$$

then

$$a < 0$$

and

$$b > 0$$

However, if

$$a = 0$$

or

$$b = 0$$

it is easy to see that the inequality follows.

3. Proof of Hoeffding's Lemma:

- Exploit e^{sX} Convexity =>

$$e^{sx} \leq \frac{b-x}{b-a} e^{sa} + \frac{x-a}{b-a} e^{sb}$$

Applying expectation to both sides

$$E[e^{sX}] \leq \frac{b-E[X]}{b-a} e^{sa} + \frac{E[X]-a}{b-a} e^{sb}$$

Using

$$E[X] = 0$$

the LHS becomes

$$\left(-\frac{a}{b-a}\right) e^{sa} \left[-\frac{b-a}{a} - 1 + e^{s(b-a)}\right] = \left[1 - \theta + \theta e^{s(b-a)}\right] e^{-s\theta(b-a)}$$

where

$$\theta = -\frac{a}{b-a} > 0$$

- Substitution => Let

$$u = s(b - a)$$

and define

$$\varphi(u) = -\theta u + log(1 - \theta + \theta e^{u})$$

and

$$(1 - \theta + \theta e^{u}) = \theta\left[-\frac{b}{a} + e^{u}\right] \geq 0$$

for

$$\theta \geq 0, \frac{b}{a} \geq 0$$

Thus

$$E[e^{sX}] \leq E\left[e^{\varphi(u)}\right]$$

- Invoke Taylor's Expansion Inequality => By Taylor's theorem, for every real u there exists a v between 0 and u such that

$$\varphi(u) = \varphi(0) + u\varphi'^{(0)} + \frac{1}{2}u^{2}\varphi''^{(v)}$$

Note that

$$\varphi(0) = 0$$

$$\varphi'^{(0)} = 0$$

and

$$\varphi''(0) = \frac{\theta e^v}{1-\theta+\theta e^v}\left[1 - \frac{\theta e^v}{1-\theta+\theta e^v}\right] = t(1-t)$$

Recognize that

$$t > 0$$

$$t(1-t) \leq \frac{1}{4}$$

- Bring it together =>

$$\varphi(u) \leq \frac{1}{2}u^2\frac{1}{4} \leq \frac{1}{8}s^2(b-a)^2 => E[e^{sX}] \leq E\left[e^{\frac{1}{8}s^2(b-a)^2}\right]$$

5. Hoeffding's Inequality Statement: Let $X_1, \ldots, X_n$ be independent random variables such that

$$Prob(X_i \in [a_i, b_i]) = 1; 1 \leq i \leq n$$

Set

$$S_n = X_1 + \cdots + X_n$$

Then

$$Prob(S_n - E[S_n] \geq t) \leq e^{-\frac{2t^2}{\Sigma_{i=1}^n (b_i{}^2 - a_i{}^2)}}$$

a. Proof Step #1: For

$$s, t \geq 0$$

the Chernoff inequality and the independence of X_i implies

$$\begin{aligned} Prob\{S_n - E[S_n] \geq t\} &\leq Prob\left\{e^{s(S_n - E[S_n])} \geq e^{st}\right\} \leq e^{-st} E\left\{e^{s(S_n - E[S_n])}\right\} \\ &= e^{-st} \prod_{i=1}^{n} E\left\{e^{s(S_n - E[S_n])}\right\} \leq e^{-st} \prod_{i=1}^{n} e^{\frac{1}{8}s^2(b-a)^2} \\ &= e^{-st + \frac{1}{8}s^2 \sum_{i=1}^{n}(b_i - a_i)^2} \end{aligned}$$

b. Proof Step #2: To get the best possible upper bound, we find the minimum of the RHS as a function of s. Defining

$$g(t) = -st + \frac{1}{8}s^2 \sum_{i=1}^{n}(b_i - a_i)^2$$

$g(t)$ achieves its minimum at

$$\frac{4}{\sum_{i=1}^{n}(b_i - a_i)^2}$$

Substituting this value of s establishes the inequality.

6. Boundedness of Hoeffding:
   a. Recall that the Hoeffding inequality may only be applied for bounded (albeit negative), zero-mean random variables.
   b. The predictor ordinate boundedness of Hoeffding's inequality enables is applicable to piece-wise spline state estimators (esp. in the context of real-valued functions).

7. Hoeffding Inequality vs. Central Limit Theorem Inequality: As can be seen above, Hoeffding inequality has the same form as that of the Central Limit Theorem above, except for the fact that the average variance σ^2 is replaced by

$$\sigma^2 \rightarrow \frac{1}{4}\sum_{i=1}^{n}(b_i - a_i)^2$$

Thus, Hoeffding's inequality ignores the information on the variance of X_i, thus an alternate approach that accommodates this becomes necessary.

a. Inequality of Absolute Value Deviation => For ERM, we are also interested in absolute value deviations. The absolute value deviation around mean is looser (and symmetric)

$$Prob(S_n - E[S_n] \geq t) \leq e^{-\frac{2t^2}{\sum_{i=1}^{n}(b_i - a_i)^2}}$$

$$Prob(|S_n - E[S_n]| \geq t) \leq 2e^{-\frac{2t^2}{\sum_{i=1}^{n}(b_i - a_i)^2}}$$

b. In general, optimized moment bounds and its variants (including optimized factorial moment bounds for integer valued random variables and other types we see later) offer tighter, asymmetric bounds in a number of cases.

Moment Based Bounds

1. Moment-Bound Series: Without loss of generality, assume that $\mathbb{E}[X_i]$ for all

$$i = 1, \ldots, n$$

We look for the bounds of $\mathbb{E}[e^{sX_i}]$. Setting

$$\sigma_i{}^2 = \mathbb{E}\left[X_i{}^2\right]$$

and

$$F_i = \sum_{r=2}^{\infty} \frac{s^{r-2}\mathbb{E}[X_i{}^r]}{r!\,\sigma_i{}^2}$$

and using

$$e^{sx} = 1 + sx + \sum_{r=2}^{\infty} \frac{s^r x^r}{r!}$$

and

$$\mathbb{E}[X_i] = 0$$

we have

$$\mathbb{E}[e^{sX_i}] = 1 + s\mathbb{E}[X_i] + \sum_{r=2}^{\infty} \frac{s^r \mathbb{E}[X_i{}^r]}{r!} = 1 + s^2\sigma_i{}^2 F_i \leq e^{s^2\sigma_i{}^2 F_i}$$

2. Bounding Individual X_i: Assume that each X_i is bounded such that

$$|X_i| \leq c$$

Then for each

$$r \geq 2$$

$$\mathbb{E}[{X_i}^r] \leq c^{r-2}{\sigma_i}^2$$

Thus

$$F_i \leq \sum_{r=2}^{\infty} \frac{s^{r-2}c^{r-2}{\sigma_i}^2}{r!\,{\sigma_i}^2} = \frac{1}{(sc)^2}\sum_{r=2}^{\infty}\frac{(sc)^r}{r!} = \frac{e^{sc}-1-sc}{(sc)^2}$$

Thus, we have obtained

$$\mathbb{E}[e^{sX_i}] \leq e^{s^2{\sigma_i}^2\frac{e^{sc}-1-sc}{(sc)^2}}$$

3. Bounding X_i Moments: The above expression for F_i may be bounded differently, in other ways. Possible options are:
    a. Applying the moment bounds considered earlier along with potential sub-Gaussian-ness.
    b. Applying the moment bounds considered earlier along with potential sub-exponential-ness.
    c. Applying extraneous "noise variance" criterion such as the Massart's or the Tsybakov's noise conditions (to be considered later).
4. Sum of Independent Variables: Using the Chernoff bound for bounding the sums of independent random variables we get

$$\mathbb{P}\left\{\sum_{i=1}^{n} X_i \geq \varepsilon\right\} \leq e^{s^2{\sigma_i}^2\frac{e^{sc}-1-sc}{(sc)^2}-s\varepsilon}$$

Optimizing the choice of s, we find that the upper bound is minimized by

$$s = \frac{1}{c}\log\left[1+\frac{tc}{n\sigma^2}\right]$$

5. Bennett's Inequality (Bennett (1962)): Let $X_1, \ldots, X_n$ be independent real-valued random variables with zero mean, and assume

$$|X_i| \leq c$$

with probability one. Let

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n} Var[X_i]$$

Then, for any

$$\varepsilon > 0$$

$$\mathbb{P}\left\{\sum_{i=1}^{n} X_i \geq \varepsilon\right\} \leq e^{-\frac{n\sigma^2}{c^2}h\left(\frac{ct}{n\sigma^2}\right)}$$

where

$$h(u) = (1+u)\log(1+u) - u, u \geq 0$$

   a. Bound for Real-Valued Regression => The bound

$$|X_i| \leq c$$

   is quiet general – it may be applied for specific empirical situations in a customized manner (e.g., real-valued regression in a piece-wise constant sense where we expect the basis spline functions to be bounded).

6. Bernstein Inequality Motivation: The meaning behind Bennett's inequality is best seen if we do additional bounding. Applying the basis inequality

$$h(u) = \frac{u^2}{2 + \frac{2u}{3}}, \forall\, u \geq 0$$

(which may be seen by comparing the derivatives of both sides), we obtain the classical (but looser) inequality of Bernstein (Bernstein (1946)).

a. Statement => Under the same conditions as Bennett's inequality, for any

$$\varepsilon > 0$$

$$\mathbb{P}\left\{\sum_{i=1}^{n} X_i \geq \varepsilon\right\} \leq e^{-\frac{n\varepsilon^2}{2\sigma^2 + \frac{2c\varepsilon}{3}}}$$

7. Bernstein's Inequality vs. Central Limit Theorem Gaussian Bounds: Except for the term $\frac{2c\varepsilon}{3}$ in the denominator of the exponent, we can see that the Bernstein's inequality is qualitatively right/tight when we compare it with the Central Limit Theorem.
8. Bernstein's Inequality vs. Poisson Bounds: If

$$\sigma^2 > \varepsilon$$

then the upper bound given by Bernstein's inequality behaves as $e^{-n\varepsilon}$ instead of the $e^{-n\varepsilon^2}$ guaranteed by Central Limit Theorem/Hoeffding. This may be intuitively explained by recalling that a $Binomial\left(n, \frac{\lambda}{n}\right)$ distribution can be approximated, for large n, by a $Poisson(\lambda)$ distribution whose tail decreases as $e^{-\lambda}$.

Binomial Tails

1. Setup: We consider functions that are binary-valued, appropriate to the binary classifier setting. Thus, for a fixed function f, the distribution of $P_n f$ is just a binomial law of

parameters Pf and n, since we are assuming n i.i.d. random variables $f(\mathbb{Z}_i)$ which can be either 0 or 1, and are equal to 1 with probability

$$\mathbb{E}[f(\mathbb{Z}_i)] = Pf$$

2. Formulation: Denoting

$$p = Pf$$

we can have an exact expression for the deviations of $P_n f$ from Pf as

$$\mathbb{P}[Pf - P_n f \geq \varepsilon] = \sum_{k=0}^{\lfloor n(p-\varepsilon) \rfloor} \binom{n}{k} p^k (1-p)^{n-k}$$

Since this is not easy to manipulate, the treatment seen earlier has used an upper bound as provided by Hoeffding's inequality. However there exist other sharper bounds.

3. Alternate Upper Bounds: The following quantities are an upper bound on $\mathbb{P}[Pf - P_n f \geq \varepsilon]$:
    a. Hoeffding:

$$\mathbb{P}[Pf - P_n f \geq \varepsilon] \sim e^{-2n\varepsilon^2}$$

    b. Bernstein:

$$\mathbb{P}[Pf - P_n f \geq \varepsilon] \sim e^{-\frac{n\varepsilon^2}{2p(1-p)+\frac{2\varepsilon}{3}}}$$

    c. Bennett:

$$\mathbb{P}[Pf - P_n f \geq \varepsilon] \sim e^{-\frac{np}{1-p}\left\{\left(1-\frac{\varepsilon}{p}\right)\log\left(1-\frac{\varepsilon}{p}\right)+\frac{\varepsilon}{p}\right\}}$$

d. Exponential:

$$\mathbb{P}[Pf - P_n f \geq \varepsilon] \sim \left(\frac{1-p}{1-p-\varepsilon}\right)^{n(1-p-\varepsilon)} \left(\frac{p}{p+\varepsilon}\right)^{n(p+\varepsilon)}$$

4. Extremes of Binomial Distribution: From the above bounds, using inversion, we can say that, roughly speaking, small deviations of $Pf - P_n f$ have a Gaussian behavior of the form $e^{-\frac{n\varepsilon^2}{2p(1-p)}}$ (i.e., Gaussian with variance $p(1-p)$) while large deviations have a Poisson behavior of the form $e^{-\frac{3n\varepsilon}{2}}$.
5. Reduction to Hoeffding's Inequality: Thus, typical binomial tails are heavier then Gaussian. Hoeffding's inequality lies in upper bounding the tails with a Gaussian of maximum variance (i.e., $\frac{1}{4}$), hence the term $e^{-2n\varepsilon^2}$.
6. Empirical Bernstein Bound: Each function $f \in \Im$ has a different variance, i.e.,

$$Pf(1 - Pf) \leq Pf$$

For each $f \in \Im$, using Bernstein's inequality, with a probability of at least $1 - \delta$,

$$Pf - P_n f \leq \sqrt{\frac{2Pf \log\frac{1}{\delta}}{n}} + \frac{2\log\frac{1}{\delta}}{3n}$$

7. Intuition Behind Common Concentration of Measure Inequalities: The Gaussian part $\sqrt{\frac{2Pf \log\frac{1}{\delta}}{n}}$ dominates when Pf is not too small, or if n is large enough, and it depends on Pf. Thus, several concentration of measure inequality formulations strive to combine Bernstein's inequality with the union bound and symmetrization.

Custom Bounds for Special i.i.d. Sequences

1. [0, 1] Bounded Random Variables: Let $X_1, \ldots, X_n$ be independent random variables taking values in $[0, 1]$. Denoting

$$m = \mathbb{E}[S_n]$$

for any

$$t \geq m$$

Chernoff bounding can be used to show that

$$\mathbb{P}\{S_n \geq t\} \leq \left(\frac{m}{t}\right)^t \left(\frac{n-m}{n-t}\right)^{n-t}$$

   a. Corollary => From the above, it can be seen that

$$\mathbb{P}\{S_n \geq t\} \leq \left(\frac{m}{t}\right)^t e^{t-m}$$

   and for all

$$\varepsilon > 0$$

$$\mathbb{P}\{S_n \geq m(1+\varepsilon)\} \leq e^{-mh(\varepsilon)}$$

   where h is the function defined in Bennett's inequality. Finally,

$$\mathbb{P}\{S_n \leq m(1-\varepsilon)\} \leq e^{-\frac{m\varepsilon^2}{2}}$$

   (Karp (1988), Hagerup and Rub (1990)).

2. Poisson Random Variable: Comparing

$$\mathbb{P}\{S_n \geq t\} \leq \left(\frac{m}{t}\right)^t e^{t-m}$$

with the best Chernoff bound for the tail of a Poisson random variable

$$Y = Poisson(m)$$

reveals that

$$\mathbb{P}\{Y \geq t\} \leq \inf_{s>0} \frac{\mathbb{E}[e^{sY}]}{e^{st}} = \left(\frac{m}{t}\right)^t e^{t-m}$$

3. Sampling with Replacement: Let $\mathcal{X}$ be a finite set with $\mathcal{N}$ elements, and let $X_1, \ldots, X_n$ be a random sample without replacement from $\mathcal{X}$ and $Y_1, \ldots, Y_n$ a random sample with replacement from $\mathcal{X}$. For any convex real-valued function f, it may be shown that

$$\mathbb{E}\left[f\left(\sum_{i=1}^{n} X_i\right)\right] \leq \mathbb{E}\left[f\left(\sum_{i=1}^{n} Y_i\right)\right]$$

In particular, by taking

$$f(x) = e^x$$

we see that all inequalities derived from sums of independent random variables Y_i using Chernoff bounding remain true for the sum of X_i's (See Hoeffding (1963)).

References

- Ahlswede, R., P. Gacs, and J. Korner (1976): Bounds on Conditional Probabilities with Applications in multi-user Communication *Zeitschrift fur Wahrscheinlichkeitstheorie und verwandte Gebiete* **34** 157-177 (with corrections in **39** 353-354 (1977)).
- Bennett, G. (1962): Probability Inequalities for the Sum of Independent Random Variables *Journal of the American Statistical Association* **57** 33-45.
- Bernstein, S. N. (1946): *The Theory of Probabilities* **Gastehizdat Publishing House** Moscow.
- Bobkov, S., and M. Ledoux (1997): Poincare's Inequalities and Talagrand's Concentration Phenomenon for the Exponential Distribution *Probability Theory and Related Fields* **107** 383-400.
- Boucheron, S., G. Lugosi, and P. Massart (2000): A Sharp Concentration Inequality with Applications *Random Structures and Algorithms* **16** 277-292.
- Boucheron, S., G. Lugosi, and P. Massart (2003): Concentration Inequalities using the Entropy Method *Annals of Probability* **31** 1583-1614.
- Bousquet, O. (2002): A Bennett Concentration Inequality and its Application to Suprema of Empirical Processes *C. R. Acad. Sci. Paris* **334** 495-500.
- Chernoff, H. (1952): A Measure of Asymptotic Efficiency of Tests of a Hypothesis based on a Sum of Observations *Annals of Mathematical Sciences* **23** 493-507.
- Dembo, A. (1997): Information Inequalities and Concentration of Measure *Annals of Probability* **25** 927-939.
- Dubdashi, D., and D. Ranjan (1998): Balls and Bins: A Study in Negative Dependence *Random Structures and Algorithms* 99-124.
- Hagerup, T., and C. Rub (1990): A Guided Tour of Chernoff Bounds *Information Processing Letters* **33** 305-308.
- Hall, G. H., J. E. Littlewood, and G. Polya (1952): *Inequalities* **Cambridge University Press** London.
- Hoeffding, W. (1963): Probability Inequalities for Sums of Bounded Random Variables *Journal of the American Statistical Association* **58** 13-30.
- Janson, S., T. Luczak, and A. Rucinski (2000): *Random Graphs* **John Wiley** New York.
- Karp, R. M. (1988): *Probabilistic Analysis of Algorithms* **University of California, Berkeley**.

- Ledoux, M., and M. Talagrand (1991): *Probability in Banach Space* **Springer-Verlag** New York.
- Ledoux, M. (1996): On Talagrand's Deviation Inequalities for Product Measures *ESAIM: Probability and Statistics* **1** 63-87.
- Ledoux, M. (1997): Isoperimetry and Gaussian Analysis *Lectures on Probability Theory and Statistics (editor: P. Bernard)* **Ecole d'Ete de Probabilites de St-Flour XXIV-1994** 165-294.
- Luczak, M. J., and C. McDiarmid (2003): Concentration for Locally Acting Permutations *Discrete Mathematics* **265** 159-171.
- Lugosi, G. (2009): *Concentration of Measure Inequalities*.
- Marton, K. (1986): A Simple Proof of the Blowing-up Lemma *IEEE Transactions on Information Theory* **32** 445-446.
- Marton, K. (1996a): Bounding d-distance by Informational Divergence: A Way to prove Measure Concentration *Annals of Probability* **24** 857-866.
- Marton, K. (1996b): A Measure Concentration Inequality for contracting Markov Chains *Geometric and Functional Analysis* **6** 556-571 (Erratum: **7** 609-613 (1997)).
- Massart, P. (1998): Optimal Constants for Hoeffding Type Inequalities *Technical Report **98.86**, Mathematiques* **Universite de Paris-Sud**.
- Massart, P. (2000): About the Constants in the Talagrand's Concentration Inequalities for Empirical Processes *Annals of Probability* **28** 863-884.
- McDiarmid, C. (1989): On the Method of Bounded Differences, in: *Surveys in Combinatorics 1989* 148-188 **Cambridge University Press** Cambridge.
- McDiarmid, C. (1998): Concentration, in: *Probabilistic Methods for Algorithmic Discrete Mathematics* (M. Habib, C. McDiarmid, J. Ramirez-Alfonsin, and B. Reed) 195-248 **Springer** New York.
- McDiarmid, C. (2002): Concentration for Independent Permutations *Combinatorics, Probability, and Computing* **2** 163-178.
- Milman, V., and G. Schechtman (1986): *Asymptotic Theory of Finite-dimensional Normed Spaces* **Springer-Verlag** New York.
- Okamoto, M. (1958): Some Inequalities relating to the partial Sum of Binomial Probabilities *Annals of the Institute of Statistical Mathematics* **10** 29-35.

- Panchenko, D. (2001): A Note on Talagrand's Concentration Inequality *Electronic Communications in Probability* **6**.
- Panchenko, D. (2002): Some Extensions of an Inequality of Vapnik and Chervonenkis *Electronic Communications in Probability* **7**.
- Panchenko, D. (2003): Symmetrization Approach to Concentration Inequalities for Empirical Processes *Annals of Probability* **31 (4)** 2068-2081.
- Rio, E. (2001): Inegalities de concentration pour les processus empiriques de classes de parties *Probability Theory and Related Fields* **119** 163-175.
- Talagrand, M. (1995): Concentration of Measures and Isoperimetric Inequalities in Product Spaces *Publications Mathematiques de l'I.H.E.S.* **81** 73-205.
- Talagrand, M. (1996a): A New Look at Independence *Annals of Probability* **24** 1-34 (Special Invited Paper).
- Talagrand, M. (1996b): New Concentration Inequalities in Product Spaces *Inventiones Mathematicae* **126** 505-563.

Efron-Stein Bounds

Introduction

1. Motivation: The main purpose of this section is to demonstrate how the many tail inequalities for the sums of independent variables seen earlier can be extended to general functions of independent random variables. The simplest, yet the most powerful inequality of this kind is called the *Efron-Stein* inequality, which attempts to bound the variance of a general function. To obtain tail inequalities, one may simply use the variance to extend the bound using the Chebyshev's inequality.
2. Setup: Let $\mathcal{X}$ be some set, and let

$$g : \mathcal{X}^n \to \mathbb{R}$$

be a measurable function of n variables. We derive inequalities for the random variable

$$\mathcal{Z} = g(X_1, \dots, X_n)$$

and its expected value $\mathbb{E}[\mathcal{Z}]$ where $X_1, \dots, X_n$ are arbitrary independent (but not necessarily identically distributed) random variables in $\mathcal{X}$.

3. Notation: The main Efron-Stein inequality follows from the next result – the *Martingale Differences Sum Inequality*. To simplify the notation, we use $\mathbb{E}_i$ for the expected value with respect to the variable X_i, that is,

$$\mathbb{E}_i = \mathbb{E}[\mathcal{Z}|X_1, \dots, X_{i-1}, X_{i+1}, \dots, X_n]$$

Martingale Differences Sum Inequality

1. Statement:

$$Var[\mathcal{Z}] \leq \sum_{i=1}^{n} \mathbb{E}[(\mathcal{Z} - \mathbb{E}_i[\mathcal{Z}])^2]$$

   i.e., $Var[\mathcal{Z}]$ is less than or equal to the sum of the conditional variances of $\mathcal{Z}$ along each random variable dimension.

2. Proof:
   a. The proof is based on the elementary properties of conditional expectations. Recall that if X and Y are arbitrary bounded random variables, then

$$\mathbb{E}[XY] = \mathbb{E}\big[\mathbb{E}[XY|Y]]\big] = \mathbb{E}\big[Y\mathbb{E}[X|Y]\big]$$

   b. Introduce

$$\mathcal{V} = \mathcal{Z} - \mathbb{E}[\mathcal{Z}]$$

   where the expectation is across all $X_1, \ldots, X_n$, and define

$$\mathcal{V}_i = \mathbb{E}[\mathcal{Z}|X_1, \ldots, X_i] - \mathbb{E}[\mathcal{Z}|X_1, \ldots, X_{i-1}] \; \forall \; i = 1, \ldots, n$$

   Under this notation, note that

$$\mathcal{Z} = \mathbb{E}[\mathcal{Z}|X_1, \ldots, X_n]$$

   and

$$\mathcal{Z} = \mathbb{E}[\mathcal{Z}|\Theta]$$

where Θ refers to a NULL variable set.

c. Clearly

$$\mathcal{V} = \sum_{i=1}^{n} \mathcal{V}_i$$

since the individual $\mathbb{E}[\mathcal{Z}|X_1, \dots, X_i]$ telescope out. Further, note that $\mathcal{V}_i$ and $\mathcal{V}_j$ are martingales in X_i and X_j, respectively. Thus, $\mathcal{V}$ is written as a sum of martingale differences.

d. Now

$$Var[\mathcal{Z}] = \mathbb{E}\left[\left(\sum_{i=1}^{n} \mathcal{V}_i\right)^2\right] = \mathbb{E}\left[\sum_{i=1}^{n} {\mathcal{V}_i}^2\right] + 2\mathbb{E}\left[\sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} \mathcal{V}_i \mathcal{V}_j\right] = \mathbb{E}\left[\sum_{i=1}^{n} {\mathcal{V}_i}^2\right]$$

since, for any

$$i > j$$

$$\mathbb{E}[\mathcal{V}_i \mathcal{V}_j] = \mathbb{E}\left[\mathbb{E}[\mathcal{V}_i \mathcal{V}_j | X_i, \dots, X_j]\right] = \mathbb{E}\left[\mathcal{V}_j \mathbb{E}[\mathcal{V}_i | X_i, \dots, X_j]\right] = 0$$

as $\mathcal{V}_i$ is a martingale.

e. To bound

$$\mathbb{E}\left[\sum_{i=1}^{n} {\mathcal{V}_i}^2\right]$$

note that by independence of X_i

$$\mathbb{E}[\mathcal{Z}|X_1, \ldots, X_{i-1}] = \mathbb{E}[\mathbb{E}[\mathcal{Z}|X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_n]|X_1, \ldots, X_i]$$

and therefore

$$\begin{aligned}
\mathcal{V}_i{}^2 &= \{\mathbb{E}[\mathcal{Z}|X_1, \ldots, X_i] - \mathbb{E}[\mathcal{Z}|X_1, \ldots, X_{i-1}]\}^2 \\
&= \{\mathbb{E}[\mathbb{E}[\mathcal{Z}|X_1, \ldots, X_i] - \mathbb{E}[\mathcal{Z}|X_1, \ldots, X_{i-1}]|X_1, \ldots, X_i]\}^2 \\
&\leq \mathbb{E}[\{\mathbb{E}[\mathcal{Z}|X_1, \ldots, X_i] - \mathbb{E}[\mathcal{Z}|X_1, \ldots, X_{i-1}]\}^2|X_1, \ldots, X_i] \\
&= \mathbb{E}[(\mathcal{Z} - \mathbb{E}_i[\mathcal{Z}])^2|X_1, \ldots, X_i]
\end{aligned}$$

the penultimate step resulting from Jensen's inequality. Taking expected values on both sides, we obtain the statement inequality.

Efron-Stein Inequality

1. Statement: The Efron-Stein inequality (Efron and Stein (1981), Steele (1986)) follows from the above. To state the theorem, let $X'_1, \ldots, X'_n$ form an independent copy of $X_1, \ldots, X_n$, and set

$$\mathcal{Z}'_i = g(X_1, \ldots, X'_i, \ldots X_n)$$

   Then

$$Var[\mathcal{Z}] \leq \frac{1}{2}\sum_{i=1}^{n} \mathbb{E}[(\mathcal{Z} - \mathcal{Z}'_i)^2]$$

2. Proof of the Inequality: The statement follows from the Martingale Differences Sum Inequality by using the elementary fact that if X and Y are independent and identically distributed random variables, then

$$Var[X] \leq \frac{1}{2}\sum_{i=1}^{n} \mathbb{E}[(X - Y)^2]$$

and therefore

$$\mathbb{E}[(\mathcal{Z} - \mathbb{E}_i[\mathcal{Z}])^2] = \frac{1}{2}\sum_{i=1}^{n} \mathbb{E}[(\mathcal{Z} - \mathcal{Z}'_i)^2]$$

3. Relation to the Case of Sum of Independent Random Variables: Observe that in the case when

$$\mathcal{Z} = \sum_{i=1}^{n} X_i$$

is a sum of independent random variables of finite variance, the Efron-Stein inequality becomes an equality. Thus, in this sense, the bound produced by the Efron-Stein inequality cannot be improved. This also shows that, among all the functions of independent random variables, sums are, in some sense, the least concentrated.

4. Extended Efron-Stein Inequality: The Efron-Stein theorem may be extended to arbitrary measurable functions. A very useful corollary is obtained by recalling that, for any random variable X,

$$Var[X] \leq \mathbb{E}[(X - a)^2]$$

for any constant $a \in \mathbb{R}$. Using this fact, we have for every

$$i = 1, \dots, n$$

$$\mathbb{E}[(\mathcal{Z} - \mathbb{E}_i[\mathcal{Z}])^2] \leq \sum_{i=1}^{n} \mathbb{E}[(\mathcal{Z} - \mathcal{Z}_i)^2]$$

where

$$\mathcal{Z}_i = g_i(X_1, \dots, X_{i-1}, X_{i+1}, \dots X_n)$$

for arbitrary measurable functions

$$g_i : \mathcal{X}^{n-1} \to \mathbb{R}$$

of $n - 1$ variables. Taking expectations, we get an extension of the Efron-Stein inequality:

$$Var[\mathcal{Z}] \leq \sum_{i=1}^{n} \mathbb{E}[(\mathcal{Z} - \mathcal{Z}_i)^2]$$

Bounded Differences Inequality

1. Functions with Bounded Differences: We say that function

$$g : \mathcal{X}^n \to \mathbb{R}$$

has the *Bounded Differences Property* if for some non-negative constants $c_1, \dots, c_n$

$$\sup_{\substack{x_1, \dots, x_n \\ x'_i \in \mathcal{X}}} |g(x_1, \dots, x_n) - g(x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n)| \leq c_i \; \forall \; 1 \leq i \leq n$$

In other words, if we change the i^{th} variable of g keeping all the others fixed, the value of the function cannot change by more than c_i.

2. Bounded Differences Inequality – Statement: If g has the bounded differences property with constants $c_1, \dots, c_n$, then

$$Var[\mathcal{Z}] \leq \sum_{i=1}^{n} {c_i}^2$$

This bound comes in particularly handy when the direct estimation of the variance becomes involved, but the bound is obtained effortlessly.

a. Bounded Differences with Probability Bounds => If the random variables $X_1, \ldots, X_n$ are independent and binary $\{0, 1\}$-valued with

$$\mathbb{P}\{X_i = 1\} = \mathcal{P}_i$$

and that g has the bounded differences property with constants $c_1, \ldots, c_n$ then it is easy to show that

$$Var[\mathcal{Z}] \leq \sum_{i=1}^{n} {c_i}^2 p_i(1 - p_i)$$

3. Empirical Estimation of Multivariate Martingale Differences Inequality: For a sequence of size s with n multivariates, the empirical estimation goes as $\mathbb{O}(s^n)$. Thus, even for a modest sample of size

$$s = n = 10$$

martingale differences empirical estimates can be $\sim 10^{10}$, i.e., the computational demands blow up very, very fast.

4. Empirical Estimation of Multivariate Symmetrized Differences: Both conceptually as well as empirically/implementation-wise, symmetrized differences inequality (a.k.a. Efron-Stein-Steele Multivariate Variance Upper Bound Inequality) is easier to handle, as drawing a parallel i.i.d. ghost variable set is easier than computing conditional multivariate sequence metrics (e.g., expectation/variance). Of course, the multivariate bounded differences inequality is the easiest to work with, but has the loosest bound.

Bounded Differences Inequality - Applications

1. Application #1 - Bin Packing:
   - One of the basic problems in operations research is as follows: Given n numbers

$$x_1, \dots, x_n \in [0,1]$$

     the question is the following: what is the minimal number of bins into which these numbers can be packed such that the sum of the numbers in each bind doesn't exceed one?
   - Let $g(x_1, \dots, x_n)$ be this number. The behavior of

$$\mathcal{Z} = g(X_1, \dots, X_n)$$

     when $X_1, \dots, X_n$ are independent has been extensively studied (Rhee and Talagrand (1987), Rhee (1993), Talagrand (1995)). By changing one of the x_i's, the value of $g(x_1, \dots, x_n)$ cannot change by more than one, so we have

$$Var[\mathcal{Z}] \leq \frac{n}{2}$$

     However, sharper bounds may be established using Talagrand's convex distance inequality discussed later.
5. Longest Common Sub-sequence: This problem has been treated in depth in Chvatal and Sankoff (1975), Deken (1979), Steele (1982), Dancik and Peterson (1994), and Steele (1996). The simplest version is the following: Let $X_1, \dots, X_n$ and $Y_1, \dots, Y_n$ be 2 sequences of coin flips. Define $\mathcal{Z}$ as the length of longest common sub-sequence that appears in both sequences, i.e.,

$$\mathcal{Z} = max\{k: X_{i_1} = Y_{j_1}, \dots, X_{i_k} = Y_{j_k};\ 1 \le i_1 < \cdots < i_k \le \text{n};\ 1 \le j_1 < \cdots < j_k \le \text{n}\}$$

a. Behavior of the Expectation => The behavior of $\mathbb{E}[\mathcal{Z}]$ has been investigated in many papers. It is known that $\frac{\mathbb{E}[\mathcal{Z}]}{n}$ converges to some number γ whose value is unknown, but conjectured to be $\frac{2}{1+\sqrt{2}}$, and is known to fall between 0.75796 and 0.83763.

b. Variance of $\mathcal{Z}$ => Changing one bit cannot change the length of the longest common sub-sequence by more than one, so $\mathcal{Z}$ satisfies the bounded differences inequality with

$$c_i = 1$$

Thus

$$Var[\mathcal{Z}] \le \frac{n}{2}$$

(Steele (1986)). Thus, by Chebyshev's inequality, with large probability, $\mathcal{Z}$ is within a constant times $\sqrt{n}$ of its expected value. In other words, it is strongly concentrated around the mean, which means that the results on $\mathbb{E}[\mathcal{Z}]$ really tell us about the behavior of the longest common sub-sequence of two random strings.

6. Uniform Deviations of Indicator Functions: Let $X_1, \dots, X_n$ be i.i.d. random variables taking their values in some set $\mathcal{X}$, and let $\mathcal{A}$ be a collection of sub-sets of $\mathcal{X}$. Let μ be a distribution of X_1, i.e.,

$$\mu_n(\mathcal{A}) = \mathbb{P}\{X_1 \in \mathcal{A}\}$$

and let μ_n denote the empirical distribution

$$\mu_n(\mathcal{A}) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{I}_{X_i \in \mathcal{A}}$$

The quantity of interest is

$$\mathcal{Z} = \underset{A \in \mathcal{A}}{sup} |\mu_n(A) - \mu(A)|$$

a. Variance Bound of $\mathcal{Z}$ => If

$$\underset{n \to \infty}{Limit} \mathbb{E}[\mathcal{Z}] = 0$$

for every distribution of X_i, then $\mathcal{A}$ is called a *Uniform Glivenko-Cantelli Class*, and Vapnik and Chervonenkis (1971) provide a combinatorial characterization of such classes. But regardless of $\mathcal{A}$, by changing a single X_i, $\mathcal{Z}$ can change by at most $\frac{1}{n}$, so regardless of the behavior of $\mathbb{E}[\mathcal{Z}]$, we always have

$$Var[\mathcal{Z}] \leq \frac{1}{2n}$$

(in other words, $\mathcal{Z}$ need not be uniform). More information on the behavior of $\mathcal{Z}$ and its role in learning theory may be found in Devroye, Gyorfi, and Lugosi (1996), van der Waart and Wellner (1996), Vapnik (1998), and Dudley (1999).

b. Efron-Stein Bound Motivation => Next we show how a closer look at the Efron-Stein inequality implies a significantly better bound for the variance of $\mathcal{Z}$. We do this in a slightly more general framework of the empirical process.

c. Efron-Stein Bound Formulation => Let $\mathcal{F}$ be a class of real-valued functions (no boundedness is assumed), and define

$$\mathcal{Z} = g(X_1, \ldots, X_n) = \underset{f \in \mathcal{F}}{sup} \sum_{j=1}^{n} f(X_j)$$

Observe that by symmetry, the Efron-Stein bound may be written as

$$Var(\mathcal{Z}) \leq \frac{1}{2}\sum_{i=1}^{n}\mathbb{E}[(\mathcal{Z}-\mathcal{Z}'_i)^2] = \sum_{i=1}^{n}\mathbb{E}\left[(\mathcal{Z}-\mathcal{Z}'_i)^2\mathbb{I}_{\mathcal{Z}'_i<\mathcal{Z}}\right]$$

Let

$$f^* \in \mathcal{F}$$

denote the function that achieves the supremum in the definition of $\mathcal{Z}$, that is,

$$\mathcal{Z} = \sum_{j=1}^{n} f^*(X_j)$$

Then, clearly,

$$(\mathcal{Z}-\mathcal{Z}'_i)^2\mathbb{I}_{\mathcal{Z}'_i<\mathcal{Z}} \leq [f^*(X_i)-f^*(X'_i)]^2$$

and therefore

$$Var(\mathcal{Z}) \leq \mathbb{E}\left[\sup_{f\in\mathcal{F}}\sum_{j=1}^{n}\{f(X_j)-f(X'_j)\}^2\right] \leq 4\mathbb{E}\left[\sup_{f\in\mathcal{F}}\sum_{j=1}^{n}f(X_j)^2\right]$$

d. Efron-Stein Bound vs. Simple Bounded Differences => For functions

$$f \in \mathcal{F}$$

taking values in $[0,1]$, from just the bounded differences property we derived

$$Var[\mathcal{Z}] \leq 2n$$

The new bound may be a significant improvement whenever the maximum of $\sum_{j=1}^{n} f(X_j)^2$ of the functions in $\mathcal{F}$ is small. More importantly, in deriving the new bounds we have not assumed any boundedness of the functions in $\mathcal{F}$.

e. Exponential Tail Inequality => The exponential tail inequality due to Talagrand (1996a) extends this variance inequality, and is one of the most important recent results in the theory of empirical processes, see also Ledoux (1997), Massart (2000a), Rio (2001), and Bousquet (2002).

7. First Passage Time in Oriented Percolation: Consider a directed graph such that the weight X_i is associated with the edge e_i such that the X_i are non-negative independent random variables with a second moment

$$\mathbb{E}[{X_i}^2] = {\sigma_i}^2$$

Let v_1 and v_2 be fixed vertices of the graph. We are interests in the total weight of the path from v_1 to v_2 with minimum weight. Set

$$\mathcal{Z} = \min_{\mathbb{P}} \sum_{e_i \in \mathbb{P}} X_i$$

where the minimum is taken over all the paths $\mathbb{P}$ from v_1 to v_2.

a. Denote the optimal path by $\mathbb{P}^*$. By replacing X_i with ${X'}_i$, we can see that the total minimum weight can only increase if the edge e_i is on $\mathbb{P}^*$, and therefore

$$(\mathcal{Z} - {\mathcal{Z}'}_i)^2 \mathbb{I}_{{\mathcal{Z}'}_i < \mathcal{Z}} \leq [X_i - {X'}_i]^2 \mathbb{I}_{e_i \in \mathbb{P}^*}$$

Thus,

$$Var[\mathcal{Z}] \leq \mathbb{E}\left[\sum_i X'_i \mathbb{I}_{e_i \in \mathbb{P}^*}\right] \leq \sigma^2 \mathbb{E}\left[\sum_i \mathbb{I}_{e_i \in \mathbb{P}^*}\right] \leq \sigma^2 L$$

where L is the length of the longest path between v_1 and v_2.

8. Minimum of the Empirical Loss: Concentration inequalities have been used as a key tool in recent developments of model selection methods in statistical learning theory. For the background, we refer to the recent work of Massart (2000b), Koltchinskii and Panchenko (2002), Bartlett, Boucheron, and Lugosi (2002), Bousquet (2003), and Lugosi and Wegkamp (2004).
    a. Efron-Stein Formulation => Let $\mathcal{F}$ denote a class of $[0, 1]$-valued functions on some space $\mathcal{X}$. For simplicity of exposition we assume that $\mathcal{F}$ is finite, although the results below remain true as long as the measurability issues are taken care of. Given an i.i.d. sample

$$\mathcal{D}_n = \{\langle X_i, Y_i \rangle\}_{i \leq n}$$

of n pairs of random variables $\langle X_i, Y_i \rangle$ taking values in $\mathcal{X} \times \{0, 1\}$, for each $f \in \mathcal{F}$ we define the empirical loss

$$\mathcal{L}_n(f) = \frac{1}{n} \sum_{i=1}^{n} l[f(X_i), Y_i]$$

where the loss functional l is defined on $\{0, 1\}^2$ by

$$l(y, y') = \mathbb{I}_{y \neq y'}$$

    b. Efron-Stein Bound – Steps => In non-parametric classification and learning theory, it is common to select an element

$$f \in \mathcal{F}$$

by minimizing the empirical loss. The quantity of interest in this section is the minimal empirical loss

$$\hat{\mathcal{L}} = \inf_{f \in \mathcal{F}} \mathcal{L}_n(f)$$

The bounded Efron-Stein bound implies that

$$Var[\hat{\mathcal{L}}] \leq \frac{1}{2n}$$

However, a more careful application of the Efron-Stein inequality reveals that $\hat{\mathcal{L}}$ may be much more concentrated than predicted by this simple inequality. Getting tight results for the fluctuations of $\hat{\mathcal{L}}$ provides better insight into the calibration of penalties in certain model selection methods.

c. Tighter Bounding of the Variance => Let

$$\mathcal{Z} = n\hat{\mathcal{L}}$$

and let $\mathcal{Z}'_i$ be defined as in the Efron-Stein inequality, that is,

$$\mathcal{Z}'_i = \inf_{f \in \mathcal{F}} \left\{ \sum_{j \neq i} l[f(X_j), Y_j] + l[f(X_i), Y_i] \right\}$$

where $\langle X'_i, Y'_i \rangle$ is independent of D_n and has the same distribution as $\langle X_i, Y_i \rangle$. The convenient form of the Efron-Stein inequality to use is the following:

$$Var[\mathcal{Z}] \leq \frac{1}{2} \sum_{i=1}^{n} \mathbb{E}[(\mathcal{Z} - \mathcal{Z}'_i)^2] = \sum_{i=1}^{n} \mathbb{E}\left[(\mathcal{Z} - \mathcal{Z}'_i)^2 \mathbb{I}_{\mathcal{Z}'_i > \mathcal{Z}}\right]$$

d. Tighter Efron-Stein Bound => Let f^* denote a possibly non-unique minimizer of the empirical risk so that

$$\mathcal{Z} = \sum_{i=1}^{n} l[f(X_i), Y_i]$$

The key observation is that

$$(\mathcal{Z} - \mathcal{Z}'_i)^2 \mathbb{I}_{\mathcal{Z}'_i > \mathcal{Z}} = \{l[f^*(X_i), Y_i] - l[f^*(X'_i), Y'_i]\}^2 \mathbb{I}_{\mathcal{Z}'_i > \mathcal{Z}}$$
$$= l[f^*(X'_i), Y'_i] \mathbb{I}_{l[f^*(X_i), Y_i]=0}$$

Thus,

$$\sum_{i=1}^{n} \mathbb{E}\left[(\mathcal{Z} - \mathcal{Z}'_i)^2 \mathbb{I}_{\mathcal{Z}'_i > \mathcal{Z}}\right] \leq \sum_{i: l[f^*(X_i), Y_i]=0} \mathbb{E}_{X'_i, Y'_i}\left[l[f^*(X'_i), Y'_i]\right] \leq n\mathbb{E}[\mathcal{L}(f^*)]$$

where $\mathbb{E}_{X'_i, Y'_i}$ denotes expectation with respect to the variables X'_i and Y'_i, and for each

$$f \in \mathcal{F}$$

$$\mathcal{L}(f) = \mathbb{E}\left[l[f(X), Y]\right]$$

is the true expected of f. Therefore, the Efron-Stein inequality implies that

$$Var\left[\hat{\mathcal{L}}\right] \leq \frac{\mathbb{E}[\mathcal{L}(f^*)]}{n}$$

e. Improvement over the Naïve Efron-Stein Bound => The result above is a significant improvement over the bound $\frac{1}{2n}$ whenever $\mathbb{E}[\mathcal{L}(f^*)]$ is much smaller than $\frac{1}{2}$. This is very often the case. For example, we have

$$\mathcal{L}(f^*) = \hat{\mathcal{L}} - [\mathcal{L}_n(f^*) - \mathcal{L}(f^*)] \leq \frac{\mathcal{Z}}{n} + \sup_{f \in \mathcal{F}} \frac{\mathcal{L}_n(f) - \mathcal{L}(f)}{n}$$

so that we obtain

$$Var[\hat{\mathcal{L}}] \leq \frac{\mathbb{E}[\hat{\mathcal{L}}]}{n} + \mathbb{E}\left[\sup_{f \in \mathcal{F}} \frac{\mathcal{L}_n(f) - \mathcal{L}(f)}{n}\right]$$

f. Bounding $\mathbb{E}\left[\sup_{f \in \mathcal{F}} \frac{\mathcal{L}_n(f) - \mathcal{L}(f)}{n}\right]$ => In most cases of interest, $\mathbb{E}\left[\sup_{f \in \mathcal{F}} \frac{\mathcal{L}_n(f) - \mathcal{L}(f)}{n}\right]$ may be bounded by a constant term depending on $\mathcal{F}$ times $n^{-\frac{1}{2}}$ (Lugosi (2002)), and the second term on the RHS is of the order of $n^{-\frac{3}{2}}$. Boucheron, Lugosi, and Massart (2003) detail exponential concentration inequalities for $\hat{\mathcal{L}}$.

9. Kernel Density Estimation: Let $X_1, \ldots, X_n$ be i.i.d. samples drawn according to some unknown density function f on the real line. The density is estimated by the kernel estimate

$$f_n(x) = \frac{1}{nh} \sum_{i=1}^{n} \mathcal{K}\left(\frac{x - X_i}{h}\right)$$

where

$$h > 0$$

is a smoothing parameter, and $\mathcal{K}$ is a non-negative function with

$$\int \mathcal{K} = 1$$

The performance of the estimate is measured by the L_1 error

$$\mathcal{Z} = g(X_1, \dots, X_n) = \int |f(x) - f_n(x)| dx$$

a. Bounding the Kernel Density Estimate => It is easy to see that

$$|g(x_1, \dots, x_i, \dots, x_n) - g(x_1, \dots, x'_i, \dots, x_n)| \leq \frac{1}{nh} \int \left| \mathcal{K}\left(\frac{x - X_i}{h}\right) - \mathcal{K}\left(\frac{x - X'_i}{h}\right) \right| dx$$

so without further work, we get

$$Var[\mathcal{Z}] \leq \frac{2}{n}$$

b. L_1 Bounds – Bounding the L_1 Error => It is known that for every f,

$$\sqrt{n}\mathbb{E}[g] \to \infty$$

(Devroye and Gyorfi (1985)) which implies, by Chebyshev's inequality, that for every

$$\varepsilon > 0$$

$$\mathbb{P}\left\{\left|\frac{\mathcal{Z}}{\mathbb{E}[\mathcal{Z}]} - 1\right| \geq \varepsilon\right\} = \mathbb{P}\{|\mathcal{Z} - \mathbb{E}[\mathcal{Z}]| \geq \varepsilon\mathbb{E}[\mathcal{Z}]\} \leq \frac{Var[\mathcal{Z}]}{\varepsilon^2(\mathbb{E}[\mathcal{Z}])^2} \to 0$$

as

$$n \to \infty$$

That is,

$$\frac{\mathcal{Z}}{\mathbb{E}[\mathcal{Z}]} \to 1$$

in probability, or in other words, $\mathcal{Z}$ is *relatively stable*. This means that the random L_1-error behaves like its expected value (Devroye (1988, 1991)). For more on the behavior of the L_1-error of the kernel density estimate, we refer to Devroye and Gyorfi (1985) and Devroye and Lugosi (2000).

Self-Bounding Functions

1. Introduction: Another simple property that is satisfied in many important functions is the *Self-Bounding Property*. We say that a non-negative function

$$g: \mathcal{X}^n \to \mathbb{R}$$

has the self-bounding property if there exists

$$g: \mathcal{X}^{n-1} \to \mathbb{R}$$

such that for all

$$x_1, \dots, x_n \in \mathcal{X}$$

and for all

$$i = 1, \dots, n$$

$$0 \leq g(x_1, \dots, x_n) - g_i(x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n) \leq 1$$

and also

$$\sum_{i=1}^{n}[g(x_1,\dots,x_n)-g_i(x_1,\dots,x_{i-1},x_{i+1},\dots,x_n)]\le g(x_1,\dots,x_n)$$

a. Concentration Properties => Concentration properties for such functions have been studied by Boucheron, Lugosi, and Massart (2000), Rio (2001), Bousquet (2002). For self-bounding functions we clearly have

$$\sum_{i=1}^{n}[g(x_1,\dots,x_n)-g_i(x_1,\dots,x_{i-1},x_{i+1},\dots,x_n)]^2\le g(x_1,\dots,x_n)$$

b. Variance Bound for Self-Bounding Functions => Using the extended Efron-Stein inequality

$$Var[\mathcal{Z}]\le\sum_{i=1}^{n}\mathbb{E}[(\mathcal{Z}-\mathcal{Z}_i)^2]$$

and setting

$$\mathcal{Z}=g(x_1,\dots,x_n)$$

and

$$\mathcal{Z}_i=g(x_1,\dots x_{i-1},x_{i+1},\dots,x_n)$$

on applying the previous observation, and finally, using the squared form above, we get

$$Var[\mathcal{Z}] \leq \mathbb{E}\left[\sum_{i=1}^{n}[g(x_1,\dots,x_n) - g_i(x_1,\dots,x_{i-1},x_{i+1},\dots,x_n)]\right] \leq \mathbb{E}[g(x_1,\dots,x_n)]$$
$$\leq \mathbb{E}[\mathcal{Z}]$$

It turns out that in many cases, the obtained bound form

$$Var[\mathcal{Z}] \leq \mathbb{E}[\mathcal{Z}]$$

is a significant improvement over what may be obtained by the bounded differences inequality.

2. Relative Stability: Bounding the variance of $\mathcal{Z}$ in many cases by its expected value implies the relative stability of $\mathcal{Z}$. A sequence of non-negative random variables $\mathcal{Z}_n$ is said to be relatively stable if

$$\frac{\mathcal{Z}}{\mathbb{E}[\mathcal{Z}]} \to 1$$

in probability. This property guarantees that the random fluctuations of $\mathcal{Z}_n$ around its expectation are of negligible size when compared to the expectation, and therefore the most important information about the size of $\mathcal{Z}_n$ is given by $\mathbb{E}[\mathcal{Z}_n]$. As seen above, if $\mathcal{Z}_n$ has the self-bounding property, then, by Chebyshev's inequality, for all

$$\varepsilon > 0$$

$$\mathbb{P}\left\{\left|\frac{\mathcal{Z}_n}{\mathbb{E}[\mathcal{Z}_n]} - 1\right| \geq \varepsilon\right\} \leq \frac{Var[\mathcal{Z}_n]}{\varepsilon^2(\mathbb{E}[\mathcal{Z}_n])^2} = \frac{1}{\varepsilon^2}$$

Thus, for relative stability, it suffices to have

$$\mathbb{E}[\mathcal{Z}_n] \to \infty$$

3. Empirical Processes: A typical example of self-bounding functions is the supremum of non-negative empirical processes. Let $\mathcal{F}$ be a class of functions taking values in $[0,1]$, and consider

$$\mathcal{Z} = g(X_1, \dots, X_n) = \sup_{f \in \mathcal{F}} \sum_{i=1}^{n} f(X_i)$$

A special case of the above is mentioned in the example of uniform deviations.

a. Formulation => Define

$$g_i = g' \forall\, i = 1, \dots, n$$

with

$$g(x_1, \dots, x_n) = \sup_{f \in \mathcal{F}} \sum_{i=1}^{n-1} f(X_i)$$

so that

$$\mathcal{Z}_i = \sup_{f \in \mathcal{F}} \sum_{j=1, j \neq i}^{n} f(X_j)$$

Let

$$f^* \in \mathcal{F}$$

be a function for which

$$\mathcal{Z} = \sum_{j=1}^{n} f^*(X_j)$$

one obviously has

$$0 \leq \mathcal{Z} - \mathcal{Z}_i \leq f^*(X_i) \leq 1$$

and therefore

$$\sum_{i=1}^{n} (\mathcal{Z} - \mathcal{Z}_i) \leq \sum_{i=1}^{n} f^*(X_i) = \mathcal{Z}$$

b. Variance Bounding => Here we have assumed that the supremum is always achieved. The modification of the argument for the general case is straightforward. Thus by the variance bound, we get

$$Var[\mathcal{Z}] \leq \mathbb{E}[\mathcal{Z}]$$

Note that the bounded differences inequality implies

$$Var[\mathcal{Z}] \leq \frac{n}{2}$$

In some applications, $\mathbb{E}[\mathcal{Z}]$ may be significantly smaller than $\frac{n}{2}$, and the improvement is essential.

4. Rademacher Averages: A less trivial example for the self-bounding functions is the Rademacher complexity. Let $\mathcal{F}$ be a class of functions taking values in $[0, 1]$. If $\sigma_1, \ldots, \sigma_n$ denote independent symmetric $\{-1, +1\}$-valued random variables independent of the X_i's (the so-called Rademacher variables), we define the *Conditional Rademacher Average* as

$$\mathcal{Z} = \mathbb{E}_{\sigma}\left[\sup_{f \in \mathcal{F}} \sum_{j=1}^{n} \{\sigma_j f(X_j) | X_1{}^n\}\right]$$

Thus, the expected values are taken with respect to the Rademacher variables σ_i, and $\mathcal{Z}$ is a function therefore of X_i's - this makes $\mathcal{Z}$ data dependent.

a. Learning Class Complexity => Quantities like $\mathcal{Z}$ have been known to measure effectively the complexity of model classes in statistical learning theory (Koltchinskii (2001), Bartlett, Boucheron, and Lugosi (2002), Bartlett and Mendelson (2002), and Bartlett, Bousquet, and Mendelson (2002)).

b. Variance Bounding => It is immediately apparent that $\mathcal{Z}$ has the bounded differences property, and therefore the bounded differences inequality implies that

$$Var[\mathcal{Z}] \leq \frac{n}{2}$$

However, this bound may be improved by observing that $\mathcal{Z}$ also has the self-bounding property, and therefore

$$Var[\mathcal{Z}] \leq \mathbb{E}[\mathcal{Z}]$$

Indeed, defining

$$\mathcal{Z} = \mathbb{E}_{\sigma}\left[\sup_{f \in \mathcal{F}} \sum_{j=1}^{n} \{\sigma_j f(X_j) | X_1{}^n\}\right]$$

it is easy to see that

$$0 \leq \mathcal{Z} - \mathcal{Z}_i \leq 1$$

and

$$\sum_{i=1}^{n}(\mathcal{Z}-\mathcal{Z}_i)\leq\mathcal{Z}$$

c. Improving the Variance Bound => The above improvement is essential since it is well-known in the empirical process theory and the statistical learning theory that in many cases (where $\mathcal{F}$ is a relatively small class of functions), $\mathbb{E}[\mathcal{Z}]$ may be bounded by something like $Cn^{\frac{1}{2}}$ where the constant C depends on the class $\mathcal{F}$ (van der Waart and Wellner (1996), Vapnik (1998), and Dudley (1999)).

Configuration Functions

1. Introduction: An important class of functions satisfying the self-bounding property consists of the so-called *Configuration Functions* defined in Talagrand (1995). The definitions presented here are a slight modification of Talagrand's (Boucheron, Lugosi, and Massart (2000)).
   a. Definition and Setup => Assume that we have a property $\mathcal{P}$ defined over a union of finite products of a set $\mathcal{X}$, that is, a sequence of sets

$$\mathcal{P}_1\subset\mathcal{X}$$

$$\mathcal{P}_2\subset\mathcal{X}\times\mathcal{X}$$

$$\ldots$$

$$\mathcal{P}_n\subset\mathcal{X}^n$$

We say that

$$(x_1, \dots, x_m) \in \mathcal{X}^m$$

satisfies the property $\mathcal{P}$ if

$$(x_1, \dots, x_m) \in \mathcal{P}_m$$

We assume that $\mathcal{P}$ is hereditary in the sense that if $(x_1, \dots, x_m)$ satisfies $\mathcal{P}_m$, so does any sub-sequence $(x_{i_1}, \dots, x_{i_k})$ of $(x_1, \dots, x_m)$. The function g_n that maps any tuple $(x_1, \dots, x_n)$ to the size of the largest sub-sequence satisfying $\mathcal{P}$ is the *Configuration Function* associated with $\mathcal{P}$.

b. Bound => Let g_n be a configuration function, and let

$$\mathcal{Z} = g_n(X_1, \dots, X_n)$$

where $X_1, \dots, X_n$ are independent random variables. Then

$$Var[\mathcal{Z}] \leq \mathbb{E}[\mathcal{Z}]$$

c. Bound Derivation => In order to apply the self-bounding function bound, it suffices to show that any configuration function is self-bounding. Let

$$\mathcal{Z}_i = g_{n-1}(X_1, \dots, X_{i-1}, X_{i+1}, \dots, X_n)$$

The condition

$$0 \leq \mathcal{Z} - \mathcal{Z}_i \leq 1$$

is trivially satisfied. On the other hand, assume that

$$\mathcal{Z} = k$$

and let

$$\left(X_{i_1}, \ldots, X_{i_k}\right) \subset (X_1, \ldots, X_n)$$

be a sub-sequence of cardinality k such that

$$g_k\left(X_{i_1}, \ldots, X_{i_k}\right) = k$$

Clearly if the index i is such that

$$i \notin \{i_1, \ldots, i_k\}$$

Then

$$\mathcal{Z} = \mathcal{Z}_i$$

and therefore

$$\sum_{i=1}^{n} (\mathcal{Z} - \mathcal{Z}_i) \leq \mathcal{Z}$$

is also satisfied, which concludes the proof.

2. Number of Distinct Values in a Discrete Sample: Let $X_1, \ldots, X_n$ be independent, identically distributed random variables taking values on a set of positive integers such that

$$\mathbb{P}\{X_1 = k\} = \mathcal{P}_k$$

and let $\mathcal{Z}$ denote the number of distinct values taken by these n random variables. Then we may write

$$\mathcal{Z} = \sum_{i=1}^{n} \mathbb{I}_{\{X_1 \neq X_i, \ldots, X_{i-1} \neq X_i\}}$$

and, thus, the expected value of $\mathcal{Z}$ is computed easily as

$$\mathbb{E}[\mathcal{Z}] = \sum_{i=1}^{n} \sum_{j=1}^{\infty} (1 - p_j)^{i-1} p_j$$

a. Basic Bounds => It is easy to see that

$$\frac{\mathbb{E}[\mathcal{Z}]}{n} \to 0$$

as

$$n \to \infty$$

In fact, the variance of $\mathcal{Z}$ may also be computed explicitly. Clearly $\mathcal{Z}$ satisfies the bounded differences property with

$$c_i = 1$$

so the bounded differences inequality implies

$$Var[\mathcal{Z}] \leq \frac{n}{2}$$

so, by Chebyshev's inequality

$$\frac{\mathcal{Z}}{n} \to 0$$

in probability.

b. Configuration Function Bounds => On the other hand, it is obvious that $\mathcal{Z}$ is a configuration function associated with the property of "distinctness", and by the Bounding of the Configuration Function we have

$$Var[\mathcal{Z}] \leq \mathbb{E}[\mathcal{Z}]$$

which is a significant improvement since

$$\frac{\mathbb{E}[\mathcal{Z}]}{n} \to 0$$

as

$$n \to \infty$$

3. VC Dimension: One of the central quantities in statistical learning theory is the *Vapnik-Chervonenkis Dimension* (Vapnik and Chervonenkis (1971, 1974), Blumer, Ehrenfeucht, Haussler, and Warmuth (1989), Devroye, Gyorfi, and Lugosi (1996), Vapnik (1998), Anthony and Bartlett (1999)).

a. Setup => Let $\mathcal{A}$ be an arbitrary collection of subsets of $\mathcal{X}$, and let

$$x_1{}^n = (x_1, \ldots, x_n)$$

be a vector n points of $\mathcal{X}$. Define the *Trace* of $\mathcal{A}$ by

$$Tr(x_1{}^n) = \{A \cap [x_1, \ldots, x_n] : A \in \mathcal{A}\}$$

The *Shatter Coefficient* (or *Vapnik-Chervonenkis Growth Function*) of $\mathcal{A}$ in $x_1{}^n$ is

$$\mathcal{T}(x_1{}^n) = |Tr(x_1{}^n)|$$

the size of the trace. $\mathcal{T}(x_1{}^n)$ is the number of different sub-sets of the n -point set generated by intersecting it with the elements of $\mathcal{A}$.

b. Formal Definition => A subset $(x_{i_1}, \ldots, x_{i_k})$ of $(x_1, \ldots, x_n)$ is said to be *shattered* if

$$\mathcal{T}(x_{i_1}, \ldots, x_{i_k}) = 2^k$$

The VC Dimension $\mathcal{D}(x_1{}^n)$ of $\mathcal{A}$ (with respect to $x_1{}^n$) is the cardinality of the largest subset of $x_1{}^n$.

c. Configuration Function Bounding => From the definition, it is obvious that

$$g_n(x_1{}^n) = \mathcal{D}(x_1{}^n)$$

is a configuration function associated with the property of "shatteredness", and therefore if $X_1, \ldots, X_n$ are independent random variables, then

$$Var[\mathcal{D}(x_1{}^n)] \leq \mathbb{E}[\mathcal{D}(x_1{}^n)]$$

4. Increasing Sub-sequences: Consider a vector

$$x_1{}^n = (x_1, \ldots, x_n)$$

of n different numbers in $[0, 1]$. The positive integers

$$i_1 < i_2 < \cdots < i_m$$

form an *increasing sub-sequence* if

$$x_{i_1} < x_{i_2} < \cdots < x_{i_m}$$

where

$$i_1 \geq 1$$

and

$$i_m \leq n$$

a. Configuration Function Bound => Let $L(x_1{}^n)$ denote the length of the longest increasing sub-sequence.

$$g_n(x_1{}^n) = L(x_1{}^n)$$

is clearly a configuration function associated with the "increasing sequence" property, and therefore if $X_1, \ldots, X_n$ are independent random variables such that they are all different with probability one (it suffices if every X_i has an absolutely continuous distribution), then

$$Var[L(x_1{}^n)] \leq \mathbb{E}[L(x_1{}^n)]$$

b. $\mathbb{E}[L(x_1{}^n)]$ Estimate => If the X_i's are uniformly distributed in $[0, 1]$, it is known that

$$\mathbb{E}[L(x_1{}^n)] \approx 2\sqrt{n}$$

(Logan and Shepp (1977), Groeneboom (2002)). A tighter, but more difficult result, obtained implies that

$$Var[L(x_1{}^n)] \leq \mathcal{O}\left(n^{\frac{1}{3}}\right)$$

(Baik, Deift, and Johansson (2000)). Frieze (1991), Bollobas and Brightwell (1992), and Talagrand (1995) contain reviews on some of the early results on the concentration of $L(X)$.

References

- Anthony, M., and P. L. Bartlett (1999): *Neural Network Learning: Theoretical Foundations* **Cambridge University Press** Cambridge.
- Baik, L., P. Deift, and K. Johansson (2000): On the Distribution of the Length of the Second Row of a Young Diagram under Plancherel Measure *Geometric and Functional Analysis* **10** 702-731.
- Bartlett, P., S. Boucheron, and G. Lugosi (2002): Model Selection and Error Estimation *Machine Learning* **48** 85-113.
- Bartlett, P., O. Bousquet, and S. Mendelson (2002): Localized Rademacher Complexities *Proceedings of the 15th Annual Conference on Machine Learning* 44-48.
- Bartlett, P., and S. Mendelson (2002): Rademacher and Gaussian Complexities: Risk Bounds and Structural Results *Journal of Machine Learning Research* **3** 463-482.
- Blumer, A., A. Ehrenfeucht, D. Haussler, and M. K. Warmuth (1989): Learnability and the Vapnik-Chervonenkis Dimension *Journal of the ACM* **36** 929-965.
- Bollobas, B., and G. Brightwell (1992): The Height of Random Partial Order: Concentration of Measure *Annals of Applied Probability* **2** 1009-1018.
- Boucheron, S., G. Lugosi, and P. Massart (2000): A Sharp Concentration Inequality with Applications *Random Structures and Algorithms* **16** 277-292.
- Boucheron, S., G. Lugosi, and P. Massart (2003): Concentration Inequalities using the Entropy Method *Annals of Probability* **31** 1583-1614.
- Bousquet, O. (2002): A Bennett Concentration Inequality and its Application to Suprema of Empirical Processes *C. R. Acad. Sci. Paris* **334** 495-500.
- Bousquet, O. (2003): New Approaches to Statistical Learning Theory *Annals of the Institute of Statistical Mathematics*.

- Chvatal, V., and D. Sankoff (1975): Longest common sub-sequences of two random sequences *Journal of applied Probability* **12** 306-315.
- Dancik, V. and M. Paterson (1994): Upper Bound for the Expected, in: *Proceedings of STACS '94: Lecture Notes in Computer Science* **775** 669-678 **Springer** New York.
- Deken, J. P. (1979): Some Limit Results for Longest Common Sub-sequences *Discrete Mathematics* **26** 17-31.
- Devroye, L., and L. Gyorfi (1985): *Non-parametric Density Estimation: The L_1-View* **John Wiley** New York.
- Devroye, L. (1988): The Kernel Estimate is Relatively Stable *Probability Theory and Related Fields* **77** 521-536.
- Devroye, L. (1991): Exponential Inequalities in Non-parametric Estimation, in: *Non-parametric Functional Estimation and Related Topics* (editor: G. Roussas) 31-44 NATO ASI Series **Kluwer Academic Publishers** Dordrecht.
- Devroye, L., L. Gyorfi, and G. Lugosi (1996): *A Probabilistic Theory of Pattern Recognition* **Springer-Verlag** New York.
- Devroye, L., and G. Lugosi (2000): *Combinatorial Methods in Density Estimation* **Springer-Verlag** New York.
- Dudley, R. M. (1999): *Uniform Central Limit Theorems* **Cambridge University Press** Cambridge.
- Efron, B., and C. Stein (1981): The Jack-knife Estimate of variance *Annals of Statistics* **9** 586-596.
- Frieze, A. M. (1991): On the Length of the Longest Monotone Sub-sequence in a Random Permutation *Annals of Applied Probability* **1** 301-305.
- Groeneboom, P. (2002): Hydro-dynamical Methods for Analyzing Longest Increasing Sub-sequences; Probabilistic Methods in Combinatorics and Combinatorial Optimization *Journal of Computational and Applied Mathematics* **142** 83-105.
- Koltchinskii, V. (2001): Rademacher Penalties and Structural Risk Minimization *IEEE Transactions on Information Theory* **47** 1902-1914.
- Koltchinskii, V., and D. Panchenko (2002): Empirical Margin Distributions and Bounding the Generalization Error of Combined Classifiers *Annals of Statistics* **30**.

- Ledoux, M. (1997): On Talagrand's Deviation Inequalities for Product Measures *ESAIM: Probability and Statistics* **1** 63-87.
- Logan, B. F., and L. A. Shepp (1977): A Variational Problem for the Young Tableaux **26** 206-222.
- Lugosi, G. (2002): Pattern Classification and Learning Theory, in: *Principles of Non-Parametric Learning* (editor: L. Gyorfi) **Springer** Wien.
- Lugosi, G., and M. Wegkamp (2004): Complexity Regularization via Localized Random Penalties *Annals of Statistics* **32** 1679-1697.
- Massart, P. (2000a): About the Constants in Talagrand's Concentration Inequalities for Empirical Processes *Annals of Probability* **28** 863-884.
- Massart, P. (2000b): Some Applications of Concentration Inequalities to Statistics *Annales de la Faculte des Sciences de Toulouse* **IX** 245-303.
- Rhee, W., and M. Talagrand (1987): Martingales, Inequalities, and NP-Complete Problems *Mathematics of Operations Research* **12** 177-181.
- Rhee, W. (1993): A Matching Problem and Sub-additive Euclidean Functionals *Annals of Applied Probability* **3** 794-801.
- Rio, E. (2001): Inegalities de concentration pour les processus empiriques de classes de parties *Probability Theory and Related Fields* **119** 163-175.
- Steele, J. M. (1982): Long Common Sub-sequences and the Proximity of Two Random Strings *SIAM Journal of Applied Mathematics* **42** 731-737.
- Steele, J. M. (1986): An Efron-Stein Inequality for Non-symmetric Statistics *Annals of Statistics* **14** 753-758.
- Steele, J. M. (1996): *Probability Theory and Combinatorial Optimization* **SIAM CBMS-NSF Regional Conference Series in Applied Mathematics** 69, 3600 University City Science Center, Philadelphia, PA 19104.
- Talagrand, M. (1995): Concentration of Measure and Isoperimetric Inequalities in Product Spaces *Publications Mathematiques de l'I.H.E.S* **81** 73-205.
- Talagrand, M. (1996a): New Concentration Inequalities in Product Spaces *Inventiones Mathematicae* **126** 505-563.
- Van der Waart, A. W., and J. A. Wellner (1996): *Weak Convergence and the Empirical Processes* **Springer-Verlag** New York.

- Vapnik, V. N., and A. Y. Chervonenkis (1971): On the Uniform Convergence of Relative Frequencies of Events to their Probabilities *Theory of Probability and Applications* **16** 264-280.
- Vapnik, V. N., and A. Y. Chervonenkis (1974): *Theory of Pattern Recognition* (in Russian) **Nauka** Moscow. German Translation: *Theorie der Zeichenerkennung* **Academie Verlag** Berlin (1979).
- Vapnik, V. N. (1998): *Statistical Learning Theory* **John Wiley** New York.

Entropy Methods

Introduction

1. Motivation: The Efron-Stein based inequalities produce variance-based linear/polynomial tail bounds. However, we can do better – so we seek exponential tail bounds, which leads us to the entropy methods.
2. Past Approaches: Originally, Martingale methods dominated the research (Rhee and Talagrand (1987), Shamir and Spencer (1987), McDiarmid (1989, 1998)), but independently information-theoretic methods were also used with success (Ahlswede, Gacs, and Korner (1977), Marton (1986, 1996a, 1996b), Massart (1998), Samson (2000), Rio (2001)).
3. Talagrand's Induction Method: Talagrand's induction method (Talagrand (1995, 1996a, 1996b)) caused an important breakthrough both in the theory and in the applications of exponential concentration inequalities.
4. The Entropy Method: Here we focus on the so-called "entropy method" based on logarithmic Sobolev inequalities developed by Ledoux (1996, 1997) – see also Bobkov and Ledoux (1997), Massart (2000a), Boucheron, Lugosi, and Massart (2000), Rio (2001), Bousquet (2002), and Boucheron, Lugosi, and Massart (2003). This method makes it possible to derive exponential analogues of the Efron-Stein inequality in possibly the simplest way.

Information Theory - Basics

1. Introduction: Here we summarize some of the basic properties of the entropy of a discrete-valued random variable. For a good introductory book on information theory, we refer to Cover and Thomas (1991).
2. Types of Entropy: Let X be a random variable taking values in a countable set $\mathcal{X}$ with distribution

$$\mathbb{P}(X = x) = p(x), x \in \mathcal{X}$$

The *entropy* of X is defined by

$$\mathcal{H}(X) = \mathbb{E}[-\log p(X)] = -\sum_{x \in \mathcal{X}} p(x) \log p(x)$$

where log denotes the natural logarithm and

$$0 \log 0 = 0$$

If X, Y is a pair of discrete random variables taking values in $\mathcal{X} \times \mathcal{Y}$, the *Joint Entropy* $\mathcal{H}(X, Y)$ of X and Y is defined as the entropy of the pair (X, Y). The *Conditional Entropy* $\mathcal{H}(X \mid Y)$ is defined as

$$\mathcal{H}(X \mid Y) = \mathcal{H}(X, Y) - \mathcal{H}(Y)$$

3. Conditional Entropy Inequality:

$$\begin{aligned}\mathcal{H}(X \mid Y) = \mathcal{H}(X, Y) - \mathcal{H}(Y) &= -\sum_{x \in \mathcal{X}, y \in \mathcal{Y}} p(x, y) \log p(x, y) + \sum_{y \in \mathcal{Y}} p(y) \log p(y) \\ &= -\sum_{x \in \mathcal{X}, y \in \mathcal{Y}} p(x, y)[\log p(x, y) - \log p(y)] = -\sum_{x \in \mathcal{X}, y \in \mathcal{Y}} p(x, y) \log \frac{p(x, y)}{p(y)} \\ &= -\sum_{x \in \mathcal{X}, y \in \mathcal{Y}} p(x, y) \log p(x|y)\end{aligned}$$

Thus

$$\mathcal{H}(X \mid Y) \geq 0$$

4. Chain Rule for Entropy: It is also easy to see that the definition of the conditional entropy remains true conditionally, i.e., for any 3 discrete random variables $X, Y, and\ Z$,

$$\mathcal{H}(X, Y|Z) = \mathcal{H}(Y|Z) + \mathcal{H}(X|Y, Z)$$

This may be easily seen by adding $\mathcal{H}(Z)$ to both sides and by using the definition of conditional entropy. A repeated application of this yields the *Chain Rule for Entropy*: for arbitrary random variables $X_1, \dots, X_n$

$$\mathcal{H}(X_1, \dots, X_n) = \mathcal{H}(X_1) + \mathcal{H}(X_2|X_1) + \mathcal{H}(X_3|X_1, X_2) + \cdots + \mathcal{H}(X_n|X_1, \dots, X_{n-1})$$

5. Kullback-Leibler Divergence: Let $\mathcal{P}$ and $\mathcal{Q}$ be two probability distributions over a countable set $\mathcal{X}$ with a probability mass function p and q respectively. Then the *Kullback-Leibler Divergence* or the *Relative Entropy* of $\mathcal{P}$ and $\mathcal{Q}$ is

$$\mathcal{D}(\mathcal{P}||\mathcal{Q}) = \sum_{x \in \mathcal{X}} p(x) \log \frac{p(x)}{q(x)}$$

6. The Relative Entropy Inequality: Since

$$\log x \leq x - 1$$

$$\mathcal{D}(\mathcal{P}||\mathcal{Q}) = -\sum_{x \in \mathcal{X}} p(x) \log \frac{q(x)}{p(x)} \geq -\sum_{x \in \mathcal{X}} p(x) \left[\frac{q(x)}{p(x)} - 1\right] = -\sum_{x \in \mathcal{X}} [q(x) - p(x)] = 0$$

thus

$$\mathcal{D}(\mathcal{P}||\mathcal{Q}) \geq 0$$

and equals zero if and only if

$$\mathcal{P} = \mathcal{Q}$$

a. Consequence of the relative entropy inequality => If $\mathcal{X}$ is a finite set with $\mathcal{N}$ elements in it, and X is a random variable with distribution $\mathcal{P}$, and we take $\mathcal{Q}$ to be the uniform distribution over $\mathcal{X}$, then

$$\mathcal{D}(\mathcal{P}||\mathcal{Q}) = \sum_{x \in \mathcal{X}} p(x) \log p(x) - \sum_{x \in \mathcal{X}} p(x) \log q(x) = \log \mathcal{N} - \mathcal{H}(X)$$

since

$$q(x) = \frac{1}{\mathcal{N}}$$

and, therefore, the entropy of X never exceeds the logarithm of the cardinality of its range.

7. Conditioning the Relative Entropy: Consider a pair of random variables with joint distribution $\mathcal{P}_{X,Y}$ and marginal distributions $\mathcal{P}_X$ and $\mathcal{P}_Y$. Then

$$\mathcal{D}\left(\mathcal{P}_{X,Y}||\mathcal{P}_X, \mathcal{P}_Y\right) = -\sum_{x \in \mathcal{X}, y \in \mathcal{Y}} p(x,y) \log \frac{p(x,y)}{p(x)p(y)}$$

Observing that

$$p(x|y) = \frac{p(x,y)}{p(y)}$$

$$\mathcal{D}\left(\mathcal{P}_{X,Y}||\mathcal{P}_X,\mathcal{P}_Y\right)\mathcal{D}\left(\mathcal{P}_{X,Y}||\mathcal{P}_X,\mathcal{P}_Y\right) = -\sum_{x\in\mathcal{X},y\in\mathcal{Y}} p(x,y)\log\frac{p(x|y)}{p(y)}$$

$$= -\sum_{x\in\mathcal{X},y\in\mathcal{Y}} p(x,y)\log p(x) + \sum_{x\in\mathcal{X},y\in\mathcal{Y}} p(x,y)\log p(x|y)$$

$$= -\sum_{x\in\mathcal{X}} p(x,y)\log p(x) + \sum_{x\in\mathcal{X},y\in\mathcal{Y}} p(x,y)\log p(x|y) = \mathcal{H}(X) - \mathcal{H}(X|Y)$$

a. Consequence of Conditioning the Relative Entropy => Thus

$$\mathcal{D}\left(\mathcal{P}_{X,Y}||\mathcal{P}_X,\mathcal{P}_Y\right) = \mathcal{H}(X) - \mathcal{H}(X|Y)$$

and the non-negativity of the relative implies that

$$\mathcal{H}(X) \geq \mathcal{H}(X|Y)$$

that is, conditioning reduces entropy. It is similarly easy to see that this fact remains true for conditional entropies as well, that

$$\mathcal{H}(X|Y) \geq \mathcal{H}(X|Y,Z)$$

8. Han's Inequality (Han (1978)): Let $X_1,\ldots,X_n$ be discrete random variables. Then

$$\mathcal{H}(X_1,\ldots,X_n) \leq \frac{1}{n-1}\sum_{i=1}^{n}\mathcal{H}(X_1,\ldots,X_{i-1},X_{i+1},\ldots,X_n)$$

a. Proof => For any

$$i = 1,\ldots,n$$

by the definition of conditional entropy and the fact that conditioning reduces entropy

$$\mathcal{H}(X_1, \dots, X_n) = \mathcal{H}(X_1, \dots, X_{i-1}, X_{i+1}, \dots, X_n) + \mathcal{H}(X_i | X_1, \dots, X_{i-1}, X_{i+1}, \dots, X_n)$$
$$\leq \mathcal{H}(X_1, \dots, X_{i-1}, X_{i+1}, \dots, X_n) + \mathcal{H}(X_i | X_1, \dots, X_{i-1})$$

Summing both sides, we get

$$n\mathcal{H}(X_1, \dots, X_n)$$
$$\leq \sum_{i=1}^{n} \mathcal{H}(X_1, \dots, X_{i-1}, X_{i+1}, \dots, X_n) + \mathcal{H}(X_1) + \mathcal{H}(X_2 | X_1)$$
$$+ \mathcal{H}(X_3 | X_1, X_2) + \cdots + \mathcal{H}(X_n | X_1, \dots, X_{n-1})$$
$$= \sum_{i=1}^{n} \mathcal{H}(X_1, \dots, X_{i-1}, X_{i+1}, \dots, X_n) + \mathcal{H}(X_1, \dots, X_n)$$

Re-arranging we get

$$\mathcal{H}(X_1, \dots, X_n) \leq \frac{1}{n-1} \sum_{i=1}^{n} \mathcal{H}(X_1, \dots, X_{i-1}, X_{i+1}, \dots, X_n)$$

9. Han's Inequality for Relative Entropy:
   a. Setup => Here we follow the layout of Massart (2000b). Let $\mathcal{X}$ be a countable set, and let $\mathcal{P}$ and $\mathcal{Q}$ be probability distributions on $\mathcal{X}^n$ such that

$$\mathcal{P} = \mathcal{P}_1 \times \dots \times \mathcal{P}_n$$

is a product measure. We denote the elements of $\mathcal{X}^n$ by

$$x_1^n = (x_1, \dots, x_n)$$

and denote

$$x^{(i)} = (x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n)$$

for the $n-1$ vector obtained by leaving out the i^{th} component of x_1^n.

b. Marginal Distributions of $\mathcal{P}$ and $\mathcal{Q}$ => Denote by $\mathcal{P}^{(i)}$ and $\mathcal{Q}^{(i)}$ the marginal distributions of $x^{(i)}$ according to $\mathcal{P}$ and $\mathcal{Q}$, that is

$$\mathcal{Q}^{(i)}\left[x^{(i)}\right] = \sum_{x \in \mathcal{X}} \mathcal{Q}(x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n)$$

and

$$\begin{aligned}\mathcal{P}^{(i)}\left[x^{(i)}\right] &= \sum_{x \in \mathcal{X}} \mathcal{P}(x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n) \\ &= \sum_{x \in \mathcal{X}} \mathcal{P}_1(x_1) \ldots \mathcal{P}_{i-1}(x_{i-1}) \mathcal{P}_i(x_i) \mathcal{P}_{i+1}(x_{i+1}) \ldots \mathcal{P}_n(x_n)\end{aligned}$$

Just as we used $\mathcal{P}$ to represent a uniform measure earlier, here we use $\mathcal{P}$ to represent an *independence* product measure, and will be used later.

c. Statement =>

$$\mathcal{D}(\mathcal{Q}||\mathcal{P}) \geq \frac{1}{n-1} \sum_{i=1}^{n} \mathcal{D}\left(\mathcal{Q}^{(i)}||\mathcal{P}^{(i)}\right)$$

or, equivalently

$$\mathcal{D}(\mathcal{Q}||\mathcal{P}) \leq \sum_{i=1}^{n} \left[\mathcal{D}(\mathcal{Q}||\mathcal{P}) - \mathcal{D}\left(\mathcal{Q}^{(i)}||\mathcal{P}^{(i)}\right)\right]$$

d. Proof:

   i. Re-casting Han's Inequality => Han's inequality may be directly re-cast as stating

$$\sum_{x_1^n \in \mathcal{X}^n} \mathcal{Q}(x_1^n) \log \mathcal{Q}(x_1^n) \geq \frac{1}{n-1} \sum_{i=1}^{n} \sum_{x^{(i)} \in \mathcal{X}^{n-1}} \mathcal{Q}^{(i)}\left(x^{(i)}\right) \log \mathcal{Q}^{(i)}\left(x^{(i)}\right)$$

ii. Re-casting the Relative Entropy => Since

$$\mathcal{D}(\mathcal{Q}||\mathcal{P}) = \sum_{x_1^n \in \mathcal{X}^n} \mathcal{Q}(x_1^n) \log \mathcal{Q}(x_1^n) - \sum_{x_1^n \in \mathcal{X}^n} \mathcal{Q}(x_1^n) \log \mathcal{P}(x_1^n)$$

and

$$\mathcal{D}\left(\mathcal{Q}^{(i)}||\mathcal{P}^{(i)}\right) = \sum_{x^{(i)} \in \mathcal{X}^{n-1}} \left[\mathcal{Q}^{(i)}\left(x^{(i)}\right) \log \mathcal{Q}^{(i)}\left(x^{(i)}\right) - \mathcal{Q}^{(i)}\left(x^{(i)}\right) \log \mathcal{P}^{(i)}\left(x^{(i)}\right)\right]$$

it is sufficient to show that

$$\sum_{x_1^n \in \mathcal{X}^n} \mathcal{Q}(x_1^n) \log \mathcal{P}(x_1^n) = \frac{1}{n-1} \sum_{i=1}^{n} \sum_{x^{(i)} \in \mathcal{X}^{n-1}} \mathcal{Q}^{(i)}\left(x^{(i)}\right) \log \mathcal{P}^{(i)}\left(x^{(i)}\right)$$

This is easily seen from the product property of $\mathcal{P}$ - we have

$$\mathcal{P}(x_1^n) = \mathcal{P}^{(i)}\left(x^{(i)}\right)\mathcal{P}_i(x_i)$$

for all i, as well as

$$\mathcal{P}(x_1^n) = \prod_{i=1}^{n} \mathcal{P}_i(x_i)$$

therefore

$$\sum_{x_1^n \in \mathcal{X}^n} \mathcal{Q}(x_1^n) \log \mathcal{P}(x_1^n) = \frac{1}{n} \sum_{i=1}^{n} \sum_{x_1^n \in \mathcal{X}^n} \mathcal{Q}(x_1^n) \left[\log \mathcal{P}^{(i)}\left(x^{(i)}\right) + \log \mathcal{P}_i(x_i) \right]$$

$$= \frac{1}{n} \sum_{i=1}^{n} \sum_{x_1^n \in \mathcal{X}^n} \mathcal{Q}(x_1^n) \log \mathcal{P}^{(i)}\left(x^{(i)}\right)$$

$$+ \frac{1}{n} \sum_{x_1^n \in \mathcal{X}^n} \mathcal{Q}(x_1^n) \log \mathcal{P}(x_1^n)$$

iii. Re-arrangement => Re-arranging the above, we obtain

$$\sum_{x_1^n \in \mathcal{X}^n} \mathcal{Q}(x_1^n) \log \mathcal{P}(x_1^n) = \frac{1}{n-1} \sum_{i=1}^{n} \sum_{x_1^n \in \mathcal{X}^n} \mathcal{Q}(x_1^n) \log \mathcal{P}^{(i)}\left(x^{(i)}\right)$$

$$= \frac{1}{n-1} \sum_{i=1}^{n} \sum_{x^{(i)} \in \mathcal{X}^{n-1}} \mathcal{Q}^{(i)}\left(x^{(i)}\right) \log \mathcal{P}^{(i)}\left(x^{(i)}\right)$$

where we have used the defining property of $\mathcal{Q}^{(i)}$.

Tensorization of the Entropy

1. Introduction: We now use the results above to prove our main exponential concentration inequality. As before, we let $X_1, \ldots, X_n$ be independent random variables, and investigate the concentration properties of

$$Z = g(X_1, \ldots, X_n)$$

2. Re-casting the Martingale Differences Inequality: We extend the martingale differences inequality in a powerful way. Noting that it may be re-cast as

$$Var[Z] \leq \sum_{i=1}^{n} \mathbb{E}[\mathbb{E}_i[Z^2] - (\mathbb{E}_i[Z])^2]$$

we generalize the martingale differences inequality to a large class of convex functions ϕ (by exploiting Jensen's inequality for convex functions) as

$$\mathbb{E}[\phi(Z)] - \phi(\mathbb{E}[Z]) \leq \sum_{i=1}^{n} \mathbb{E}[\mathbb{E}_i[\phi(Z)] - \phi(\mathbb{E}_i[Z])]$$

Using

$$\phi(x) = x^2$$

recovers the original martingale differences inequality.

3. Applicability of the Extended Martingale Differences Inequality: It turns out that this inequality remains valid for a large class of convex functions (Beckner (1989), Ledoux (1997), Latala and Oleszkiewicz (2000), Chafai (2002)).
4. Usage for the Relative Entropy Function: The function of interest in our case is

$$\phi(x) = x \log x$$

- the relative entropy function. For this function, as can be seen below, the LHS of the inequality may be written as the relative entropy induced by Z on $\mathcal{X}^n$ and the distribution of X_1^n. Hence the name "Tensorization of the entropy inequality" (Ledoux (1997)).

5. Tensorization of Entropy Theorem - Statement: Let

$$\phi(x) = x \log x$$

for

$$x > 0$$

Let $X_1, \ldots, X_n$ be independent random variables taking values in $\mathcal{X}$ and let f be a positive-valued function on $\mathcal{X}^n$. Letting

$$Y = f(X_1, \ldots, X_n)$$

we have

$$\mathbb{E}[\phi(Y)] - \phi(\mathbb{E}[Y]) \leq \sum_{i=1}^{n} \mathbb{E}[\mathbb{E}_i[\phi(Y)] - \phi(\mathbb{E}_i[Y])]$$

a. Proof Step #1 – Outline => We only prove the statement for discrete random variables $X_1, \ldots, X_n$. The extension to the general case is technical but straightforward. The inequality is a direct consequence of Han's inequality for relative entropies. First, note that if the inequality is true for a random variable Y, then it is also true for cY where c is a positive constant. Hence, without loss of generality, we assume that

$$\mathbb{E}[Y] = 1$$

b. Proof Step #2 – The Tensorial Measure => We set the probability measure $\mathcal{Q}$ on $\mathcal{X}^n$ to be

$$\mathcal{Q}(x_1^n) = f(x_1^n)\, \mathcal{P}(x_1^n)$$

where $\mathcal{P}$ denotes the distribution of

$$X_1^n = [X_1, \ldots, X_n]$$

Then clearly

$$\mathbb{E}[\phi(Y)] - \phi(\mathbb{E}[Y]) = \mathbb{E}[\phi(Y)] - 0 = \mathbb{E}[Y \log Y] = \mathcal{D}(\mathcal{Q}||\mathcal{P})$$

Using Han's inequality for relative entropy, $\mathcal{D}(\mathcal{Q}||\mathcal{P})$ cannot exceed $\sum_{i=1}^{n}\left[\mathcal{D}(\mathcal{Q}||\mathcal{P}) - \mathcal{D}\left(\mathcal{Q}^{(i)}||\mathcal{P}^{(i)}\right)\right]$. However, straightforward calculation shows that

$$\sum_{i=1}^{n}\left[\mathcal{D}(\mathcal{Q}||\mathcal{P}) - \mathcal{D}\left(\mathcal{Q}^{(i)}||\mathcal{P}^{(i)}\right)\right] = \sum_{i=1}^{n}\mathbb{E}[\mathbb{E}_i[\phi(Y)] - \phi(\mathbb{E}_i[Y])]$$

and the original statement of inequality follows.

6. Ledoux's Entropy Method: The main idea behind Ledoux's entropy method for proving concentration bounds is the application of the tensorization inequality to the positive random variable

$$Y = e^{sZ}$$

Denoting the moment generating function of Z by

$$F(s) = \mathbb{E}[e^{sZ}]$$

the LHS of the tensorization of entropy inequality becomes

$$s\mathbb{E}[Ze^{sZ}] - \mathbb{E}[e^{sZ}]\log\mathbb{E}[e^{sZ}] = s\frac{dF(s)}{ds} - F(s)\log F(s)$$

   a. Approach Behind the Ledoux's Entropy Method => The strategy then is to derive upper bounds for $\frac{dF(s)}{ds}$, and then derive the tail bounds using Chernoff bounding. To do this in a convenient way, however, requires additional bounds for the RHS of the tensorization of the entropy inequality.

Logarithmic Sobolev Inequalities

1. Introduction: As before, we set

$$Z_i = g_i(X_1, \dots, X_{i-1}, X_{i+1}, \dots, X_n)$$

where g_i is some function over $\mathcal{X}^n$. Here we further develop the RHS of the tensorization of the entropy inequality to obtain further inequalities that serve as basis for deriving the exponential concentration inequalities. These inequalities are closely related to the so-called *Logarithmic Sobolev Inequalities of Analysis* (Ledoux (1997, 1999), Massart (2000a), Ledoux (2001)).

2. Relative Entropy Inequality Lemma: Let Y be a positive random variable. Then, for any

$$u > 0$$

$$\mathbb{E}[Y \log Y] - (\mathbb{E}[Y]) \log \mathbb{E}[Y] \leq \mathbb{E}[Y \log Y - Y \log u - (Y - u)]$$

   a. Proof => Since for any

$$x > 0$$

$$\log x \leq x - 1$$

   we have

$$\log \frac{u}{\mathbb{E}[Y]} \leq \frac{u}{\mathbb{E}[Y]} - 1$$

   hence

$$(\mathbb{E}[Y]) \log \frac{u}{\mathbb{E}[Y]} \leq u - \mathbb{E}[Y]$$

upon re-arranging which, you retrieve the original statement.

3. Logarithmic Sobolev Inequality: Denote

$$\psi(x) = e^x - x - 1$$

Then

$$s\mathbb{E}[Ze^{sZ}] - \mathbb{E}[e^{sZ}]\log\mathbb{E}[e^{sZ}] \leq \sum_{i=1}^{n}\mathbb{E}[\psi(-s\{Z - Z_i\})e^{sZ}]$$

a. Proof => We bound each term on the RHS of the tensorial entropy inequality. Note that the relative entropy inequality implies that if Y_i is a positive function of $X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_n$, then

$$\mathbb{E}_i[Y\log Y] - (\mathbb{E}_i[Y])\log\mathbb{E}_i[Y] \leq \mathbb{E}_i[Y\log Y - Y\log Y_i] - \mathbb{E}_i[Y - Y_i]$$

Applying the above inequality to the variables

$$Y = e^{sZ}$$

and

$$Y_i = e^{sZ_i}$$

one gets

$$\mathbb{E}_i[Y\log Y] - (\mathbb{E}_i[Y])\log\mathbb{E}_i[Y] \leq \sum_{i=1}^{n}\mathbb{E}[\psi(-s\{Z - Z_i\})e^{sZ}]$$

4. Symmetrized Logarithmic Sobolev Inequality (Massart (2000a)): If $\psi(x)$ is defined as in the logarithmic Sobolev inequality, then

$$s\mathbb{E}[Ze^{sZ}] - \mathbb{E}[e^{sZ}]\log\mathbb{E}[e^{sZ}] \leq \sum_{i=1}^{n}\mathbb{E}[\psi(-s\{Z - Z'_i\})e^{sZ}]$$

Moreover, denote

$$\tau(x) = x(e^x - 1)$$

Then, for all

$$s \in \mathbb{R}$$

$$s\mathbb{E}[Ze^{sZ}] - \mathbb{E}[e^{sZ}]\log\mathbb{E}[e^{sZ}] \leq \sum_{i=1}^{n}\mathbb{E}\left[\tau(-s\{Z - Z'_i\})\mathbb{I}_{Z>Z'_i}e^{sZ}\right]$$

and

$$s\mathbb{E}[Ze^{sZ}] - \mathbb{E}[e^{sZ}]\log\mathbb{E}[e^{sZ}] \leq \sum_{i=1}^{n}\mathbb{E}\left[\tau(s\{Z - Z'_i\})\mathbb{I}_{Z<Z'_i}e^{sZ}\right]$$

a. Proof => The first inequality is proved exactly as in the logarithmic Sobolev inequality, by nothing that, just like Z_i, Z'_i is also independent of X_i. To prove the 2nd and the 3rd inequalities, write

$$\psi(-s\{Z - Z'_i\})e^{sZ} = \psi(-s\{Z - Z'_i\})\mathbb{I}_{Z>Z'_i}e^{sZ} + \psi(+s\{Z'_i - Z\})\mathbb{I}_{Z<Z'_i}e^{sZ}$$

By symmetry, the conditional expectation of the 2nd term may be written as

$$\begin{aligned}\mathbb{E}\left[\psi(+s\{Z'_i - Z\})\mathbb{I}_{Z<Z'_i}e^{sZ}\right] &= \mathbb{E}\left[e^{sZ'_i}\psi(s\{Z - Z'_i\})\mathbb{I}_{Z>Z'_i}\right] \\ &= \mathbb{E}\left[e^{sZ}e^{-s(Z-Z'_i)}\psi(s\{Z - Z'_i\})\mathbb{I}_{Z>Z'_i}\right]\end{aligned}$$

Summarizing, we have

$$\mathbb{E}_i[\psi(-s\{Z-Z'_i\})e^{sZ}] = \mathbb{E}_i\left[\psi(-s\{Z-Z'_i\}) + e^{-s(Z-Z'_i)}\psi(s\{Z-Z'_i\})e^{sZ}\mathbb{I}_{Z>Z'_i}\right]$$

The 2nd inequality of the theorem follows simply by noting that

$$\psi(x) = e^x\psi(-x) = x(e^x - 1) = \tau(x)$$

The last inequality follows similarly.

Logarithmic Sobolev Inequalities - Applications

1. Introduction: Here we show how the logarithmic Sobolev inequalities developed in the previous section maybe used to obtain powerful exponential concentration inequalities. The first result is fairly easy to obtain, yet it turns out to be very useful. Also, its proof is proto-typical in the sense that, it shows in a transparent way, the main ideas.
2. Difference Square Bound: Assume that there exists a positive constant C such that, almost surely

$$\sum_{i=1}^{n}(Z-Z'_i)^2\mathbb{I}_{Z>Z'_i} \leq \mathcal{C}$$

Then, for all

$$t > 0$$

$$\mathbb{P}[Z - \mathbb{E}[Z] \geq t] \leq e^{-\frac{t^2}{4\mathcal{C}}}$$

a. Proof Step #1 – Application of the Symmetrized Logarithmic Sobolev Inequality => Observe that for

$$x > 0$$

$$\tau(-x) \leq x^2$$

and therefore, for any

$$s > 0$$

the symmetrized logarithmic inequality implies

$$s\mathbb{E}[Ze^{sZ}] - \mathbb{E}[e^{sZ}]\log\mathbb{E}[e^{sZ}] \leq \mathbb{E}\left[e^{sZ}\sum_{i=1}^{n}(Z - Z_i')^2\mathbb{I}_{Z<Z'_i}\right] \leq s^2\mathcal{C}\mathbb{E}[e^{sZ}]$$

where we have used the assumptions of the theorem.

b. Proof Step #2 – Bounding of the Expectation => Denoting the moment generating function of Z by

$$F(s) = \mathbb{E}[e^{sZ}]$$

the above inequality may be re-written as

$$s\frac{dF(s)}{ds} - F(s)\log F(s) \leq s^2\mathcal{C}F(s)$$

After diving both sides by $s^2F(s)$, we observe that the LHS is just a derivative of

$$H(s) = \frac{\log F(s)}{s}$$

that is, we obtain the inequality

$$\frac{dH(s)}{ds} \leq \mathcal{C}$$

By l'Hospital's rule, we note that

$$\underset{s \to 0}{Lim}\, H(s) = \frac{F'(0)}{F(0)} = \mathbb{E}[Z]$$

so by integrating the above inequality, we get

$$H(s) \leq \mathbb{E}[Z] + s\mathcal{C}$$

or, in other words

$$F(s) \leq e^{s\mathbb{E}[Z]+s^2\mathcal{C}}$$

c. Proof Step #3 – Apply Markov's Inequality and Chernoff Bounds => By Markov's inequality

$$\mathbb{P}[Z - \mathbb{E}[Z] \geq t] \leq F(s)e^{-s\mathbb{E}[Z]-st} \leq e^{-s^2\mathcal{C}-st}$$

The choice of

$$s = \frac{t}{2\mathcal{C}}$$

makes the upper bound $e^{-\frac{t^2}{4\mathcal{C}}}$. Replace Z by $-Z$ to obtain the same upper bound for $\mathbb{P}[Z \leq \mathbb{E}[Z] - t]$.

3. Two-sided Bounded Differences Inequality: It is clear from the proof above that under the condition

$$\sum_{i=1}^{n}(Z - Z_i')^2 \mathbb{I}_{Z>Z_i'} \leq \mathcal{C}$$

one has a two-sided inequality

$$\mathbb{P}[|Z - \mathbb{E}[Z]| \geq t] \leq 2e^{-\frac{t^2}{4\mathcal{C}}}$$

An immediate corollary of this is that functions following the bounded-square differences inequality also satisfy sub-Gaussian tail probability.

4. McDiarmid Bounded Differences Inequality: Under the conditions of bounded square differences, McDiarmid (1989) showed that

$$\mathbb{P}[|Z - \mathbb{E}[Z]| \geq t] \leq 2e^{-\frac{2t^2}{\mathcal{C}}}$$

This may be seen by writing Z as a sum of Martingale differences as in the Martingale Differences Theorem, using Chernoff bounding, and proceeding as in the proof for Hoeffding's inequality, since that argument also works for sums of Martingale differences.

5. Bounded Square Differences Inequality - Statement: Assume that the function g satisfies the bounded square differences assumption with constants $\mathcal{C}_1, \ldots, \mathcal{C}_n$, then

$$\mathbb{P}[|Z - \mathbb{E}[Z]| \geq t] \leq 2e^{-\frac{t^2}{4\mathcal{C}}}$$

where

$$\mathcal{C} = \sum_{i=1}^{n} \mathcal{C}_i^{\,2}$$

6. Usage of the Bounded Square Differences Inequality: We remark here that the constant $\mathcal{C}$ appearing in the exponent above may be improved (e.g., the Martingale Differences Technique used in McDiarmid (1989) shows an approach). Thus, we have been able to extend the bounded differences (in addition to the bounded square differences condition) to an exponential concentration inequality. Note that by combining the variance bound of the bounded differences inequality with Chebyshev's inequality, we only obtained

$$\mathbb{P}[|Z - \mathbb{E}[Z]| \geq t] \leq \frac{\mathcal{C}}{2t^2}$$

therefore, the exponential improvement obtained here is substantial. All the sample applications that use the bounded differences inequality are now improved in an essential way without any further work.

7. Relaxing the Bounded Differences Square Bound: The differences square bound maybe relaxed in the following way to produce expected data-dependent tail bounds. If

$$\mathbb{E}\left[\sum_{i=1}^{n}(Z - Z_i')^2 \mathbb{I}_{Z>Z_i'} | X_1^n\right] \leq \mathcal{C}$$

then for all

$$t > 0$$

$$\mathbb{P}[Z \geq \mathbb{E}[Z] + t] \leq e^{-\frac{t^2}{4\mathcal{C}}}$$

(This can be seen by using the Association Inequality for the monotonic multi-variate functions $(Z - Z_i')^2$ and e^{sZ}). Further, if

$$\mathbb{E}\left[\sum_{i=1}^{n}(Z - Z_i')^2 \mathbb{I}_{Z<Z_i'} | X_1^n\right] \leq \mathcal{C}$$

we can see that

$$\mathbb{P}[Z \leq \mathbb{E}[Z] - t] \leq e^{-\frac{t^2}{4\mathcal{C}}}$$

8. Example - Hoeffding's Inequality in Hilbert Space: As a simple illustration of the power of bounded differences inequality, we derive a Hoeffding-type inequality for the sums of random variables taking values in a Hilbert space. In particular, we show that if $X_1, \ldots, X_n$ are independent zero-mean random variables taking values in a separable Hilbert space such that

$$\|X_i\| \leq \frac{\mathcal{C}_i}{2}$$

with probability one, then for all

$$t \geq 2\sqrt{\mathcal{C}}$$

$$\mathbb{P}\left[\left\|\sum_{i=1}^{n} X_i\right\| \geq t\right] \leq e^{-\frac{t^2}{2\mathcal{C}}}$$

where

$$\mathcal{C} = \sum_{i=1}^{n} \mathcal{C}_i{}^2$$

   a. Proof => The above follows simply by observing that, by triangle inequality

$$Z = \left\|\sum_{i=1}^{n} X_i\right\|$$

satisfies the bounded differences inequality with constants $\mathcal{C}_i$, and therefore

$$\mathbb{P}\left[\left\|\sum_{i=1}^{n} X_i\right\| \geq t\right] = \mathbb{P}\left[\left\|\sum_{i=1}^{n} X_i\right\| - \mathbb{E}\left[\left\|\sum_{i=1}^{n} X_i\right\|\right] \geq t - \mathbb{E}\left[\left\|\sum_{i=1}^{n} X_i\right\|\right]\right]$$
$$\leq e^{-\frac{2\left(t-\mathbb{E}\left[\left\|\sum_{i=1}^{n} X_i\right\|\right]\right)^2}{\mathcal{C}}}$$

The proof is completed by observing that, by independence

$$\left[\left\|\sum_{i=1}^{n} X_i\right\|\right] \leq \sqrt{\left\{\mathbb{E}\left[\left\|\sum_{i=1}^{n} X_i\right\|\right]\right\}^2} = \sqrt{\sum_{i=1}^{n} \mathbb{E}\left[\left\|{X_i}^2\right\|\right]} \leq \sqrt{\mathcal{C}}$$

Application of the Markov/Chebyshev inequality produces the final result.

9. Bounded Square Differences vs. Bounded Differences: Note that the bounded square differences criterion is much stronger than the bounded differences criterion. This is because the former does not require that g be bounded – all that is required is that

$$\sup_{\substack{x_1, \ldots, x_n \\ x'_1, \ldots, x'_n}} \sum_{i=1}^{n} |g(x_1, \ldots, x_n) - g(x_1, \ldots, x_{i-1}, x'_i, x_{i+1}, \ldots, x_n)|^2 \leq \sum_{i=1}^{n} {\mathcal{C}_i}^2$$

an obviously milder requirement. Thus, as seen in the next example, in situations where the bounded differences technique may not work, the bounded square differences inequality may provide a sharp bound.

10. Example - Largest Eigen-value of a Random Symmetric Matrix: Using the bounded square differences inequality, we derive a result of Alon, Krivelevich, and Vu (2002). Let A be a symmetric real matrix whose entries

$$X_{i,j} \; \forall \; 1 \leq i \leq j \leq n$$

are independent random variables with absolute value bounded by 1. We seek to show that if

$$Z = \lambda_1$$

is the largest eigenvalue of A, then

$$\mathbb{P}[Z \geq \mathbb{E}[Z] + t] \leq e^{-\frac{t^2}{16}}$$

The property of the largest eigenvalue we need is that if

$$v = (v_1, \ldots, v_n) \in \mathbb{R}^n$$

is an eigenvector corresponding to the largest eigenvalue λ_1 with

$$\|v\| = 1$$

then

$$\lambda_1 = v^T A v = \sup_{u:\, \|u\| = 1} u^T A u$$

a. Using the Bounded Square Differences Theorem => To use the bounded square differences inequality, consider the symmetric matrix $A'_{i,j}$ obtained by replacing $X_{i,j}$ in A by the independent copy $X'_{i,j}$ while keeping all other variables fixed. Let $Z'_{i,j}$ denote the largest eigenvalue of the obtained matrix. Then, using the above-mentioned property of the largest eigenvalue

$$\begin{aligned}(Z - Z'_{i,j})\mathbb{I}_{Z>Z'_{i,j}} \leq (v^T A v - v^T A'_{i,j} v)\mathbb{I}_{Z>Z'_{i,j}} = v^T (A - A'_{i,j}) v \mathbb{I}_{Z>Z'_{i,j}} \\ = [v_i v_j (X_{i,j} - X'_{i,j})] \leq 2|v_i v_j|\end{aligned}$$

Therefore

$$\sum_{1\le i\le j\le n}\left(Z-Z'_{i,j}\right)^2\mathbb{I}_{Z>Z'_{i,j}}\le\sum_{1\le i\le j\le n}4\left|v_iv_j\right|^2\le\left[4\sum_{i=1}^{n}{v_i}^2\right]=4$$

The result now follows from the bounded differences inequality.

b. Comparison with Efron-Stein => Note that by the Efron-Stein inequality, we also have

$$Var[Z]\ge 4$$

A similar exponential inequality, though with a somewhat worse constant in the exponent, can also be derived for the lower tail. In particular using a theorem from below, it can be shown that for

$$t>0$$

$$\mathbb{P}[Z\le\mathbb{E}[Z]-t]\le e^{-\frac{t^2}{16(e-1)}}$$

c. Alternate Eigenvalues => Notice that the same proof works for the smallest eigenvalue as well. Alon, Krivelevich, and Vu (2002) show that, with a simple extension of the argument, if Z is the k^{th} largest eigenvalue (or the k^{th} smallest eigenvalue), then the upper bound becomes $e^{-\frac{t^2}{16k^2}}$, although it is not clear whether the factor $\frac{1}{k^2}$ in the exponent is necessary.

Exponential Inequalities for Self-Bounding Functions

1. Introduction: Recall that the bounded differences inequality implies that for self-bounding functions

$$Var[Z] \leq \mathbb{E}[Z]$$

Based on the logarithmic Sobolev inequality, we may now obtain exponential concentration inequality bounds (Boucheron, Lugosi, and Massart (2000), Massart (2000a)).

2. Bennett and Logarithmic Sobolev Functions: Recall the definition of the following two functions that we have already seen above in the Bennett's inequality and in the logarithmic Sobolev inequalities:

$$h(u) = (1+u)\log(1+u) - u \; \forall \; u \geq -1$$

$$\sup_{u \geq -1} [uv - h(v)] = e^v - v - 1$$

3. Logarithmic Sobolev Inequality for Self-Bounding Functions: Assume that g satisfies the self-bounding property. Then for every

$$s \in \mathbb{R}$$

$$\log \mathbb{E}\left[e^{s(Z-\mathbb{E}[Z])}\right] \leq \psi(s)\mathbb{E}[Z]$$

Moreover, for every

$$t > 0$$

$$\mathbb{P}[Z \geq \mathbb{E}[Z] + t] \leq e^{-\mathbb{E}[Z]h\left(\frac{t}{\mathbb{E}[Z]}\right)}$$

and for every

$$0 < t < \mathbb{E}[Z]$$

$$\mathbb{P}[Z \leq \mathbb{E}[Z] - t] \leq e^{-\mathbb{E}[Z]h(-\frac{t}{\mathbb{E}[Z]})}$$

a. Corollary from the re-cast => By recalling that

$$h(u) = \frac{u^2}{2 + \frac{2u}{3}}$$

for

$$u \geq 0$$

(we have already used this in the proof for Bernstein's inequality), and observing that

$$h(u) \geq \frac{u^2}{2}$$

for

$$u \leq 0$$

we obtain the following two corollaries:

i. For every

$$t < 0$$

$$\mathbb{P}[Z \geq \mathbb{E}[Z] + t] \leq e^{-\frac{t^2}{2\mathbb{E}[Z] + \frac{2t}{3}}}$$

ii. For every

$$0 < t < \mathbb{E}[Z]$$

$$\mathbb{P}[Z \leq \mathbb{E}[Z] - t] \leq e^{-\frac{t^2}{2\mathbb{E}[Z]}}$$

4. Proof of Logarithmic Sobolev Inequality for Self-Bounding Functions:
    a. Step #1 – Applying the Logarithmic Sobolev Inequality => Since the function $\psi(s)$ is convex with

$$\psi(0) = 0$$

for any s and any

$$u \in [0, 1]$$

$$\psi(-su) \leq u\psi(-s)$$

Thus, since

$$(Z - Z_i) \in [0, 1]$$

we have that, for every s

$$\psi\big(-s(Z - Z_i)\big) \leq (Z - Z_i)\psi(-s)$$

Therefore, the combination of the logarithmic inequality and the condition

$$\sum_{i=1}^{n}(Z - Z_i) \leq Z$$

implies that

$$s\mathbb{E}[Ze^{sZ}] - \mathbb{E}[e^{sZ}]\log\mathbb{E}[e^{sZ}] \leq \mathbb{E}\left[\psi(-s)e^{sZ}\sum_{i=1}^{n}(Z - Z_i)\right] \leq \psi(-s)\mathbb{E}[Ze^{sZ}]$$

b. Step #2 – Variable Transformation => Introduce

$$\tilde{Z} = Z - \mathbb{E}[Z]$$

and define for any s

$$\tilde{F}(s) = \mathbb{E}\left[e^{s\tilde{Z}}\right]$$

The inequality above then becomes

$$[s - \psi(-s)]\frac{\tilde{F}'(s)}{\tilde{F}(s)} - \log\tilde{F}(s) \leq \psi(-s)\mathbb{E}[Z]$$

which, on writing

$$G(s) = \log\tilde{F}(s)$$

implies

$$[1 - e^{-s}]\frac{dG(s)}{ds} - G(s) \leq \psi(-s)\mathbb{E}[Z]$$

c. Step #3 – Inequality for $G(s)$ => Observe that the function

$$G_0(s) = \psi(s)\mathbb{E}[Z]$$

is a solution to the ODE

$$[1 - e^{-s}]\frac{dG(s)}{ds} - G(s) \leq \psi(-s)\mathbb{E}[Z]$$

Here we intend to show that

$$G(s) \leq G_0(s)$$

In fact, if

$$G_1(s) = G(s) - G_0(s)$$

then

$$[1 - e^{-s}]\frac{dG_1(s)}{ds} - G_1(s) \leq 0$$

Thus, defining

$$\tilde{G}(s) = \frac{G_1(s)}{e^s - 1}$$

we have

$$[1 - e^{-s}][e^s - 1]\frac{d\tilde{G}(s)}{ds} \leq 0$$

This clearly indicates that $\frac{d\tilde{G}(s)}{ds}$ is non-positive, and therefore $\tilde{G}(s)$ is non-increasing.

d. Step #4 – Proof that $G(s) \leq G_0(s)$ => Now, since $\tilde{Z}$ is centered

$$\left.\frac{dG_1(s)}{ds}\right|_{s=0} = 0$$

Using the fact that

$$\frac{s}{e^s - 1} \to 1$$

as

$$s \to 0$$

we conclude that

$$\frac{d\tilde{G}(s)}{ds} \to 0$$

as

$$s \to 0$$

This shows that $\tilde{G}(s)$ is non-positive over $(0, \infty)$ and non-negative over $(-\infty, 0)$, hence $G_1(s)$ is non-positive everywhere, therefore

$$G(s) \leq G_0(s)$$

This proves the first inequality.

e. Step #5 – Tail Probabilities Calculation => The proof of inequalities for the tail probabilities maybe completed by Chernoff Bounding:

$$\mathbb{P}[Z \geq \mathbb{E}[Z] + t] \leq e^{-\sup_{s>0}[ts - \psi(s)\mathbb{E}[Z]]}$$

$$\mathbb{P}[Z \leq \mathbb{E}[Z] - t] \leq e^{-\sup_{s<0}[-ts - \psi(s)\mathbb{E}[Z]]}$$

f. Step #6 – Reduced Form Bounds => The final step of the proof is done using the following easy-to-check and well-known relations:

$$\sup_{s>0}\left[ts-\psi(s)\mathbb{E}[Z]\right]=\mathbb{E}[Z]h\left(\frac{t}{\mathbb{E}[Z]}\right)$$

for

$$t>0$$

$$\sup_{s<0}\left[-ts-\psi(s)\mathbb{E}[Z]\right]=\mathbb{E}[Z]h\left(-\frac{t}{\mathbb{E}[Z]}\right)$$

for

$$0<t<\mathbb{E}[Z]$$

Combinatorial Entropy

1. VC Entropy: The VC Entropy is closely related to the VC dimension discussed earlier. Let $\mathcal{A}$ be an arbitrary collection of subsets of $\mathcal{X}$, and let

$$x_1^n=(x_1,\ldots,x_n)$$

be a vector of n points of $\mathcal{X}$. Recall that the *shatter coefficient* is defined as the size of the trace of $\mathcal{A}$ on x_1^n, that is

$$\mathcal{T}(x_1^n)=|Trace(x_1^n)|=|\{A\cap[x_1,\ldots,x_n]\colon A\in\mathcal{A}\}|$$

The VC entropy is defined as the log of the shatter coefficient, that is

$$h(x_1^n) = log_2 \mathcal{T}(x_1^n)$$

2. VC Entropy Self-Bounding Property: The VC Entropy has the self-bounding property.
   a. Proof Step #1 – Setup => We need to show that there exists a function h' of $n-1$ variables such that for all

$$i = 1, \ldots, n$$

   writing

$$x^{(i)} = (x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$$

   the following true self-bounding conditions are valid:

$$0 \leq h(x_1^n) - h'\left(x^{(i)}\right) \leq 1$$

   and

$$\sum_{i=1}^{n} \left[h(x_1^n) - h'\left(x^{(i)}\right)\right] \leq h(x_1^n)$$

   b. Proof Step #2 – Outline => We define h' in the natural way, that is, as the entropy based on the $n-1$ points in its arguments. Then, clearly for any

$$h(x_1^n) \geq h'\left(x^{(i)}\right)$$

   and the difference cannot be more than one. The non-trivial part of the proof is to demonstrate the validity of the second condition. We do this using Han's inequality for joint entropy.

c. Proof Step #3 – Invoking Uniform Outcome Distribution => Consider the uniform distribution over the outcome set $Trace(x_1^n)$. This defines a random vector

$$Y = (Y_1, \dots, Y_n) \in \mathcal{Y}^n$$

Then clearly

$$h(x_1^n) = log_2|Trace(x_1^n)| = \frac{1}{\log 2}\mathcal{H}(Y_1, \dots, Y_n)$$

where $\mathcal{H}(Y_1, \dots, Y_n)$ is the joint entropy of $Y_1, \dots, Y_n$. Since the uniform distribution maximizes the entropy (in other words, the uniform distribution in the outcome space – the uniform trace – maximizes the outcome entropy), we also have for all

$$i \leq n$$

$$h'\left(x^{(i)}\right) \geq \frac{1}{\log 2}\mathcal{H}(Y_1, \dots, Y_{i-1}, Y_{i+1}, \dots, Y_n)$$

d. Proof Step #4 – Invoking Han's Inequality => Since by Han's inequality for joint entropy

$$\mathcal{H}(Y_1, \dots, Y_n) \leq \frac{1}{n-1}\sum_{i=1}^{n}\mathcal{H}(Y_1, \dots, Y_{i-1}, Y_{i+1}, \dots, Y_n)$$

we have

$$\sum_{i=1}^{n}\left[h(x_1^n) - h'\left(x^{(i)}\right)\right] \leq h(x_1^n)$$

as desired.

3. VC Entropy Probabilistic Bounds: Let $X_1, \ldots, X_n$ be a set of independent random variables taking values in $\mathcal{X}$ and let

$$Z = h(x_1^n)$$

denote the random VC entropy. Then the following are true:

$$Var[Z] \leq \mathbb{E}[Z]$$

For every

$$t < 0$$

$$\mathbb{P}[Z \geq \mathbb{E}[Z] + t] \leq e^{-\frac{t^2}{2\mathbb{E}[Z]+\frac{2t}{3}}}$$

For every

$$0 < t < \mathbb{E}[Z]$$

$$\mathbb{P}[Z \leq \mathbb{E}[Z] - t] \leq e^{-\frac{t^2}{2\mathbb{E}[Z]}}$$

a. Moreover, for the random shatter coefficient $\mathcal{T}(x_1^n)$, we have

$$\mathbb{E}[log_2\mathcal{T}(x_1^n)] \leq log_2\mathbb{E}[\mathcal{T}(x_1^n)] \leq log_2 e\mathbb{E}[log_2\mathcal{T}(x_1^n)]$$

4. VC Entropy Probabilistic Bounds - Proof: The LHS of the VC Entropy Bounds statement follows from the Jensen's inequality, while the RHS arises by taking

$$s = \log 2$$

in the first inequality of the exponential inequality for self-bounding functions. This last statement shows that the expected VC entropy $\mathbb{E}[log_2\mathcal{T}(x_1^n)]$ and the annealed VC Entropy $log_2\mathbb{E}[\mathcal{T}(x_1^n)]$ are tightly connected regardless of the class of sets $\mathcal{A}$ and the distribution of X_i's.

5. Non-trivial ERM Consistency (Vapnik (1995)): The probabilistic VC entropy bounds above answers in a positive way an open question raised by Vapnik (1995): the EMR procedure is *non-trivially consistent* and *rapidly convergent* if and only if the annealed entropy rate $\frac{1}{n}log_2\mathbb{E}[\mathcal{T}(x_1^n)]$ converges to zero.
6. Bounded Supremum of the Self Bounding Exponential Inequality: Let C and a denote two positive real numbers and denote

$$h_1(x) = x + 1 - \sqrt{2x+1}$$

It is easy to show that the local supremum is given from

$$\sup_{\lambda \in \left[0, \frac{1}{a}\right]} \left(\lambda t - \frac{C\lambda^2}{1-a\lambda}\right) = \frac{2C}{a^2} h\left(\frac{at}{2C}\right) \geq \frac{t^2}{2(2C+at)}$$

and that the supremum is attained at

$$\lambda = \frac{1}{a}\left[1 - \frac{1}{\sqrt{1+\frac{at}{C}}}\right]$$

7. Unbounded Supremum of the Self-Bounding Exponential Inequality: Further

$$\sup_{\lambda \in [0, \quad \infty)} \left(\lambda t - \frac{C\lambda^2}{1+a\lambda}\right) = \frac{2C}{a^2} h\left(-\frac{at}{2C}\right) \geq \frac{t^2}{4C}$$

if

$$t < \frac{C}{a}$$

and the supremum is attained at

$$\lambda = \frac{1}{a}\left[\frac{1}{\sqrt{1 - \frac{at}{C}}} - 1\right]$$

8. Generalization of the VC Entropy: The proof of concentration if the VC entropy may be generalized in a straightforward way to a class of functions called *combinatorial entropies* defined below.
9. Combinatorial Entropy - Setup: Let

$$x_1^n = (x_1, \ldots, x_n)$$

be an n-vector of elements

$$x_i \in \mathcal{X}_i$$

to which we associate a set

$$Trace(x_1^n) \subset \mathcal{Y}^n$$

of n-vectors whose component are elements of a possibly different set $\mathcal{Y}$. We assume that for each

$$x \in \mathcal{X}^n$$

and

$$i \leq n$$

the set

$$Trace\left(x^{(i)}\right) = Trace(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$$

is a projection of $Trace(x_1^n)$ along the i^{th} co-ordinate, that is

$$Trace\left(x^{(i)}\right) = \{y^{(i)} = (y_1, \ldots, y_{i-1}, y_{i+1}, \ldots, y_n) \in \mathcal{Y}^{n-1} : \exists\, y_i \in \mathcal{Y} : (y_1, \ldots, y_n) \in Trace(x_1^n)\}$$

The associated combinatorial entropy is

$$h(x_1^n) = log_b|Trace(x_1^n)|$$

where b is an arbitrary positive number ≥ 2.

10. Self-Bounding Property of Combinatorial Entropy: Just as in the case of VC entropy, the combinatorial entropy may also be shown to possess the self-bounding property, i.e., assuming

$$h(x_1^n) = log_b|Trace(x)|$$

is a combinatorial entropy such that for all

$$x \in \mathcal{X}^n$$

and

$$i \leq n$$

$$h(x_1^n) - h\left(x^{(i)}\right) \leq 1$$

then h has the self-bounding property.

11. Combinatorial Entropy - Exponential Bounds Generalization: Assume that

$$h(x_1^n) = log_b|Trace(x)|$$

is a combinatorial entropy such that for all

$$x \in \mathcal{X}^n$$

and

$$i \leq n$$

$$h(x_1^n) - h\left(x^{(i)}\right) \leq 1$$

If

$$X_1^n = (X_1, \dots, X_n)$$

is a vector of n-independent random variables taking values in $\mathcal{X}$, then the random combinatorial entropy

$$Z = h(x_1^n)$$

satisfies

$$\mathbb{P}[Z \geq \mathbb{E}[Z] + t] \leq e^{-\frac{t^2}{2\mathbb{E}[Z]+\frac{2t}{3}}}$$

for

$$t < 0$$

and

$$\mathbb{P}[Z \leq \mathbb{E}[Z] - t] \leq e^{-\frac{t^2}{2\mathbb{E}[Z]}}$$

for

$$0 < t < \mathbb{E}[Z]$$

Moreover

$$\mathbb{E}[log_b|Trace(X_1^n)|] \leq log_b\mathbb{E}[|Trace(X_1^n)|] \leq \frac{b-1}{\log b}\mathbb{E}[log_b|Trace(X_1^n)|]$$

12. Combinatorial Entropy Example - Increasing Sub-sequences: Recall the setup of the example of the increasing sub-sequences earlier, and let $N(x_1^n)$ denote the number of different increasing sub-sequences of x_1^n. Observe that $log_2 N(x_1^n)$ is a combinatorial entropy. This is easy to see by considering

$$\mathcal{Y} = \{0, 1\}$$

and by assigning, to each increasing sub-sequence

$$i_1 < i_2 < \cdots < i_m$$

of x_1^n a binary n-vector

$$y_1^n = (y_1, \ldots, y_n)$$

such that

$$y_j = 1$$

if and only if

$$j = i_k$$

for some

$$k = 1, \ldots, m$$

(i.e., the indices appearing in the increasing sub-sequence are marked by 1).

a. Exponential Bounds => Now that the conditions for combinatorial entropy are met

$$Z = log_2 N(x_1^n)$$

satisfies all the inequalities associated with the combinatorial entropy exponential inequality. This result improves a concentration inequality obtained by Frieze (1991) for $log_2 N(x_1^n)$.

Variations on the Theme of Self-Bounding Functions

1. Introduction: Here we show how the techniques for the entropy method for proving concentration inequalities may be used in various situations not considered so far. The versions differ in the assumptions on how $\sum_{i=1}^{n}(Z - Z'_i)^2$ is controlled by different functions of Z. For various other versions with applications, we refer to Boucheron, Lugosi, and Massart (2003).

2. Generalized Self-Bounding Functions - Probabilistic Bounds: In all cases the upper bound is roughly of the form $e^{-\frac{t^2}{\sigma^2}}$ where σ^2 is the corresponding Efron-Stein upper bound of $Var[Z]$. The first inequality may now be regarded as a generalization of the exponential bounds for self-bounding functions.
3. Exponential Bounds for Generalized Self-Bounding Functions: Assume that there exist positive constants a and b such that

$$\sum_{i=1}^{n}(Z - Z'_i)^2 \mathbb{I}_{Z>Z'_i} \leq aZ + b$$

Then, for

$$s \in \left(0, \frac{1}{a}\right)$$

$$\log \mathbb{E}\left[e^{s(Z-\mathbb{E}[Z])}\right] \leq \frac{s^2}{1-as}(a\mathbb{E}[Z] + b)$$

and for all

$$t > 0$$

$$\mathbb{P}(Z \geq \mathbb{E}[Z] + t) \leq e^{-\frac{t^2}{4a\mathbb{E}[Z]+4b+2at}}$$

a. Reduction to Expectations Bound => Let

$$s > 0$$

Just as in the first step of the proof for the Bounded Square Differences Exponential Bounds, we use the fact that for

$$x > 0$$

$$\tau(-x) \leq x^2$$

and, therefore, by the Symmetrized Logarithmic Inequality we have

$$s\mathbb{E}[Ze^{sZ}] - \mathbb{E}[e^{sZ}]\log\mathbb{E}[e^{sZ}] \leq \mathbb{E}\left[e^{sZ}\sum_{i=1}^{n}(Z - Z'_i)^2\mathbb{I}_{Z>Z'_i}\right]$$
$$\leq s^2(a\mathbb{E}[Ze^{sZ}] + b\mathbb{E}[e^{sZ}])$$

where at the final step we have used the assumptions of our postulate.

b. Inequality ODE for $H(s)$ => Denoting, again

$$F(s) = \mathbb{E}[e^{sZ}]$$

the above inequality becomes

$$s\frac{dF(s)}{ds} - F(s)\log F(s) \leq as^2\frac{dF(s)}{ds}$$

After scaling both sides by $s^2F(s)$, we notice that the LHS is just the derivative of

$$H(s) = \frac{1}{s}\log F(s)$$

so we obtain

$$\frac{dH(s)}{ds} \leq a\frac{d\log F(s)}{ds} + b$$

c. Bounding the Expectation and Extracting Exponential Inequality => Using the fact that

$$\underset{s \to 0}{Limit}\, H(s) = \frac{1}{F(s)} \frac{d \log F(s)}{ds}\Big|_{s=0} = \mathbb{E}[Z]$$

and

$$0 \log F(0) = 0$$

and integrating the inequality, we obtain

$$H(s) \leq \mathbb{E}[Z] + a \log F(s) + bs$$

or, if

$$s < \frac{1}{a}$$

$$\log \mathbb{E}\left[e^{s(Z - \mathbb{E}[Z])}\right] \leq \frac{s^2}{1 - as}(a\mathbb{E}[Z] + b)$$

thereby proving the first inequality. The inequality for the upper tail follows from the Markov inequality along with the bounded/unbounded suprema of the self-bounding exponential inequality.

4. Distinction Between the Upper and the Lower Bounds: Bounds for the lower tail $\mathbb{P}(Z \geq \mathbb{E}[Z] + t)$ maybe more easily derived by exploiting the association inequalities seen before, under much more general conditions on $\sum_{i=1}^{n}(Z - Z'_i)^2 \mathbb{I}_{Z<Z'_i}$. Notice the difference between this and the quantity $\sum_{i=1}^{n}(Z - Z'_i)^2 \mathbb{I}_{Z>Z'_i}$ appearing in the upper bound above.
5. Generalized Self-Bounding Lower Tail: Assume that for some non-decreasing function

$$\sum_{i=1}^{n}(Z - Z'_i)^2 \mathbb{I}_{Z<Z'_i} \leq g(Z)$$

Then for all

$$t > 0$$

$$\mathbb{P}(Z \leq \mathbb{E}[Z] - t) \leq e^{-\frac{t^2}{4\mathbb{E}[g(Z)]}}$$

a. Proof Step #1 - Applying the Symmetrized Logarithmic Sobolev Inequality => To prove the lower tail inequalities, we obtain the upper bounds for

$$F(s) = \mathbb{E}[e^{sZ}]$$

with

$$s < 0$$

By the third inequality of the symmetrized logarithmic Sobolev inequality

$$s\mathbb{E}[Ze^{sZ}] - \mathbb{E}[e^{sZ}] \log \mathbb{E}[e^{sZ}] \leq \sum_{i=1}^{n} \mathbb{E}\left[e^{sZ}\tau\left(s(Z - Z'_i)\right)\mathbb{I}_{Z<Z'_i}\right]$$

By using

$$s < 0$$

and

$$\tau(-x) \leq x^2$$

for

$$x > 0$$

the RHS above becomes

$$\sum_{i=1}^{n} \mathbb{E}\left[e^{sZ} s^2 (Z - Z'_i)^2 \mathbb{I}_{Z<Z'_i}\right] = s^2 \mathbb{E}\left[e^{sZ} \sum_{i=1}^{n} (Z - Z'_i)^2 \mathbb{I}_{Z<Z'_i}\right] \leq s^2 \mathbb{E}[e^{sZ} g(Z)]$$

b. Prof Step #2 – Apply Chebyshev's Association Inequality => Since $g(Z)$ is a non-decreasing function, and e^{sZ} is a decreasing function of Z, Chebyshev's association inequality implies that

$$\mathbb{E}[e^{sZ} g(Z)] \leq \mathbb{E}[e^{sZ}]\mathbb{E}[g(Z)]$$

c. Proof Step #3 – Extraction of the Lower Tail => Scaling both sides of the above inequality by $s^2 F(s)$ and writing

$$H(s) = \frac{1}{s} \log F(s)$$

we obtain

$$\frac{dH(s)}{ds} \leq \mathbb{E}[g(Z)]$$

Integrating in the interval $[s, 0)$, we obtain

$$F(s) \leq e^{s^2 \mathbb{E}[g(Z)] + s\mathbb{E}[ZX]}$$

Finally, applying Markov's inequality and carrying out the optimization results in the statement of the inequality.

6. Data-Dependent Lower Tail: Assume that

$$Z = g(X_1^n) = g(X_1, \dots, X_n)$$

where $X_1, \dots, X_n$ are independent real-valued random variables, and that g is a non-decreasing function of each variable. Suppose there exists another non-decreasing function

$$f: \mathbb{R}^n \to \mathbb{R}$$

such that

$$\sum_{i=1}^{n} (Z - Z'_i)^2 \mathbb{I}_{Z<Z'_i} \leq f(X_1^n)$$

Then, for all

$$t > 0$$

it can be shown, following above, that

$$\mathbb{P}(Z \leq \mathbb{E}[Z] - t) \leq e^{-\frac{t^2}{4\mathbb{E}[f(X_1^n)]}}$$

7. Alternate Approach to the Lower Tail Bound: The next result is useful when one is interested in lower tail-bounds, but $\sum_{i=1}^{n} (Z - Z'_i)^2 \mathbb{I}_{Z<Z'_i}$ is difficult to handle. In certain situations, the right symmetrization pivot, i.e., $\sum_{i=1}^{n} (Z - Z'_i)^2 \mathbb{I}_{Z>Z'_i}$, is easier to bound. In such a situation, however, we need the additional guarantee that $|Z - Z'_i|$ remains bounded. Without loss of generality, we assume that this bound is 1.
8. Lower Tail Bound Using the Right Symmetrization Pivot: Assume that there exists a non-decreasing function g such that

$$\sum_{i=1}^{n}(Z - Z'_i)^2 \mathbb{I}_{Z>Z'_i} \leq g(Z)$$

and for any value of X_1^n and X_1'

$$|Z - Z'_i| \leq 1$$

Then the following are true:

$$\log \mathbb{E}\left[e^{-s(Z-\mathbb{E}[Z])}\right] \leq s^2 \frac{\tau(\kappa)}{\kappa^2} \mathbb{E}[g(Z)]$$

for all

$$\kappa > 0$$

and

$$s \in \left[0, \frac{1}{\kappa}\right]$$

$$\mathbb{P}(Z \leq \mathbb{E}[Z] - t) \leq e^{-\frac{t^2}{4(e-1)\mathbb{E}[g(Z)]}}$$

for all

$$t > 0$$

and

$$t \leq (e-1)\mathbb{E}[g(Z)]$$

9. Proof Step #1 - Using the Right Symmetrization Pivot: The key observation is that the function

$$\frac{\tau(x)}{x^2} = \frac{e^x - 1}{x}$$

is increasing if

$$x > 0$$

Choose

$$\kappa > 0$$

Thus, for

$$s \in \left[-\frac{1}{\kappa}, 0\right]$$

the right symmetrization pivot of the symmetrized logarithmic Sobolev inequality implies that

$$s\mathbb{E}[Ze^{sZ}] - \mathbb{E}[e^{sZ}]\log\mathbb{E}[e^{sZ}] \leq \sum_{i=1}^{n} \mathbb{E}\left[e^{sZ}\tau\left(-s(Z - Z'_i)\right)\mathbb{I}_{Z>Z'_i}\right]$$

where at the last step, we have used the assumption of our postulate.

10. Proof Step #2 - Use of the Association Inequality: Just as in the proof for generalized self-bounding functions lower tail inequality, we bound $\mathbb{E}[e^{sZ}g(Z)]$ by $\mathbb{E}[e^{sZ}]\mathbb{E}[g(Z)]$. For the rest of the proof, we proceed as in the proof for self-bounding functions lower tail inequality. Here we have taken

$$\kappa = 1$$

References

- Ahlswede, R., P. Gacs, and J. Korner (1976): Bounds on Conditional Probabilities with Applications in multi-user Communication *Zeitschrift fur Wahrscheinlichkeitstheorie und verwandte Gebiete* **34** 157-177 (with corrections in **39** 353-354 (1977)).
- Alon, N., Krivelevich, M., and V. H. Vu (2002): On the Concentration of Eigen-values of Random Symmetric Matrices *Israeli Mathematical Journal* **131** 259-267.
- Beckner, W. (1989): A Generalized Poincare's Inequality for Gaussian Measures *Proceedings of the American Mathematical Society* **105** 397-400.
- Bobkov, S., and M. Ledoux (1997): Poincare's Inequalities and Talagrand's Concentration Phenomenon for the Exponential Distribution *Probability Theory and Related Fields* **107** 383-400.
- Boucheron, S., G. Lugosi, and P. Massart (2000): A Sharp Concentration Inequality with Applications *Random Structures and Algorithms* **16** 277-292.
- Boucheron, S., G. Lugosi, and P. Massart (2003): Concentration Inequalities using the Entropy Method *Annals of Probability* **31** 1583-1614.
- Bousquet, O. (2002): A Bennett Concentration Inequality and its Application to Suprema of Empirical Processes *C. R. Acad. Sci. Paris* **334** 495-500.
- Chafai, D. (2002): *On ϕ-entropies and ϕ-Sobolev Inequalities* **arXiv**.
- Cover, T. M., and J. A. Thomas (1991): *Elements of Information Theory* **John Wiley** New York.
- Dembo, A. (1997): Information Inequalities and Concentration of Measure *Annals of Probability* **25** 927-939.
- Frieze, A. M. (1991): On the Length of the Longest Monotone Sub-sequence in a Random Permutation *Annals of Applied Probability* **1** 301-305.
- Han, T. S. (1978): Non-negative Entropy Measures of Multi-variate Symmetric Correlations *Information and Control* **36**.

- Latala, R., and C. Oleszkiewicz (2000): Between Sobolev and Poincare, in: *Geometric Aspects of Functional Analysis, Israeli Seminar (GAFA), 1996-2000* 147-168 **Springer** Lecture Notes in Mathematics **1745**.
- Ledoux, M. (1996): Isoperimetry and Gaussian Analysis *Lectures on Probability Theory and Statistics (editor: P. Bernard)* **Ecole d'Ete de Probabilites de St-Flour XXIV-1994** 165-294.
- Ledoux, M. (1997): On Talagrand's Deviation Inequalities for Product Measures *ESAIM: Probability and Statistics* **1** 63-87.
- Ledoux, M. (1999): Concentration of Measure and Logarithmic Sobolev Inequalities, in: *Seminaire de Probabilites XXXIII, Lecture Notes in Mathematics* **1709** 120-216 **Springer**.
- Ledoux, M. (2001): *The Concentration of Measure Phenomenon* **American Mathematical Society** Providence, RI.
- Marton, K. (1986): A Simple Proof of the Blowing-up Lemma *IEEE Transactions on Information Theory* **32** 445-446.
- Marton, K. (1996a): Bounding d-distance by Informational Divergence: A Way to prove Measure Concentration *Annals of Probability* **24** 857-866.
- Marton, K. (1996b): A Measure Concentration Inequality for contracting Markov Chains *Geometric and Functional Analysis* **6** 556-571 (Erratum: **7** 609-613 (1997)).
- Massart, P. (1998): Optimal Constants for Hoeffding Type Inequalities *Technical Report **98.86**, Mathematiques* **Universite de Paris-Sud**.
- Massart, P. (2000a): About the Constants in Talagrand's Concentration Inequalities for Empirical Processes *Annals of Probability* **28** 863-884.
- Massart, P. (2000b): Some Applications of Concentration Inequalities to Statistics *Annales de la Faculte des Sciences de Toulouse* **IX** 245-303.
- McDiarmid, C. (1989): On the Method of Bounded Differences, in: *Surveys in Combinatorics 1989* 148-188 **Cambridge University Press** Cambridge.
- McDiarmid, C. (1998): Concentration, in: *Probabilistic Methods for Algorithmic Discrete Mathematics* (M. Habib, C. McDiarmid, J. Ramirez-Alfonsin, and B. Reed) 195-248 **Springer** New York.
- Rhee, W., and M. Talagrand (1987): Martingales, Inequalities, and NP-Complete Problems *Mathematics of Operations Research* **12** 177-181.

- Rio, E. (2001): Inegalities de concentration pour les processus empiriques de classes de parties *Probability Theory and Related Fields* **119** 163-175.
- Samson, P. M. (2000): Concentration of Measure Inequalities for Markov Chains and ϕ-mixing Processes *Annals of Probability* **28** 416-461.
- Shamir, E., and J. Spencer (1987): Sharp Concentration of Chromatic Number on Random Graphs $g_{n,p}$ *Combinatorica* **7** 374-387.
- Talagrand, M. (1995): Concentration of Measure and Isoperimetric Inequalities in Product Spaces *Publications Mathematiques de l'I.H.E.S* **81** 73-205.
- Talagrand, M. (1996a): New Concentration Inequalities in Product Spaces *Inventiones Mathematicae* **126** 505-563.
- Talagrand, M. (1996b): A New Look at Independence *Annals of Probability* **24** 1-34 (Special Invited Paper).
- Vapnik, V. N. (1995): *The Nature of Statistical Learning Theory* **Springer-Verlag** New York.

Concentration of Measure

Introduction

1. Overview: In this section we address the *isoperimetric* approach to concentration inequalities, developed in large parts by Talagrand (1995, 1996a, 1996b).
2. Equivalent Bounded Differences Inequality: First, we provide an equivalent formulation of the bounded differences inequality, which shows that any not-too-small set in a product probability space has the property that the probability of those points whose Hamming distance from the set is much larger than $\sqrt{n}$ is exponentially small.
3. Convex Distance Inequality: Using the full power of the bounded square differences inequality, we provide a significant improvement on this concentration-of-measure result, also called Talagrand' *Convex Distance Inequality*.

Equivalent Bounded Differences Inequality

1. Setup: Consider independent random variables $X_1, \dots, X_n$ taking their values in a measurable set $\mathcal{X}$, and denote a vector of these variables by

$$X_1^n = (X_1, \dots, X_n)$$

   taking its values in X^n.
2. Hamming Distance: Let

$$\mathcal{A} \subset \mathcal{X}^n$$

   be an arbitrary measurable set, and write

$$\mathbb{P}[\mathcal{A}] = \mathbb{P}[X_1^n \in \mathcal{A}]$$

The *Hamming Distance* $d(x_1^n, y_1^n)$ between the vectors

$$x_1^n, y_1^n \in X_1^n$$

is defined as the number of co-ordinates in which x_1^n and y_1^n differ. Introduce

$$d(x_1^n, \mathcal{A}) = \min_{y_1^n \in \mathcal{A}} d(x_1^n, y_1^n)$$

as the Hamming distance between set $\mathcal{A}$ and the point x_1^n.

3. Hamming Distance Inequality: For any

$$t > 0$$

$$\mathbb{P}\left[d(x_1^n, \mathcal{A}) \geq t + \sqrt{\frac{n}{2}\log\frac{1}{\mathbb{P}[\mathcal{A}]}}\right] \leq e^{-\frac{2t^2}{n}}$$

   a. Proof Step #1 – Bounded Differences Property => Observe that the function

$$g(x_1^n) = d(x_1^n, \mathcal{A})$$

   cannot change more than 1 by altering one component of x_1^n, that is, it follows the bounded differences property with constants

$$c_1 = \cdots = c_n = 1$$

   Thus, by the bounded square differences inequality, with the optimal constants provided by the McDiarmid's inequality

$$\mathbb{P}[\mathbb{E}[d(x_1^n, \mathcal{A})] - d(x_1^n, \mathcal{A}) \geq t] \leq e^{-\frac{2t^2}{n}}$$

b. Proof Step #2 - Inequality for $\mathbb{E}[d(x_1^n, \mathcal{A})]$ => By taking

$$t = \mathbb{E}[d(x_1^n, \mathcal{A})]$$

the LHS becomes

$$\mathbb{P}[d(x_1^n, \mathcal{A}) \leq 0] \leq \mathbb{P}(\mathcal{A})$$

so the above inequality implies

$$\mathbb{E}[d(x_1^n, \mathcal{A})] \leq \sqrt{\frac{n}{2}\log\frac{1}{\mathbb{P}[\mathcal{A}]}}$$

By using the bounded differences inequality again, we obtain

$$\mathbb{P}\left[d(x_1^n, \mathcal{A}) \geq t + \sqrt{\frac{n}{2}\log\frac{1}{\mathbb{P}[\mathcal{A}]}}\right] \leq e^{-\frac{2t^2}{n}}$$

as desired.

4. *t-Blowup* of the Set $\mathcal{A}$: Observe that on the RHS we have a measure of the complement of the *t-Blowup* of the set $\mathcal{A}$, that is, the measure of the set of points whose Hamming distance from $\mathcal{A}$ is at least t. If we consider a set, say with

$$\mathbb{P}[\mathcal{A}] = \frac{1}{10^6}$$

we see something very surprising; the measure of the set of points whose Hamming distance to $\mathcal{A}$ is more than $10\sqrt{n}$ is smaller than e^{-108}! In other words, product measures are concentrated on extremely small *sets* – hence the name concentration of measure.

5. Recovering the Bounded Differences Inequality: Observe that the bounded differences inequality may also be derived from the above theorem. Indeed, if we consider a function g on $\mathcal{X}^n$ having the bounded differences property with constants

$$c_1 = \cdots = c_n = 1$$

(for simplicity), then we may let

$$\mathcal{A} = \{x_1^n \in \mathcal{X}^n; g(x_1^n) \leq \mathbb{M}[Z]\}$$

where $\mathbb{M}[Z]$ denotes the median of the random variable

$$Z = g(X_1, \ldots, X_n)$$

Then clearly

$$\mathbb{P}[\mathcal{A}] \geq \frac{1}{2}$$

so the above inequality implies

$$\mathbb{P}\left[Z - \mathbb{M}[Z] \geq t + \sqrt{\frac{n}{2}\log 2}\right] \leq e^{-\frac{2t^2}{n}}$$

Note that the Hamming distance in this case of

$$Z = g(X_1, \ldots, X_n)$$

can be represented by $Z - \mathbb{M}[Z]$.

6. Reduction to the Bounded Differences Form: This has the same form as the bounded differences inequality, except that the expected value of Z is replaced by its mean. This difference is usually negligible, since

$$|\mathbb{E}[Z] - \mathbb{M}[Z]| \leq \mathbb{E}[|Z - \mathbb{M}[Z]|] = \int_0^\infty \mathbb{P}[|Z - \mathbb{M}[Z]| \geq t]dt$$

so whenever the deviation of Z from its mean is small, its expected value must also be close to the mean.

   a. Mean-Median Closeness Statement => Let Z be a random variable with median $\mathbb{M}[Z]$ such that there exist positive constants a and b such that for all

$$t > 0$$

$$\mathbb{P}[|Z - \mathbb{M}[Z]| \geq t] \leq ae^{-\frac{t^2}{b}}$$

Then

$$|\mathbb{E}[Z] - \mathbb{M}[Z]| \leq \frac{a\sqrt{\pi b}}{2}$$

Convex Distance Inequality

1. Motivation: Talagrand (1995, 1996a, 1996b) developed an induction method to prove powerful concentration results in many cases where the bounded differences inequality fails. The most widely used among these is the so-called *convex distance inequality* – Steele (1996) and McDiarmid (1998) contain surveys with several interesting applications.

2. Bounded Square Differences Convex Differences Inequality: Here we use the bounded square differences inequality to derive a version of the convex distance inequality. Talagrand (1995, 1996a, 1996b) contains extensions and several variations.
3. Weighted Hamming Distance: The following simple argument first presented in McDiarmid (1998) helps understand the rationale behind Talagrand's inequality. First, observe that the *Equivalent Bounded Square Differences Theorem* may be easily generalized by allowing the distance of point X_1^n from the set $\mathcal{A}$ to be measured by a *Weighted Hamming Distance*

$$d_\alpha(x_1^n, \mathcal{A}) = \inf_{y_1^n \in \mathcal{A}} d_\alpha(x_1^n, y_1^n) = \inf_{y_1^n \in \mathcal{A}} \sum_{i:x_i \neq y_i} |\alpha_i|$$

where

$$\alpha = (\alpha_1, \ldots, \alpha_n)$$

is a vector of non-negative numbers.

4. Applying the Equivalent Bounded Square Differences to the Weighted Hamming Distance: Repeating the argument used in the proof of the *Equivalent Bounded Square Differences Theorem* for the Weighted Hamming Distance, we obtain, for all α

$$\mathbb{P}\left[d_\alpha(X_1^n, \mathcal{A}) \geq t + \sqrt{\frac{\|\alpha\|^2}{2} \log \frac{1}{\mathbb{P}[\mathcal{A}]}}\right] \leq e^{-\frac{2t^2}{\|\alpha\|^2}}$$

where

$$\|\alpha\|^2 = \sum_{i=1}^{n} {\alpha_i}^2$$

denotes the Euclidean norm of α - recall that $\|\alpha\|^2$ serves the role of

$$c = \sum_{i=1}^{n} {c_i}^2$$

in the bounded square differences inequality.

5. Reduction to Euclidean Norm: For all vectors α with unit norm

$$\|\alpha\|^2 = 1$$

$$\mathbb{P}\left[d_\alpha(X_1^n, \mathcal{A}) \geq t + \sqrt{\frac{1}{2}\log\frac{1}{\mathbb{P}[\mathcal{A}]}}\right] \leq e^{-2t^2}$$

Thus, denoting

$$u = \sqrt{\frac{1}{2}\log\frac{1}{\mathbb{P}[\mathcal{A}]}}$$

for any

$$t \geq u$$

we get

$$\mathbb{P}[d_\alpha(X_1^n, \mathcal{A}) \geq t] \leq e^{-2(t-u)^2}$$

6. Reduction to the Talagrand Form: On the one hand, if

$$u = \sqrt{2\log\frac{1}{\mathbb{P}[\mathcal{A}]}}$$

$$\mathbb{P}[\mathcal{A}] \leq e^{-\frac{t^2}{2}}$$

On the other hand, since

$$(t-u)^2 \geq \frac{t^2}{4}$$

for

$$t \geq 2u$$

for any

$$t \geq \sqrt{2\log\frac{1}{\mathbb{P}[\mathcal{A}]}}$$

the inequality implies

$$\mathbb{P}[d_\alpha(X_1^n, \mathcal{A}) \geq t] \leq e^{-\frac{t^2}{2}}$$

Thus, for all

$$t > 0$$

we have

$$\mathbb{P}[\mathcal{A}].\mathbb{P}[d_\alpha(X_1^n, \mathcal{A}) \geq t] \leq \min(\mathbb{P}[\mathcal{A}], \mathbb{P}[d_\alpha(X_1^n, \mathcal{A}) \geq t]) \leq e^{-\frac{t^2}{2}}$$

7. Talagrand Supremum Form: The main impact of the Talagrand's inequality is that the above inequality remains that even if the supremum over $\|\alpha\|$ with unit norm is taken with probability, i.e.

$$\sup_{\|\alpha\|: \|\alpha\| = 1} \mathbb{P}[\mathcal{A}].\mathbb{P}[d_\alpha(X_1^n, \mathcal{A}) \geq t] \leq \sup_{\|\alpha\|: \|\alpha\| = 1} \min(\mathbb{P}[\mathcal{A}], \mathbb{P}[d_\alpha(X_1^n, \mathcal{A}) \geq t]) \leq e^{-\frac{t^2}{2}}$$

8. Relation of $\|\alpha\|$ to Chernoff/Hoeffding Type Optimization Parameters: To make above statement more precise, we introduce, for any

$$x_1^n = (x_1, \ldots, x_n) \in \mathcal{X}^n$$

the *convex distance* of x_1^n from the set $\mathcal{A}$ by

$$d_\tau(x_1^n, \mathcal{A}) = \sup_{\alpha \in [0,\infty)^n: \|\alpha\| = 1} d_\alpha(x_1^n, \mathcal{A})$$

Using this convex distance metric, we are now ready to derive the prototypical result from Talagrand (1995) – for an even stronger concentration-of-measure result, refer to Talagrand (1996a).

Convex Distance Inequality - Proof

1. Statement: For any subset

$$\mathcal{A} \subseteq \mathcal{X}^n$$

with

$$\mathbb{P}[X_1^n \in \mathcal{A}] \geq \frac{1}{2}$$

and

$$t > 0$$

$$\min(\mathbb{P}[\mathcal{A}], \mathbb{P}[d_\tau(X_1^n, \mathcal{A}) \geq t]) \leq e^{-\frac{t^2}{4}}$$

2. Step #1 - Saddle Point Setup: Define the random variable

$$Z = d_\tau(X_1^n, \mathcal{A})$$

First we observe that $d_\tau(x_1^n, \mathcal{A})$ can be expressed as a saddle-point. Let $\mathcal{M}(\mathcal{A})$ denote the set of probability measure on $\mathcal{A}$. Then

$$d_\tau(x_1^n, \mathcal{A}) = \sup_{\alpha: \|\alpha\| \leq 1} \inf_{\nu \in \mathcal{M}(\mathcal{A})} \sum_j \alpha_j \mathbb{E}_\nu \left[\mathbb{I}_{x_j \neq Y_j}\right]$$

where Y_1^n is distributed according to ν

$$d_\tau(x_1^n, \mathcal{A}) = \inf_{\nu \in \mathcal{M}(\mathcal{A})} \sup_{\alpha: \|\alpha\| \leq 1} \sum_j \alpha_j \mathbb{E}_\nu \left[\mathbb{I}_{x_j \neq Y_j}\right]$$

thereby the saddle point is achieved.

3. Step #2 - Saddle Point Establishment: We apply Sion's theorem (Sion (1958)), which may be viewed at the association inequality applied to functional space saddle points. Sion's minimax theorem states that if $f(x, y)$ denotes a function

$$\mathcal{X} \times \mathcal{Y} \to \mathbb{R}$$

that is convex and lower semi-continuous with respect to x, concave and upper semi-continuous with respect to y, where $\mathcal{X}$ is convex and compact, then

$$\inf_{x} \sup_{y} f(x,y) = \sup_{y} \inf_{x} f(x,y)$$

Additional details relating to checking the conditions of Sion's theorem are available in Boucheron, Lugosi, and Massart (2003).

4. Step #3 - Applying the Saddle Point: Let now $(\hat{\alpha}, \hat{\nu})$ be a saddle point for x_1^n. We have

$$Z_i' = \inf_{\nu \in \mathcal{M}(\mathcal{A})} \sup_{\alpha} \sum_j \alpha_j \mathbb{E}_\nu \left[\mathbb{I}_{x_j{}^{(i)} \neq Y_j}\right] \geq \inf_{\nu \in \mathcal{M}(\mathcal{A})} \sum_j \hat{\alpha}_j \mathbb{E}_\nu \left[\mathbb{I}_{x_j{}^{(i)} \neq Y_j}\right]$$

where

$$x_j{}^{(i)} = x_j$$

if

$$j \neq i$$

and

$$x_j{}^{(i)} = x_j{}'$$

Let $\hat{\nu}$ denote the distribution on $\mathcal{A}$ that achieves the infimum in the latter expression. Now we have

$$Z = \inf_{\nu} \sum_j \hat{\alpha}_j \mathbb{E}_\nu \left[\mathbb{I}_{x_j{}^{(i)} \neq Y_j}\right] \leq \sum_j \hat{\alpha}_j \mathbb{E}_{\hat{\nu}} \left[\mathbb{I}_{x_j{}^{(i)} \neq Y_j}\right]$$

Hence, we get

$$Z - Z_i' \leq \sum_j \hat{\alpha}_j \mathbb{E}_{\hat{\nu}} \left[\mathbb{I}_{x_j \neq Y_j} - \mathbb{I}_{x_j^{(i)} \neq Y_j} \right] \leq \hat{\alpha}_i \mathbb{E}_{\hat{\nu}} \left[\mathbb{I}_{x_i \neq Y_i} - \mathbb{I}_{x_i^{(i)} \neq Y_i} \right] \leq \hat{\alpha}_i$$

5. Step #4 - Use of the Bounded Square Differences Inequality: From above, we can see that

$$\sum_{i=1}^{n} (Z - Z_i')^2 \mathbb{I}_{Z > Z_i'} \leq \sum_{i=1}^{n} \hat{\alpha}_i{}^2 = 1$$

Thus, by the bounded square differences inequality (or, more precisely, using its generalized expectations), for any

$$t > 0$$

$$\mathbb{P}[d_\tau(X_1^n, \mathcal{A}) - \mathbb{E}[d_\tau(X_1^n, \mathcal{A})] \geq t] \leq e^{-\frac{t^2}{4}}$$

6. Step #5 - Application of the Equivalent Bounded Square Differences Inequality for the Upper Tail: Applying the above to the lower tail, we get

$$\mathbb{P}[d_\tau(X_1^n, \mathcal{A}) - \mathbb{E}[d_\tau(X_1^n, \mathcal{A})] \leq -t] \leq e^{-\frac{t^2}{4(e-1)}}$$

which by taking

$$t = \mathbb{E}[d_\tau(X_1^n, \mathcal{A})]$$

implies

$$\mathbb{E}[d_\tau(X_1^n, \mathcal{A})] \leq \sqrt{4(e-1) \log \frac{1}{\mathbb{P}[\mathcal{A}]}}$$

Thus, this means

$$\mathbb{P}\left[d_\tau(X_1^n, \mathcal{A}) - \sqrt{4(e-1)\log\frac{1}{\mathbb{P}[\mathcal{A}]}} \geq t\right] \leq e^{-\frac{t^2}{4}}$$

7. Step #6 - Reduction to the Talagrand Form: Now, if

$$0 < u < \sqrt{4\log\frac{1}{\mathbb{P}[\mathcal{A}]}}$$

then

$$\mathbb{P}[\mathcal{A}] \leq e^{-\frac{u^2}{4}}$$

On the other hand, if

$$u \geq \sqrt{4\log\frac{1}{\mathbb{P}[\mathcal{A}]}}$$

then

$$\mathbb{P}[d_\tau(X_1^n, \mathcal{A}) \geq u] \leq \mathbb{P}\left[d_\tau(X_1^n, \mathcal{A}) - \sqrt{4(e-1)\log\frac{1}{\mathbb{P}[\mathcal{A}]}} \geq u\left[1 - \sqrt{\frac{e-1}{e}}\right]\right]$$

$$\leq e^{-\frac{u^2\left[1-\sqrt{\frac{e-1}{e}}\right]^2}{4}}$$

where the second inequality follows from the upper-tail inequality above.

8. Step #7 - Bringing it all together: In conclusion, for all

$$u > 0$$

$$\min(\mathbb{P}[\mathcal{A}], \mathbb{P}[d_\tau(X_1^n, \mathcal{A}) \geq u]) \leq e^{-\frac{u^2\left[1-\sqrt{\frac{e-1}{e}}\right]^2}{4}}$$

which concludes the proof of the convex distance inequality. The constant in the exponent may be improved by other means – say using McDiarmid's inequality.

Application of the Convex Distance Inequality – Bin Packing

1. Overview: Here we apply the convex distance inequality to the bin packing problem (Talagrand (1995)). Let $g(x_1^n)$ denote the minimum number of bins of size 1 into which the numbers

$$x_1, \ldots, x_n \in [0, 1]$$

can be packed. We consider the random variable

$$Z = g(X_1^n)$$

where $X_1, \ldots, X_n$ are independent, and take values in $[0, 1]$.

2. Convex Distance Inequality for Bin Packing: For each

$$t > 0$$

$$\mathbb{P}[|Z - \mathbb{M}[Z]| \geq t + 1] \leq 8e^{-\frac{t^2}{16\left(2\mathbb{E}\left[\sum_{i=1}^n X_i^2\right]+t\right)}}$$

3. Proof Step #1 - Symmetrization of Bin Packing: We first establish (and this is the only specific property of g that we use in the proof) that for any

$$x_1^n, y_1^n \in [0,1]^n$$

$$g(x_1^n) \leq g(y_1^n) + 2 \sum_{i:x_i \neq y_i} x_i + 1$$

To see this, it suffices to show that those x_i for which

$$x_i \neq y_i$$

can be packed into at most $2\sum_{i:x_i \neq y_i} x_i + 1$ bins. For this, it is enough to find a packing such that at most one bin is less than half full. But such a packing must exist, since we can always pack the contents of two half-empty bins into one.

4. Proof Step #2 - Recasting this into the Hamming Distance Metric: Denoting by

$$\alpha = \alpha(x_1^n) \in [0,\infty)^n$$

the unit vector $\frac{x_1^n}{\|x_1^n\|}$, we clearly have

$$\sum_{i:x_i \neq y_i} x_i = \|x_1^n\| \sum_{i:x_i \neq y_i} \alpha_i = \|x_1^n\| d_\alpha(x_1^n, y_1^n)$$

5. Proof Step #3 - Symmetrization Bounding Using Convex Distance: Let a be a positive number, and define the set

$$\mathcal{A}_a = \{y_1^n : g(y_1^n) \leq a\}$$

Then, by the argument above and using the definition of convex distance, for each

$$x_1^n \in [0,1]^n$$

there exists a

$$y_1^n \in \mathcal{A}_a$$

such that

$$g(x_1^n) \leq g(y_1^n) + 2 \sum_{i:x_i \neq y_i} x_i + 1 \leq a + 2\|x_1^n\| d_\alpha(x_1^n, y_1^n) + 1$$

from which we set that for each

$$a > 0$$

$$Z \leq a + 2\|x_1^n\| d_\alpha(x_1^n, y_1^n) + 1$$

6. Proof Step #4 - Applying the Hamming Distance Metric: From the above, we have for any

$$t > 0$$

$$\mathbb{P}[Z \geq a + 1 + t]$$
$$\leq \mathbb{P}\left[Z \geq a + 1 + t \frac{2\|X_1^n\|}{2\sqrt{2\mathbb{E}\left[\sum_{i=1}^n {X_i}^2\right] + t}}\right] + \mathbb{P}\left[\|X_1^n\| \geq 2\sqrt{2\mathbb{E}\left[\sum_{i=1}^n {X_i}^2\right] + t}\right]$$

7. Proof Step #5 - Application of the Bernstein Inequality: Given that $X_1, \ldots, X_n$ are independent random variables taking values in $[0,1]$, it may be shown that, by applying the Bernstein inequality to the sum $\sum_{i=1}^n {X_i}^2$

$$\mathbb{P}\left[\sqrt{\sum_{i=1}^{n} {X_i}^2} \geq \sqrt{2\mathbb{E}\left[\sum_{i=1}^{n} {X_i}^2\right] + t}\right] \leq e^{-\frac{3}{8}\left(\mathbb{E}\left[\sum_{i=1}^{n} {X_i}^2\right]+t\right)}$$

Using this along with the Hamming metric in Step #7, we get

$$\mathbb{P}[Z \geq a + 1 + t] \leq \mathbb{P}\left[d_\tau(X_1^n, \mathcal{A}_a) \geq \frac{t}{2\sqrt{2\mathbb{E}\left[\sum_{i=1}^{n} {X_i}^2\right] + t}}\right] + e^{-\frac{3}{8}\left(\mathbb{E}\left[\sum_{i=1}^{n} {X_i}^2\right]+t\right)}$$

8. Proof Step #6 - Acquiring the Bounds: To obtain the desired inequality, we incorporate two difference choices of a on the inequality above. To derive the bound for the upper tail, we take

$$a = \mathbb{M}[Z]$$

Then

$$\mathbb{P}[\mathcal{A}_a] \geq \frac{1}{2}$$

and the convex distance inequality yields

$$\mathbb{P}[Z \geq \mathbb{M}[Z] + 1 + t] \leq 2\left\{e^{-\frac{t^2}{16\left(2\mathbb{E}\left[\sum_{i=1}^{n} {X_i}^2\right]+t\right)}} + e^{-\frac{3}{8}\left(\mathbb{E}\left[\sum_{i=1}^{n} {X_i}^2\right]+t\right)}\right\} \leq 4e^{-\frac{t^2}{16\left(2\mathbb{E}\left[\sum_{i=1}^{n} {X_i}^2\right]+t\right)}}$$

We obtain a similar equality in the same way for $\mathbb{P}[Z \leq \mathbb{M}[Z] - 1 - t]$ by taking

$$a = \mathbb{M}[Z] - 1 - t$$

References

- Boucheron, S., G. Lugosi, and P. Massart (2003): Concentration Inequalities using the Entropy Method *Annals of Probability* **31** 1583-1614.
- McDiarmid, C. (1998): Concentration, in: *Probabilistic Methods for Algorithmic Discrete Mathematics* (M. Habib, C. McDiarmid, J. Ramirez-Alfonsin, and B. Reed) 195-248 **Springer** New York.
- Sion, M. (1958): On General Minimax Theorems *Pacific Journal of Mathematics* **8** 171-176.
- Steele, J. M. (1996): Probability Theory and Combinatorial Optimization *SIAM, CBMS-NSF Regional Conference Series in Applied Mathematics* **Philadelphia**.
- Talagrand, M. (1995): Concentration of Measure and Isoperimetric Inequalities in Product Spaces *Publications Mathematiques de l'I.H.E.S* **81** 73-205.
- Talagrand, M. (1996a): New Concentration Inequalities in Product Spaces *Inventiones Mathematicae* **126** 505-563.
- Talagrand, M. (1996b): A New Look at Independence *Annals of Probability* **24** 1-34 (Special Invited Paper).

Standard SLT Framework

Statistical Learning Theory Monographs

1. Mathematical and Technical Surveys: Fukunaga (1972), Duda and Hart (1973), Vapnik and Chervonenkis (1974), Vapnik (1982), Breiman, Friedman, Olshen, and Stone (1984), Natarajan (1991), McLachlan (1992), Kearns and Vazirani (1994), Vapnik (1995), Devroye, Gyorfi, and Lugosi (1996), Kulkarni, Lugosi, and Venkatesh (1998), Vapnik (1998), Anthony and Bartlett (1999), Herbrich and Williamson (2002), Lugosi (2002), Scholkopf and Smola (2002), Bosquet, Boucheron, and Lugosi (2003), Mandelson (2003), Boucheron, Bousquet, and Lugosi (2005).

Computational Learning Theory

1. Purpose: Computational learning theory is a mathematical field related to the analysis of machine learning algorithms. It seeks to address issues of:
    a. Learnability/Learn Feasibility/Learn Complexity
    b. Learning Speed/Performance
    c. Learning Formulation Criterion
    d. Learning Basis/Representation Choice
    e. Probabilistic Learning Framework and Viability
4. Learn Complexity: In addition to performance bounds computational learning theory studies the time complexity and flexibility of learning. In computational learning theory, a computation is considered feasible if it can be performed in polynomial time (Computational Learning Theory (Wiki)). There are two kinds of time complexity results:
    a. Positive Results => Showing that a certain class of functions is learnable in polynomial time.

   b. Negative Results => Showing that a certain class of functions cannot be learned in polynomial time.

5. Non-Polynomial Time Results: Negative results often rely on commonly believed, but as yet unproven assumptions such as:
   a. Computational Complexity –

$$P \neq NP$$

   b. Cryptographic – One-way functions exist

6. Types of Approaches: There are several different approaches to computational learning theory. These differences are based on making assumptions about inference principles used to generalize from limited data. These include different conceptions of probability (frequentist, Bayesian) and different assumptions on the generation of samples.
7. Computational Learning Theory Approaches:
   a. Probably Approximately Correct Learning (PAC) proposed by Valiant (1984)
   b. VC Theory proposed by Vapnik and Chervonenkis (1971)
   c. Bayesian Inference
   d. Algorithmic learning theory, from the work of Gold (1967)
   e. Online machine learning algorithms, from the work of Nick Littlestone
8. Practical Algorithms from Computational Learning Theory: PAC theory inspired boosting, VC theory led to support vector machines, and Bayesian inference led to belief networks (by Judea Pearl).

Probably Approximately Correct (PAC) Learning

1. PAC Learning Motivation: PAC learning is a framework for mathematical analysis of machine learning (Probably Approximately Correct Learning (Wiki), Valiant (1984)). In this framework, the learner selects a generalization hypothesis from a certain class of possible functions. The goal is that, with high probability (the “probably” part) the selected hypothesis function will have a low generalization error (The “approximately” part). The learner must be

able to learn the concept given any arbitrary approximation ratio, probability of success, or distribution of samples. The original model has been expanded later to treat noise (misclassified samples).

   a. Probably/Approximately => This model uses $P(x, y)$, which is the "probably" part, and within that chooses the low error, the "approximately" part. Since $P(x, y)$ is explicit, the treatment in regards to empirical risk is different.

2. The PAC Approach and Goals: An important innovation of the PAC framework is the introduction of the computational complexity theory concepts to machine learning (for e.g., VC deals primarily with "hypothesis complexity"). In particular, the learner is expected to find efficient functions (i.e., the time and space requirements are bounded to a polynomial on the sample size), and the learner itself must implement an efficient procedure (requiring an example count to be bounded to a polynomial of the concept size, modified by the approximation and the likelihood bounds).

PAC Definitions and Terminology

1. Sample Background: To elaborate on the PAC learnability, we use the 2 samples employed in Natarajan (1991) and Kearns and Vazirani (1994). The first is the problem of character recognition given an array of bits encoding a binary image. The other is the problem of locating the interval that correctly classifies points inside as positive and outside as negative.
2. The PAC Instance Space: Let X be a set called the instance space or encoding of all the samples, and each instance has a length assigned. In the character recognition problem, the instance space is $\{0, 1\}^n$. In the interval problem, this would be

$$X = \mathbb{R}$$

   where $\mathbb{R}$ denotes the set of all real numbers.

3. PAC Concept and the Concept Class: A concept is the subset

$$c \in X$$

One concept is the set of all patterns in bits

$$X = \{0,1\}^n$$

that encode the picture of the letter P. An example concept from the second example is the set of all numbers between $\frac{\pi}{2}$ and 10. A concept class C is a set of concepts over X. For instance, this could be the set of all the subsets of array of bits that are skeletonized 4-connected (the width of the font being 1).

4. The PAC Procedure: Let $EX(c, D)$ be a procedure that draws an example x using a probability distribution D and gives the correct label

$$c(x) = \begin{cases} 1 & x \in C \\ 0 & Else \end{cases}$$

5. PAC Learnability and the Learning Algorithm: Say that there is an algorithm A that, given access to $EX(c, D)$ alongwith inputs ε and δ that, with a probability of at least $1 - \delta$, A outputs a hypothesis h that has error less than or equal to ε with examples drawn from X using the distribution D. If there is an algorithm for every concept

$$c \in C$$

for every distribution D over X, and for all

$$0 < \varepsilon < \frac{1}{2}$$

and

$$0 < \delta < \frac{1}{2}$$

then C is PAC learnable (or distribution-free PAC learnable). Further, we can also say that A is a PAC learning algorithm for C.

6. Efficient PAC Algorithm: An algorithm runs in time t if it draws at most t samples and requires at most t time steps. A concept class is efficiently PAC learnable if it is PAC learnable by an algorithm that runs in time polynomial in $\frac{1}{\varepsilon}$, $\frac{1}{\delta}$, and the instance length.

SLT Introduction

1. Motivation: Statistical Learning Theory provides the framework for machine learning by drawing from the fields of statistics and functional analysis (Statistical Learning Theory (Wiki), Mohri, Rostamizadeh, and Talwalkar (2012)), with a wide variety of applications (e.g., Sidhu and Caffo (2014))
2. Binary Classification in SLT: The reason that the binary classifier is well-studied is that the classifier prediction error is easily reducible to a) estimation error/precision, and b) approximation error/accuracy. For multinomial and/or linear/non-linear classifiers, other factors will need to be set.
3. Error Factors in Multinomial Classification and Regression: In multinomial classification, the error factor contributions come from each of the component error contributions per label/class above. For regression, in addition to the nodal "best-fit" error contributions (due to the components above), penalization may be applied to limit curvature and improve smoothness.
4. SLT Goals: These are spelt out in von Luxburg and Scholkopf (2008):
    a. Which learning tasks can be performed by computers in general (producing positive/negative results)?
    b. What kind of assumptions do we have to make to ensure that such machine learning tasks are successful?
    c. What key properties does a learning algorithm need to satisfy in order to be successful?
    d. What performance guarantees can we provide on certain learning algorithms?

The Setup

1. Empirical Error Determinants: In general, the empirical error depends upon:
    a. The Joint Probability Distribution $P(x, y)$
    b. The Sample Size n
    c. The Function Space/Sub-space Complexity
2. Sample vs. Population Mean: Likewise, the probability that the sample mean approaches the population mean for a finite sample of size n to within a tolerance δ depends on:
    a. The tolerance δ
    b. The sample size n
    c. The Joint Probability Distribution $P(x, y)$
    d. The Function Complexity – the complexity of the function proxying $P(x, y)$
3. SLT Objectives:
    a. No assumptions on $P(x, y)$
    b. Labels can be non-deterministic so as to be able to accommodate label noise and overlapping label classes
    c. Sampling i.i.d.
    d. $P(x, y)$ is unknown at the time of learning
4. Use of $P(x, y)$ in SLT: In light of the assumptions above, SLT seeks to make statements on the empirical error and sample mean departure probability without making references to specific forms for $P(x, y)$, i.e., only using sample size, function space complexity, and tolerance.
5. Loss and Risk: The loss function l is the cost of misclassifying one observation point, and is written as

$$l[X, Y, f(X)] = \begin{cases} 1, & f(X) \neq Y \\ 0, & f(X) = Y \end{cases}$$

Risk of a function is the average of the loss over the data points generated according to the underlying distribution $P(x, y)$, so the distribution DOES factor in

$$R(f) = E\{l[X, Y, f(X)]\}$$

Empirical risk, however, is simply a miscalculation counter.

6. Bayes' Classifier:

$$f_{Bayes}(x) = \begin{cases} -1, & P(Y = 1|X = x) \geq 0.5 \\ +1, & P(Y = 1|X = x) < 0.5 \end{cases}$$

Almost certainly, Bayes' classifier does not possess zero empirical risk. Further, it has explicit dependence on $P(x, y)$ (or $P(y|x)$).

7. SLT Problem Statement: Given a set of training points $(X_1, Y_1), \ldots, (X_n, Y_n)$ that have been drawn from an unknown distribution $P(x, y)$, and given a loss function l, how can we construct a function

$$f: X \to Y$$

that has its risk $R(f)$ as close as possible to that of the Bayes' classifier?

Algorithms for Reducing Over-fitting

1. Statistical Regularization: Overfitting is symptomatic of unstable solution, i.e., small training perturbations can cause large shifts in the learned functions. It can be shown that if the stability for the solution can be guaranteed, generalization and consistency are guaranteed as well (Vapnik, Chervonenkis (1971), Mukherjee, Niyogi, Poggio, and Rifkin (2006)). Regularization can address over-fitting and provide problem stability.
2. The Strategy: Essentially 2 ways. The first way is to restrict the function class space (say, via the classical SLT and ERM). The second is to modify the criterion to be minimized using a penalizer for "complicated" functions. Structural Risk Minimization, Standardized Regularization, and Normalization all combine the first and the second.

3. Structure Risk Minimization: We discuss this in detail later on. The idea here is to choose an infinite sequence

$$\{\Im_d : d = 1,2, \dots\}$$

of models of increasing size and to minimize the empirical risk in each model with an added penalty term for the model size, i.e.

$$f_n = \underset{f \in \Im_d, d \in \mathbb{N}}{\arg min} R_n(f) + pen(d,n)$$

The penalty $pen(d,n)$ gives preference to those models whose estimation error is small, and thus measures the size/capacity of the model.

4. Standardized Regularization: This easier to implement approach consists in choosing of a large model $\Im$ (possibly dense in continuous functions, for example), and to define on $\Im$ a regularizer, usually a norm $\|f\|$. One then minimizes the regularized empirical risk

$$f_n = \underset{f \in \Im}{\arg min} R_n(f) + \lambda \|f\|^2$$

Most existing and successful methods can be thought of as a variant of the regularization method.

5. Regularization by Hypothesis Restriction: Regularization can also occur by restricting the hypothesis space. A common example is the restriction of H to linear functions – thereby reducing the problem to that of linear regression. H may also be reduced to polynomials of degree p, exponentials, or bounded functions on L_1. Restrictions of the hypothesis space avoids over-fitting because the form of the possible functions is limited, and therefore may avoid choosing the function that gives the empirical risk to be arbitrarily close to zero.
6. Tikhonov Regularization: This consists of minimizing

$$\frac{1}{n}\sum_{i=1}^{n} L[h(x_i), y_i] + \gamma \|f\|_H{}^2$$

where γ is a fixed positive parameter referred to as the regularization parameter. Tikhonov regularization ensures the existence, uniqueness, and stability of the solution (Poggio, Rosasco, Ciliberto, Frogner, Evangelopoulus (2012)).

7. Normalized Regularization: In addition to the above regularization approaches, there are other possible approaches where the regularizer can, in some sense, be "normalized", i.e., when it corresponds to some probability distribution over $\mathfrak{I}$.

Bayesian Normalized Regularizer Setup

1. Regularization from Probability: Given a probability distribution π (referred to as a prior) defined on $\mathfrak{I}$, one can use $-\log \pi(f)$ as the regularizer. In case $\mathfrak{I}$ is countably/uncountably infinite/continuous, we use the density associated with $\pi(f)$ instead.
2. "Prior" from the Regularizer: Reciprocally, from the regularizer of the form $\|f\|^2$, if there exists a measure μ on $\mathfrak{I}$ such that

$$\int e^{-\lambda \|f\|^2} d\mu(f) < \infty$$

for some

$$\lambda > 0$$

then one may construct a prior corresponding to this regularizer.

   a. As an example, if $\mathfrak{I}$ is a set of hyper-planes in $\mathbb{R}^d$ going through the origin, $\mathfrak{I}$ can be identified with $\mathbb{R}^d$, and taking μ as the Lebesgue measure, it is possible to go from Euclidean norm regularizer to spherical Gaussian measure on $\mathbb{R}^d$ as a prior.

3. RKHS Generalization: Generalization to the infinite dimensional Hilbert spaces can also be done, but requires more care. One can, for example, establish a correspondence between the norm of a Reproducing Kernel Hilbert Space and a Gaussian process prior whose covariance function is the kernel of this space.
4. Posterior: From this type of normalized regularizer (or its posterior), we can construct another probability distribution ρ on $\mathfrak{I}$ (typically called the posterior) as

$$\rho(f) = \frac{e^{-\gamma R_n(f)}}{\mathbb{Z}(\gamma)}\pi(f)$$

where

$$\gamma \geq 0$$

is a free parameter and $\mathbb{Z}(\gamma)$ is a normalization factor.

5. Uses of the Posterior:
    a. MAP as the Regularization Process => If we maximize ρ, we recover the original regularization framework as

$$\underset{f \in \mathfrak{I}}{\arg max}\ \rho(f) = \underset{f \in \mathfrak{I}}{\arg max}\ [\gamma R_n(f) - \log \pi(f)]$$

where the regularizer is $-\frac{\log \pi(f)}{\gamma}$ (note that maximizing $\gamma R_n(f) - \log \pi(f)$ is equivalent to minimizing $R_n(f) - \frac{\log \pi(f)}{\gamma}$).

    b. Randomization of Predictions => ρ can also be used to randomize the predictions. In this case, before computing the predicted labels for the input x, one samples a function f according to ρ and computes $f(x)$. This procedure is called Gibbs classification.
    c. Bayesian Averaging => Another way in which the ρ constructed above can be used is by taking the expected prediction of the functions in $\mathfrak{I}$;

$$f_n(x) = sign\left[\mathbb{E}_\rho\{f(x)\}\right]$$

This is called Bayesian averaging.

References

- Anthony, M., and P. L. Bartlett (1999): *Neural Network Learning: Theoretical Foundations* **Cambridge University Press** Cambridge.
- Boucheron, S., O. Bousquet, and G. Lugosi (2005): Theory of Classification *ESAIM: Probability and Statistics* **9** 323-375.
- Bousquet, O., S. Boucheron, and G. Lugosi (2003): Introduction to Statistical Learning Theory 169-207 *Advances in Machine Learning* (editors: Bousquet, O., U. von Luxburg, and G. Ratsch) **Springer** Berlin.
- Breiman, L., Friedman, J., Olshen, R., and C. Stone (1984): *Classification and Regression Trees* **Wadsworth International** Belmont, CA.
- Computational Learning Theory (Wiki): *Wikipedia Entry for Computational Learning Theory*.
- Devroye, L., L. Gyorfi, and G. Lugosi (1996): *A Probabilistic Theory of Pattern Recognition* **Springer** New York.
- Duda, R., and P. Hart (1973): *Pattern Classification and Scene Analysis* **John Wiley** New York.
- Fukunaga, K. (1972): *Introduction to Statistical Pattern Recognition* **Academic Press** New York.
- Gold, E. (1967): Language Identification in the Limit, *Information and Control* **10 (5)**: 447-474.
- Herbrich, R., and R.C. Williamson (2002): Learning and Generalization: Theoretical Bounds, in *Handbook of Brain Theory and Neural Networks* (editor: M Arbib).
- Kearns, M., and U. Vazirani (1994): *An Introduction to Computational Learning Theory* **MIT Press** Cambridge, MA.

- Kulkarni, S., G. Lugosi, and S. Venkatesh (1998): Learning Pattern Classification – A Survey *IEEE Transaction on Information Theory* **44** 2178-2206 *Information Theory: 1948-1998. Commemorative Special Issue*.
- Lugosi, G. (2002): Pattern Classification and Learning Theory, in: *Principles of Nonparametric Learning* (editor: L Gyorfi) 5-62 **Springer** Vienna.
- Mandelson, S. (2003): A few Notes on Statistical Learning Theory *Advanced Lectures on Machine Learning* **LNCS** 1-40 **Springer**.
- McLachlan, G. (1992): *Discriminant Analysis and Statistical Pattern Recognition* **John Wiley** New York.
- Mohri, M., A. Rostamizadeh, and A. Talwalkar (2012): *Foundations of Machine Learning* **The MIT Press**.
- Mukherjee, S., P. Niyogi, T. Poggio, and R. Rifkin (2006): Learning Theory: Stability is sufficient for Generalization, and necessary and sufficient for Consistency of Empirical Risk Minimization *Advances in Computational Mathematics* **25** 161-193.
- Natarajan, B. K. (1991): *Machine Learning – A Theoretical Approach* **Morgan Kaufmann Publishers**.
- Poggio, T., L. Rosasco, C. Ciliberto, C. Frogner, and G. Evangelopoulus (2012): *Statistical Learning Theory and Applications*
- Probably Approximately Correct Learning (Wiki): *Wikipedia Entry for Probably Approximately Correct Learning*.
- Scholkopf, B. and A. Smola (2002): *Learning with Kernels* **MIT Press** Cambridge, MA.
- Sidhu, G., and B. Caffo (2014): Exploiting Pitcher Decision-making using Reinforcement Learning *Annals of Applied Statistics* **8 (2)** 926-955.
- Statistical Learning Theory (Wiki): *Wikipedia Entry for Statistical Learning Theory*.
- Valiant, L. (1984): A Theory of the Learnable *Communications of the ACM* **27 (11)** 1134-1142.
- Vapnik, V. and A. Chervonenkis (1971): On the Uniform Convergence of relative Frequencies of Events to their Probabilities *Theory of Probability and its Applications* **16 (2)** 264-280.
- Vapnik, V., and A. Chervonenkis (1974): *Theory of Pattern Recognition (in Russian)* **Nauka** Moscow.

- Vapnik, V. (1982): *Estimation of Dependencies based on Empirical Data* **Springer Verlag** New York.
- Vapnik, V. (1995): *The Nature of Statistical Learning Theory* **Springer Verlag** New York.
- Vapnik, V. (1998): *Statistical Learning Theory* **Springer** New York.
- von Luxburg, U., and B. Scholkopf (2005): Statistical Learning Theory: Models, Concepts, and Results, in *Handbook of the History of Logic* (editors: D. Gabbay, S. Hartmann, and J. Woods).

Generalization and Consistency

Concept Motivation

1. Good Generalizer: A classifier f_n generalizes well if the difference $\left|R(f_n) - R_{Emp}(f_n)\right|$ between the overall risk $R(f_n)$ and the empirical risk $R_{Emp}(f_n)$ is small. This does NOT mean that the classifier has a small overall error R_{Emp}; it simply means that the empirical error $R_{Emp}(f_n)$ is a good estimate of the true error $R(f_n)$ (Scholkopf and Smola (2002)).
2. The Concept of Consistency: The notion of consistency in SLT is the same as that in statistics; it aims to make a statement about what happens to the learning with more and more sample points – the expectation for an appropriate algorithm would be to converge to the "optimal" solution (in hopefully one direction, asymptotically). As opposed to generalization, SLT consistency is not a property of f_n, it is a property of the hypothesis space $\Im$ (von Luxburg and Scholkopf (2008)).

Types of Consistency

1. The Setup: Let $(X_i, Y_i)_{i \in N}$ be an infinite sequence of training points that have been drawn independently from some probability distribution $P(x, y)$. Let l be a loss function. For each

$$n \in N$$

   let f_n be a classifier constructed by some learning algorithm on the basis of the first n training points.
2. Consistency with respect to $\Im$ and $P(x, y)$: The learning algorithm is called consistent with respect to $\Im$ and $P(x, y)$ if the risk $R(f_n)$ converges in probability to the risk $R(f_\Im)$ of the best classifier in $\mathfrak{I}$, that is, for all

$$\varepsilon > 0$$

$$P[R(f_n) - R(f_{\Im}) > \varepsilon] \to 0$$

as

$$n \to \infty$$

3. Bayes' Consistent: The learning algorithm is called Bayes' consistent with respect to $P(x, y)$ if the risk $R(f_n)$ converges to $R(f_{Bayes})$ of the Bayes' classifier, that is, for all

$$\varepsilon > 0$$

$$P[R(f_n) - R(f_{Bayes}) > \varepsilon] \to 0$$

as

$$n \to \infty$$

4. Universal Consistency: The learning algorithm is called universally consistent with respect to $\Im$ (respectively universally Bayes' consistent) if it is consistent with respect to $\Im$ (respectively Bayes' consistent) for all probability distributions $P(x, y)$.
5. Convergence vs Weak Consistency: The consistency as stated above is called *weak consistency* in probability, as it is a statement about the consistency in probability. The analogous statement for convergence *almost surely* (i.e., in the ∞ sample size limit) would be called *strong consistency* (Devroye, Gyorfi, and Lugosi (1996)).

Bias-Variance or Estimation-Approximation Trade-off

1. Definition: Bias Variance Dilemma (or trade off) refers to the problem of simultaneously minimizing the bias (i.e., how accurate the model is across different training sets) and the variance of the model error (how sensitive is the model to small changes in the training set) (Bias-Variance Dilemma (Wiki)).
2. Applicability Suite: This trade off applies to all forms of supervised learning – classification, function fitting (Geman, Bienenstock, and Doursat (1992)), and structured output learning.
3. Motivation: Ideally one wants to choose a model that captures the irregularities in the training data, which at the same time generalizes well to unseen data.
4. High Bias Models: High-bias models are intuitively simple models, and imposed restrictions on the kinds of irregularities that can be learned (examples include linear classifiers). Problem is that they under-fit, i.e., they do not learn the relationship between the predicted (i.e., target) variables and the features.
5. High Variance Models: These can learn many kinds of complex irregularities, which unfortunately includes the noise in the training data as well (i.e., over-fitting).
6. Origin of the Tradeoff: A common model selection criterion is that decrease of bias with an increase in model complexity results in increase of variance. Other considerations such as error losses and complexity costs lead to alternate tradeoff criteria. The choice of model may also introduce biases that correspond to useful previous information, e.g., the output may need to be range bound.

Bias Variance Decomposition

1. Error Decomposition:

$$R(f_n) - R\left(f_{Bayes}\right) = \left[R(f_n) - R(f_{\Im})\right] + \left[R(f_{\Im}) - R\left(f_{Bayes}\right)\right]$$

$[R(f_n) - R(f_{\Im})]$ is called the Variance or the Estimation Error. Traditionally this has occupied much of the focus of SLT. $\left[R(f_{\Im}) - R\left(f_{Bayes}\right)\right]$ is called the Bias or the Approximation Error. This does not depend on the sample size, but it does depend on the underlying probability distribution $P(x, y)$.

2. Setup: We employ the terminology of Vijayakumar (2007). Let the data specified

$$D = \{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)\}$$

be derived from the true model

$$y = f(x) + \epsilon$$

where ϵ is a random error with

$$\mathbb{E}(\epsilon) = 0$$

Using this we train a model $g(x|D)$ to approximate f. We also set

$$g(x_i) = g_i$$

and

$$f(x_i) = f_i$$

3. Mean Squared Error:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}[y_i - g(x_i|D)]^2$$

The quantity of interest is the expectation of the MSE across the different realizations of the data:

$$\mathbb{E}[MSE] = \frac{1}{N}\sum_{i=1}^{N}\mathbb{E}[y_i - g(x_i|D)]^2$$

4. MSE Expansion: Re-writing $\frac{1}{N}\sum_{i=1}^{N}[y_i - g_i]^2$ as $\frac{1}{N}\sum_{i=1}^{N}[y_i - f_i + f_i - g_i]^2$, we recognize that $y_i - f_i$ is the noise term ϵ above, and $f_i - g_i$ is simply the proxying error. Thus

$$\begin{aligned}\mathbb{E}[y_i - g_i]^2 &= \mathbb{E}[y_i - f_i]^2 + \mathbb{E}[f_i - g_i]^2 + 2\mathbb{E}\{[y_i - f_i][f_i - g_i]\} \\ &= \mathbb{E}[\epsilon^2] + \mathbb{E}[f_i - g_i]^2 + 2\{\mathbb{E}[y_i f_i] - \mathbb{E}[f_i^2] - \mathbb{E}[y_i g_i] + \mathbb{E}[f_i g_i]\}\end{aligned}$$

Given that f is deterministic

$$\mathbb{E}[y_i f_i] - \mathbb{E}[f_i^2] = 0$$

Likewise

$$\mathbb{E}[y_i g_i] = \mathbb{E}[g_i(f_i + \varepsilon)] = \mathbb{E}[f_i g_i] + \mathbb{E}[\varepsilon g_i] = \mathbb{E}[f_i g_i]$$

if we assume that

$$\mathbb{E}[\varepsilon g_i] = 0$$

Thus, the last term also vanishes. Therefore

$$\mathbb{E}[y_i - g_i]^2 = \mathbb{E}[\epsilon^2] + \mathbb{E}[f_i - g_i]^2$$

5. MSE Reduction:

$$\begin{aligned}\mathbb{E}[f_i - g_i]^2 &= \mathbb{E}[f_i - \mathbb{E}[g_i] + \mathbb{E}[g_i] - g_i]^2 \\ &= \mathbb{E}[f_i - E[g_i]]^2 + \mathbb{E}[\mathbb{E}[g_i] - g_i]^2 + 2\{\mathbb{E}([f_i - \mathbb{E}[g_i]][\mathbb{E}[g_i] - g_i])\}\end{aligned}$$

Here

$$\{\mathbb{E}([f_i - \mathbb{E}[g_i]][\mathbb{E}[g_i] - g_i])\} = \mathbb{E}\{f_i\mathbb{E}[g_i]\} - \mathbb{E}\{\mathbb{E}[g_i]\mathbb{E}[g_i]\} - \mathbb{E}\{f_i g_i\} + \mathbb{E}\{g_i\mathbb{E}[g_i]\}$$
$$= f_i\mathbb{E}[g_i] - \{\mathbb{E}[g_i]\}^2 - f_i\mathbb{E}[g_i] + \{\mathbb{E}[g_i]\}^2 = 0$$

Thus

$$\mathbb{E}[f_i - g_i]^2 = \mathbb{E}\left[f_i - \mathbb{E}[g_i]\right]^2 + \mathbb{E}[\mathbb{E}[g_i] - g_i]^2$$

6. MSE Final Form:

$$\mathbb{E}[y_i - g_i]^2 = \mathbb{E}[\epsilon^2] + \mathbb{E}\left[f_i - \mathbb{E}[g_i]\right]^2 + \mathbb{E}[\mathbb{E}[g_i] - g_i]^2$$

Here $\mathbb{E}[\epsilon^2]$ is the irreducible sample error (assuming infinite sample), $\mathbb{E}\left[f_i - \mathbb{E}[g_i]\right]^2$ is the square of the bias term, and $\mathbb{E}[\mathbb{E}[g_i] - g_i]^2$ is the model variance term. Cast in another manner

$$\mathbb{E}[y_i - g_i]^2 = Mean\ Squared\ Error = Irreducible\ Error + Bias^2 + Model\ Variance$$

Bias Variance Optimization

1. Impact of Feature Addition: Feature selection/filtering and dimensionality reduction can decrease variance by simplifying the models. Adding features (predictors) tends to decrease bias at the expense of introducing extra variance.
2. Approaches for Bias Variance Tuning: Learning algorithms typically have some tunable parameters that control bias and variance. For example:
    a. Generalized linear models can be regularized to increase their bias.
    b. In neural nets, deeper models with more layers will have stronger variance, this regularization (as in GLM) is applied.
    c. In k-nearest neighbor models, a high value of k leads to low variance.

d. In Instance based learning, regularization can be achieved by varying the mixture of prototypes and examplars (Gagliardi (2011)).
e. In decision trees, the depth of the tree determines the variance. Decision trees are commonly pruned to control the variance (James, Witten, Hastie, and Tibshirani (2013)).

3. Mixture Models and Ensemble Learning: These provide another way to address the bias-variance tradeoff (Ting, Vijayakumar, and Schaal (2011), Fortmann-Roe (2012)). For example, as seen earlier, boosting combines many "weak" (high bias) models in an ensemble that has greater variance than the individual models, while bagging combines strong learners in a way that reduces their variance.

Generalization and Consistency for kNN

1. Setup: Assume that there exists a distance metric function

$$d: \chi \times \chi \to \mathbb{R}$$

that assigns a distance value $d(X, X')$ to each pair of training points (X, X'). Then $NN(X)$, the nearest neighbor to X among all the training points is

$$NN(X) = \underset{X' \in \{X_1, \ldots, X_n\}}{\arg min} \{d(X, X') \leq d(X, X'') \vee X'' \in \{X_1, \ldots, X_n\}$$

(Stone (1977), Scholkopf and Smola (2002)). The corresponding classifier f_n is

$$f_n(X) = Y_i$$

where

$$X_i = NN(X)$$

2. kNN Bayes' Risk vs. f_n Risk: It is easy to construct instances to see where the Bayes' risk for 1NN is quite different from $R_{1NN}(f_n)$. However, by expanding the nearest neighbor out to kNN, we can see the difference between $R_{kNN}(f_n)$ and $R_{kNN}(f_{Bayes})$ decreases, thereby improving the consistency.
3. Stone (1977) kNN Consistency Theorem: Let f_n be the kNN classifier constructed on n sample points. If

$$n \to \infty$$

and

$$k \to \infty$$

such that

$$\frac{k}{n} \to 0$$

then

$$R_{kNN}(f_n) \to R_{kNN}(f_{Bayes})$$

for all probability distributions $P(x, y)$. Therefore, kNN classification thus constructed can be made universally consistent.

   a. Interpretation => Essentially, one has to allow the sample size k of the neighborhood under consideration to grow with the sample size n in a controlled way. For e.g., if the parameter k grows slowly with n, say

$$k \approx \log n$$

then kNN becomes universally Bayes' consistent.

4. Intuition behind the kNN Function Complexity: Treating k as the hypothesis variant dimension in the space of functions $\Im_{kNN}$, the extremes to observe are: a) 1NN, where potentially each node can switch its ± 1 label depending upon its neighbors, and b) kNN with

$$k = n$$

the sample size, which has only one label. In this situation, for a location independent $P(x, y)$

$$R(f_{Bayes}) = R(f_{kNN})$$

(Devroye, Gyorfi, and Lugosi (1996)).

References

- Bias-variance dilemma (Wiki): *Wikipedia Entry for Bias-variance dilemma*.
- Devroye, L., L. Gyorfi, and G. Lugosi (1996): *A Probabilistic Theory of Pattern Recognition* **Springer** New York.
- Fortmann-Roe, S. (2012): *Understanding the Bias-Variance Tradeoff*
- Gagliardi, F. (2011): Instance-based Classifiers applied to Medical Databases *Artificial Intelligence in Medicine* **52 (3)** 123-139.
- Geman, S., E. Bienenstock, and R. Doursat (1992): Neural Networks and the Bias/Variance Dilemma *Neural Computation* **4** 1-58.
- James, G., D. Witten, T. Hastie, and R. Tibshirani (2013): *An Introduction to Statistical Learning*
- Scholkopf, B. and A. Smola (2002): *Learning with Kernels* **MIT Press** Cambridge, MA.
- Stone, C. J. (1977): Consistent non-parametric regression (with discussion) *Annals of Statistics* **5** 595-645.

- Ting, J. A., S. Vijayakumar, and S. Schaal (2011): Locally Weighted Regression for Control in *Encyclopedia of Machine Learning* (eds. C. Sammut and G. I. Webb) **Springer** 615.
- Vijayakumar, S. (2007): *The Bias-Variance Tradeoff*.
- von Luxburg, U., and B. Scholkopf (2005): Statistical Learning Theory: Models, Concepts, and Results, in *Handbook of the History of Logic* (editors: D. Gabbay, S. Hartmann, and J. Woods).

Empirical Risk Minimization

ERM Literature and Introduction

1. Empirical Process Theory - Literature: Uniform deviations of averages form their expectations is one of the central problems of empirical process theory, and comprehensive coverages include, Gine (1996), van der Waart and Wellner (1996), Vapnik (1998), and Dudley (1999).
2. Empirical Processes in Classification: The use of empirical processes in classification was pioneered by Vapnik and Chervonenkis (1971, 1974) and re-discovered 20 years later by Blumer, Ehrenfeucht, Haussler, Warmuth (1989), and Ehrenfeucht, Haussler, Kearns, and Valiant (1989). For surveys, see Natarajan (1991), Kearns and Vazirani (1994), Vapnik (1995), Devroye, Gyorfi, and Lugosi (1996), Vapnik (1998), and Anthony and Bartlett (1999).
3. ERM Induction Principle: The proxying of the risk corresponding to the full population to the empirical risk of the sample, choosing that particular

$$f_n \in \Im$$

that minimizes the empirical risk of the given finite sample is called the ERM induction principle

$$f_n := \underset{f \in \Im}{\arg min}\, R_{emp}(f)$$

4. Principle Caveats: With every sample set change, f_n will vary. Ideally, f_{Bayes} should be part of $\Im$, so that as

$$n \to \infty$$

$$f_n \to f_{Bayes}$$

Otherwise

$$f_n \to f_{\Im}$$

but

$$f_{\Im} \neq f_{Bayes}$$

even as

$$n \to \infty$$

this produces sizeable estimation error $[R(f_n) - R(f_{\Im})]$ as well as approximation error $\left[R(f_{\Im}) - R\left(f_{Bayes}\right)\right]$.

Overview

1. Definition: ERM is a principle in statistical learning which defines a family of learning algorithms constructed using a specific risk principle, and is used to provide performance bounds on these algorithms.
2. Background/Setting: Consider the following situation typical of supervised learning problems. We have 2 spaces of objects X and Y and would like to learn a function (called a hypothesis)

$$h_i: X \to Y$$

which outputs an object

$$y \in Y$$

given

$$x \in X$$

We have a set of training examples $(x_1, y_1), \ldots, (x_N, y_N)$ where

$$x_i \in X$$

is the input and

$$y_i \in Y$$

is the corresponding response, and we wish to get the response predictor hypothesis $h(x_i)$.

3. Training Set Generation: Formally we assume that there is a joint probability distribution $P(x, y)$ over X and Y, and the training set consists of N instances drawn i.i.d. from $P(x, y)$. Note that the assumption of joint probability allows us to model uncertainties in the predictors (e.g., noise from data) because y is not a deterministic function of x, but rather a random variable with a conditional distribution $P(y|x)$ for a fixed x.
4. ERM Loss Function: We also assume that we are given a non-negative real-valued loss-function $L(\hat{y}, y)$ that quantifies how different the prediction $\hat{y}$ of the hypothesis is from the true outcome y. The risk associated with the hypothesis is then defined as the expectation of the loss function:

$$R(h) = E\{L[h(x), y]\} = \int L[h(x), y]\, dP(x, y)$$

A commonly used loss function for classification is the $0 - 1$ loss function

$$L(\hat{y}, y) = I(\hat{y} \neq y)$$

where $I(...)$ is the indicator notation.

a. Expectation Maximization Insight => The least squares loss function corresponds to maximizing the joint normal probability $P(x, y)$. If $P(x, y)$ is not Gaussian, and/or if there is a non-uniform (i.e., Bayesian, not frequentist) prior, then the corresponding for the loss function may also be derived (it will not be Gaussian in general).

5. The Goal of the Computational Learning Algorithm: To find the hypothesis h^* among the fixed class of hypotheses H that minimizes the risk $R(h)$:

$$h^* = \underset{h \in H}{\arg min}\, R(h)$$

The Loss Function and the Empirical Risk Minimization Principles

1. Choice of Loss Function: The choice of loss function is critical in computing the hypothesis – this choice significantly impacts the algorithm convergence rate. Further it is also important for the loss function to be convex (Rosasco, Vito, Caponnetto, Piana, and Verri (2004)).
2. ERM Definition: In general, $R(h)$ cannot be computed since $P(x, y)$ is unknown to the learning algorithm (this situation is referred to as agnostic learning). Empirical risk is the approximation that results from averaging the loss function evenly across the training set (this simply corresponds to a uniform $P(x, y)$):

$$R_{Emp}(h) = \frac{1}{N}\sum_{i=1}^{N} L[h(x_i), y_i]$$

(See Empirical Risk Minimization (Wiki)).

3. Problem Statement: Empirical Risk Minimization Principle states that the learning algorithm should choose a hypothesis h^* that minimizes the empirical risk:

$$h^* = \underset{h \in H}{\arg min}\, R_{Emp}(h)$$

Depending upon the nature of the learner, this base formulation may end up as a combinatorial optimization problem for which globally optimal solutions are infeasible, thereby forcing a resort to meta-heuristic techniques.

Application of the Central Limit Theorem (CLT) and Law of Large Numbers (LLN)

1. CLT vs. LLN over Function Spaces: CLT and LLN are alternate views at the same fact; CLT looks at

$$n \to \infty$$

expected mean of the sample, whereas LLN looks at the probability of the sample exceeding the population mean. If CLT is computed over the hypotheses spaces, it is called $\mathcal{P}$-Donsker; LLN over hypothesis space is, likewise, called Glivenko-Cantelli. Thus, one implies the other.

2. Glivenko-Cantelli Literature: The question of how

$$\underset{f \in \Im}{sup}[P(f) - P_n(f)]$$

is also called the Glivenko-Cantelli, and is covered in Vapnik and Chervonenkis (1971), Dudley (1978), Vapnik and Chervonenkis (1981), Dudley (1984, 1987), Talagrand (1987, 1994), and Alon, Ben-David, Cesa-Bianchi, and Haussler (1997).

3. ERM LLN: LLN simply states that

$$\frac{1}{n}\sum_{i=1}^{n}\xi_i \to E(\xi)$$

as

$$n \to \infty$$

where ξ_i and ξ are drawn from the same distribution. LLN is valid under very mild conditions, and is easily extensible to ERM as follows:

$$\frac{1}{n}\sum_{i=1}^{n} l[X_i, Y_i, f(X_i, Y_i)] \to E\{l[X, Y, f(X, Y)]\}$$

as

$$n \to \infty$$

Again, (X_i, Y_i) is drawn from the same distribution that generates $f(X, Y)$.

4. Probability Bounds for LLN: A famous inequality by Chernoff (1952) expanded later to the i.i.d. observation set by Hoeffding (1963) characterizes how well the sample empirical mean approaches the population mean as a function of the sample size (the result is independent of $P(x, y)$, and is quiet conservative). Applied to ERM, it is:

$$P\left[\left|R_{Emp}(f) - R(f)\right| \geq \varepsilon\right] \leq 2e^{-2n\varepsilon^2}$$

5. Other Probability Bounds: Both Chernoff and Hoeffding bounds are derived from the Markov inequality. As such, they are conservative and provide only loose bounds, since they do not use $P(x, y)$ at all. Even among these agnostic bounds, moment bounds (and factorial moment bounds) can be much tighter than the Hoeffding bound (which does not use moments at all). Examples include the Bennett and the Bernstein bounds (naïve usage of

limited number of moments, like the Chebyshev bounds which uses the 2nd moment-based bounds, provide even looser convergence than Hoeffding).

Inconsistency of Empirical Risk Minimizers

1. Memorizing Classifiers: While memorizing classifiers can produce zero empirical risk, thereby being the "perfect" empirical risk minimizer, they would often need to "wiggle" enormously, thereby possessing a high "complexity" quotient i.e.,

$$VCdim \to \infty$$

   This makes them terrible learners.
2. Small Wavelength Trigonometric Functions: Is a small λ sine/cosine (or any periodic function) as example of a perfect memorizer? It appears so.

Uniform Convergence

1. Motivation: As it turns out, the conditions required to render ERM consistent involve restricting the set of admissible functions (thereby limiting complexity). The main insight of VC theory is that the consistency of ERM is determined by the worst-case behavior over all

$$f \in \Im$$

   that the learning machine could choose.
2. The ERM Approach: For each function

$$f \in \Im$$

   LLN tells us that as

$$n \to \infty$$

$$R_{Emp}(f) \to R(f)$$

However, this does not necessarily mean that the ER minimizer function f_n produces a risk that approaches that of $f_{\Im}$ as

$$n \to \infty$$

The latter is what results in statistical/functional class consistency. For this to be true, we explicitly require that

$$R_{Emp}(f) \to R(f)$$

as

$$n \to \infty$$

for all

$$f \in \Im$$

(Scholkopf and Smola (2002)).

3. Sample Size producing Uniform Convergence: Of course, from LLN, there will exist some large enough n such that

$$\left|R_{Emp}(f) - R(f)\right| \geq \varepsilon$$

for all

$$f \in \mathfrak{J}$$

- we write this as

$$\sup_{f \in \mathfrak{J}} \left| R_{Emp}(f) - R(f) \right| \geq \varepsilon$$

Ideally, we hope that this

$$n \to \infty$$

is not too large.

4. f_n ERM Consistency Workout: For the hypothesis class estimation error consistency, we require that $|R(f_n) - R(f_{\mathfrak{J}})|$ be bounded. Now

$$\begin{aligned} |R(f_n) - R(f_{\mathfrak{J}})| &= \left| R(f_n) - R_{Emp}(f_n) + R_{Emp}(f_n) - R_{Emp}(f_{\mathfrak{J}}) + R_{Emp}(f_{\mathfrak{J}}) - R(f_{\mathfrak{J}}) \right| \\ &\leq \left| R(f_n) - R_{Emp}(f_n) \right| + \left| R(f_{\mathfrak{J}}) - R_{Emp}(f_{\mathfrak{J}}) \right| + \left| R_{Emp}(f_n) - R_{Emp}(f_{\mathfrak{J}}) \right| \\ &\leq \left| R(f_n) - R_{Emp}(f_n) \right| + \left| R_{Emp}(f_n) - R_{Emp}(f_{\mathfrak{J}}) \right| \end{aligned}$$

ERM Complexity

1. ERM Complexity Analysis: ERM for a classification problem with a $0-1$ loss function is known to be NP-hard even for relatively simple class of functions such as linear classifiers (owing to the supra-exponential growth of the combinatorial optimization search space – Feldman, Guruswamy, Raghavendra, and Wu (2010)). However, when the minimal empirical risk is zero AND when the data is linearly separable, it can be solved efficiently.
2. ERM Complexity Handling Approaches: In practice, machine learning algorithms cope with NP-hard situations as above either by using a convex approximation to $0-1$ loss function (e.g., hinge loss for SVM – this alters the combinatorial search space onto a convex real

search space) that is easier to work with/optimize, or by posing simplifying assumptions on $P(x, y)$ (thereby not being agnostic anymore).

a. Loss Functions for Classification => The $0-1$ indicator loss function can be implemented via the Heaviside step function. To convert it to a convex function, a hinge loss function is often used:

$$L[h(x), y] = [-yh(x)]_+$$

References

- Alon, N, S. Ben-David, N. Cesa-Bianchi, and D. Haussler (1997): Scale-sensitive Dimensions, Uniform Convergence, and Learnability *Journal of the ACM* **44** 615-631.
- Anthony, M., and P. L. Bartlett (1999): *Neural Network Learning: Theoretical Foundations* **Cambridge University Press** Cambridge.
- Blumer, A., A. Ehrenfeucht, D. Haussler, and M. Warmuth (1989): Learnability and the Vapnik Chervonenkis Dimension *Journal of the ACM* **36** 929-965.
- Ehrenfeucht, A., D. Haussler, M. Kearns, and L. Valiant (1989): A General Lower Bound on the Number of Sample needed for Learning *Information and Computation* **82** 247-261.
- Empirical Risk Minimization (Wiki): *Wikipedia Entry for Empirical Risk Minimization*.
- Chernoff, H. (1952): A Measure of Asymptotic Efficiency of Tests of a Hypothesis based on the Sum of Observations *Annals of Mathematical Statistics* **23** 493-507.
- Devroye, L., L. Gyorfi, and G. Lugosi (1996): *A Probabilistic Theory of Pattern Recognition* **Springer** New York.
- Dudley, R. (1978): Central Limit Theorems for Empirical Measures *Annals of Probability* **6** 899-929.
- Dudley, R. (1984): Empirical Processes, in: *Ecole de Probabilite de St. Fleur 1982, Lecture Notes in Mathematics #1097* **Springer-Verlag** New York.
- Dudley, R. (1987): Universal Donsker Classes and Metric Entropy *Annals of Probability* **15** 1306-1326.

- Dudley, R. (1999): *Uniform Central Limit Theorems* **Cambridge University Press** Cambridge.
- Feldman, V., V. Guruswamy, P. Raghavendra, and Y. Wu (2010): *Agnostic Learning of Monomials by Half-spaces is Hard* **arXiV**.
- Gine, E. (1996): Empirical Processes and Applications: An Overview *Bernoulli* **2** 1-28.
- Hoeffding, W. (1963): Probability Inequalities for Sums of Bounded Random Variables *Journal of American Statistical Association* **58** 13-30.
- Kearns, M., and U. Vazirani (1994): *An Introduction to Computational Learning Theory* **MIT Press** Cambridge, MA.
- Natarajan, B. (1991): *Machine Learning: A Theoretical Approach* **Morgan Kauffman** San Mateo, CA.
- Rosasco, L., E. Vito, A. Caponnetto, M. Piana, and A. Verri (2004): Are All Loss Functions of Same? *Neural Computation* **5 (16)** 1063-1076.
- Scholkopf, B. and A. Smola (2002): *Learning with Kernels* **MIT Press** Cambridge, MA.
- Talagrand, M. (1987): The Glivenko-Cantelli Problem *Annals of Probability* **15** 837-870.
- Talagrand, M. (1994): Sharper Bounds for Gaussian and Empirical Processes *Annals of Probability* **22** 28-76.
- Van der Waart, A. and J. Wellner (1996): *Weak Convergence and Empirical Processes* **Springer Verlag** New York.
- Vapnik, V., and A. Chervonenkis (1971): On the Uniform Convergence of Relative Frequencies of Events to their Probabilities *Theory of Probability and its Applications* **16** 264-280.
- Vapnik, V., and A. Chervonenkis (1974): *Theory of Pattern Recognition (in Russian)* **Nauka** Moscow.
- Vapnik, V., and A. Chervonenkis (1981): The Necessary and Sufficient Conditions for the Uniform Convergence of Averages to their Expected Values *Teoriya Veroyatnostei i Ee Primeneniya* **26 (3)** 543-564.
- Vapnik, V. (1995): *The Nature of Statistical Learning Theory* **Springer Verlag** New York.
- Vapnik, V. (1998): *Statistical Learning Theory* **Wiley** New York.

Symmetrization

Introduction

1. Motivation: The purpose of the Symmetrization step is to replace $R(f)$ in

$$\sup_{f \in \Im} \left| R_{Emp}(f) - R(f) \right| \geq \varepsilon$$

(since $R(f)$ is not an observable) by $R_{Emp}(f)$, which is an observable. To this end, we introduce a new copy $(X_i', Y_i')_{i=1,\ldots,n}$ of our original sample, and refer to this as our ghost sample. Thus, all expectations for $R(f)$ now occur over (X_i', Y_i').

2. Symmetrization is Very Generic: Since Symmetrization converts all the expectation calculations to happen over the ghost sample, it may be very effectively applied across all operators that perform expectations, e.g., moment bounds, Chernoff/Chebyshev bounds etc.

3. Vapnik Chervonenkis Symmetrization Lemma: For

$$m\varepsilon^2 \geq 2$$

we have

$$\mathbb{P}\left[\sup_{f \in \Im} \left| R(f) - R_{Emp}(f) \right| \geq \varepsilon\right] \leq 2\mathbb{P}\left[\sup_{f \in \Im} \left| R'_{Emp}(f) - R_{Emp}(f) \right| \geq \frac{\varepsilon}{2}\right]$$

The $\mathbb{P}$ on the LHS refers to the distribution of an i.i.d. sample, while the second one refers to the distribution of 2 samples of size n each – the original and the ghost samples – thereby an i.i.d. distribution of size $2n$.

4. Transformation onto the Outcome Space: Using the lemma, we can now convert the probability bounds estimation in the infinite sample case, i.e., $R(f)$, to samples over the dual

sequence of n and n', i.e., $2n$ sample outcomes. Remember that the set (n, n') can only assume a finite number of outcomes, i.e., in the case of binary classifiers, at most

$$2^{n+n'} = 2^{2n}$$

outcomes.

5. The Continuous Case: Since we are evaluating functions only on their outcomes, we treat all functions that produce the same outcome identically. Thus, even in the infinite function space case, as long as the space only results in a fixed output tuple, that's all we use to evaluate. For a binary classifier this reduces to 2^{2n} cases.

Proof of the Symmetrization Lemma

1. Literature: Treatment in detail of the Symmetrization lemma may be found in Vapnik and Chervonenkis (1971, 1974) and Gine and Zinn (1984).
2. Symmetrization Lemma - Statement: For any

$$\varepsilon > 0$$

such that

$$n\varepsilon^2 \geq 2$$

$$\mathbb{P}\left[\sup_{f \in \Im} (P - P_n)f \geq \varepsilon\right] \leq 2\mathbb{P}\left[\sup_{f \in \Im} (P'_n - P_n)f \geq \frac{\varepsilon}{2}\right]$$

where P'_n is the corresponding empirical measure taken over an independent ghost sample $z'_1, \ldots, z'_n$.

3. #1 - Applying Indicator Functions over the Split Subsamples: Let f_n be the function that achieves the supremum (note that it depends on $z_1, \ldots, z_n$). Using $\wedge$ to represent the conjunction of the 2 events we get

$$1_{(P-P_n)f_n>\varepsilon} 1_{(P-P'_n)f_n<\frac{\varepsilon}{2}} = 1_{(P-P_n)f_n>\varepsilon \wedge (P'_n-P)f_n>-\frac{\varepsilon}{2}} \leq 1_{(P'_n-P_n)f_n<\frac{\varepsilon}{2}}$$

4. #2 - Expectation over the Ghost Sample: Taking expectations over the ghost samples $\mathbb{P}'$ gives

$$1_{(P-P_n)f_n>\varepsilon} P'\left[(P-P'_n)f_n < \frac{\varepsilon}{2}\right] \leq P'\left[(P'_n-P_n)f_n > \frac{\varepsilon}{2}\right]$$

5. #3 - Applying Chebyshev Inequality to the Ghost Sample: By Chebyshev's inequality

$$P'\left[(P-P'_n)f_n \geq \frac{\varepsilon}{2}\right] \leq \frac{4Var(f_n)}{n\varepsilon^2} \leq \frac{1}{n\varepsilon^2}$$

since the random variable in the range $[0, 1]$ has a variance $\leq \frac{1}{4}$. Thus, switching from $\geq \frac{\varepsilon}{2}$ to $< \frac{\varepsilon}{2}$

$$1_{(P-P_n)f_n>\varepsilon}\left[1-\frac{1}{n\varepsilon^2}\right] \leq P'\left[(P'_n-P_n)f_n > \frac{\varepsilon}{2}\right]$$

6. #4 Expectation over the Original Sample: Taking expectation with respect to the original sample ($\mathbb{P}$), and observing that

$$n\varepsilon^2 \geq 2$$

we recover the statement we set out to prove, i.e., the Symmetrization lemma.

7. Applying Hoeffding's Union Bound:

$$\mathbb{P}\left[\sup_{f \in \Im} (P - P_n)f \geq \varepsilon\right] \leq 2\mathbb{P}\left[\sup_{f \in \Im} (P'_n - P_n)f \geq \frac{\varepsilon}{2}\right]$$
$$= \mathbb{P}\left[\sup_{f \in \Im_{Z_1,\ldots,Z_n,Z'_1,\ldots,Z'_n}} (P'_n - P_n)f \geq \frac{\varepsilon}{2}\right]$$
$$\leq 2S_{\Im}(2n)\mathbb{P}\left[(P'_n - P_n)f \geq \frac{\varepsilon}{2}\right] \leq 4S_{\Im}(2n)e^{-\frac{n\varepsilon^2}{8}}$$

Inversion of this proves the VC theorem for risk bound using the growth function; For

$$n\varepsilon^2 \geq 2$$

$$\delta > 0$$

and, with probability at least $1 - \delta$

$$R(f) - R_n(f) \leq 2\sqrt{2\frac{\log S_{\Im}(2n) + \log\frac{2}{\delta}}{n}}$$

References

- Gine, E., and J. Zinn (1984): Some Limit Theorems for Empirical Processes *Annals of Probability* **12** 929-989.
- Vapnik, V., and A. Chervonenkis (1971): On the Uniform Convergence of Relative Frequencies of Events to their Probabilities *Theory of Probability and its Applications* **16** 264-280.
- Vapnik, V., and A. Chervonenkis (1974): *Theory of Pattern Recognition (in Russian)* **Nauka** Moscow.

Generalization Bounds

Union Bound

1. Motivation: Thus far, all we have are bounds for a single function (across the sample point space), "in the limit". We seek bounds that work across the entire function space for finite sample sizes, i.e., we seek

$$Prob\left[\sup_{f \in \Im} \left|R_{Emp}(f) - R(f)\right| \geq \varepsilon\right]$$

2. Setup: Given a finite sample error probability for a single function, union bound seeks to find the probability that any one function in the function space exceeds the specified error bound. This is necessary to compute a conservative lower bound for the empirical case.
3. Formulation: We work out the discrete hypothesis set case:

$$Prob\left[\sup_{f \in \Im} \left|R_{Emp}(f) - R(f)\right| \geq \varepsilon\right] \leq \sum_{i=1}^{m} Prob\left[\left|R_{Emp}(f_i) - R(f_i)\right| \geq \varepsilon\right] \leq 2me^{-2n\varepsilon^2}$$

   where m is the number of (discrete) functions in the function space $\Im$, and n is the number of data points. Note that we seek to optimize the modulus $\left|R_{Emp}(f_i) - R(f_i)\right|$.
4. Hoeffding for the Function Space: We start from

$$Prob(|S_n - E[S_n]| \geq t) \leq 2e^{-\frac{2t^2}{\sum_{i=1}^{n}(b_i - a_i)^2}}$$

   Note that

$$S_n \to R_{Emp}(f)$$

and

$$E[S_n] \to R(f)$$

Further

$$t \to n\varepsilon$$

and

$$b_i - a_i \to 1$$

Therefore

$$\sum_{i=1}^{n} (b_i - a_i)^2 = n^2$$

Thus

$$Prob\left[\left|R_{Emp}(f_i) - R(f_i)\right| \geq \varepsilon\right] \leq 2e^{-2n\varepsilon^2}$$

5. Discrete Union Bound - Convergence: Obviously, as

$$n \to \infty$$

$$Prob\left[\sup_{f \in \Im} \left|R_{Emp}(f) - R(f)\right| \geq \varepsilon\right] \to 0$$

This ensures statistical sample consistency. Extension to the continuous function space case requires a few more techniques, including replacing m with a more general capacity measure.

6. Refined Union Bound and the Countable Case #1: For each

$$f \in \mathfrak{I}$$

for each

$$\delta > 0$$

(possibly depending on f, which we write as $\delta(f)$), Hoeffding's inequality says that

$$\mathbb{P}\left[Pf - P_n f > \sqrt{\frac{\log\frac{1}{\delta(f)}}{2n}}\right]$$

Hence, if we have a countable set $\mathfrak{I}$, the union bound immediately yields

$$\mathbb{P}\left[\ni f \in \mathfrak{I}\colon Pf - P_n f > \sqrt{\frac{\log\frac{1}{\delta(f)}}{2n}}\right] \leq \sum_{f\in\mathfrak{I}} \delta(f)$$

7. Countably Infinite Case: Choosing

$$\delta(f) = \delta p(f)$$

$$\sum_{f\in\mathfrak{I}} p(f) = 1$$

this makes the RHS above equal to δ, and we get the following result; with probability at least $1 - \delta$

$$\forall f \in \Im: Pf - P_n f < \sqrt{\frac{\log \frac{1}{p(f)} + \log \frac{1}{\delta}}{2n}}$$

We notice that if $\Im$ is finite in N, taking a uniform p results in $\log N$ as before.

8. ERM Consistency Bounds: Applying the bounds separately to each counterpart above, we see that

$$|R(f_n) - R(f_\Im)| \leq 2 \sup_{f \in \Im} \left| R_{Emp}(f) - R(f) \right|$$

Expressing this in terms of probabilities using LLN produces the following:

$$P[|R(f_n) - R(f_\Im)| \geq \varepsilon] \leq P\left[\sup_{f \in \Im} \left| R_{Emp}(f) - R(f) \right| \geq \frac{\varepsilon}{2}\right]$$

9. Necessary and Sufficient Condition for ERM Convergence: Vapnik and Chervonenkis (1971), Devroye, Gyorfi, and Lugosi (1996), and Mendelson (2003) contain the details.

$$P\left[\left|R(f) - R_{Emp}(f)\right| \geq \varepsilon\right] \to 0$$

as

$$n \to \infty$$

for all

$$\varepsilon > 0$$

is a necessary and sufficient condition for the consistency of ERM with respect to $\Im$.

Shattering Coefficient

1. Motivation: The infinite function space case may now be reduced to

$$N \leq 2^{2n}$$

   (over the original and the ghost) subset. Of course, in many classifiers, the full spectrum 2^{2n} will never be produced, so N is used as the corresponding outcome "capacity measure", i.e., the outcome spectrum of the hypothesis set.
2. Definition: Let

$$\mathbb{Z}_n \coloneqq [(X_1, Y_1), \ldots, (X_n, Y_n)]$$

   be a given sample set. Denote by $\left|\mathfrak{I}_{\mathbb{Z}_n}\right|$ the cardinality of $\mathfrak{I}$ when restricted to $\{x_1, \ldots, x_n\}$, i.e., the number of f in $\mathfrak{I}$ that can be distinguished by their values on $\{x_1, \ldots, x_n\}$. The shattering coefficient $\mathcal{N}(\mathfrak{I}, n)$ is defined as

$$\mathcal{N}(\mathfrak{I}, n) = max\left\{\left|\mathfrak{I}_{\mathbb{Z}_n}\right| : x_i \in \{x_1, \ldots, x_n\}\right\}$$

3. Intuition: Shattering coefficient is simply a count of the number of ways the function space can classify the input pattern set. When

$$\mathcal{N}(\mathfrak{I}, n) = 2^k$$

   this means that there a sample of size k on which all possible separations can be achieved. Then $\mathfrak{I}$ is said to shatter $\{x_1, \ldots, x_n\}$ into k points.
4. Continuous $\mathfrak{I}$ Supremum Bound: It is easy to see that

$$\mathbb{P}\left[\sup_{f \in \mathfrak{I}} \left|R(f) - R_{Emp}(f)\right| \geq \varepsilon\right] \leq 2\mathcal{N}(\mathfrak{I}, 2n)e^{-\frac{n\varepsilon^2}{4}}$$

Convergence therefore depends on the growth of $2\mathcal{N}(\mathfrak{I}, 2n)$ with n.

5. Polynomial Growth of the Shattering Coefficient: If

$$\mathcal{N}(\mathfrak{I}, 2n) \leq (2n)^k$$

then

$$2\mathcal{N}(\mathfrak{I}, 2n)e^{-\frac{n\varepsilon^2}{4}} = 2e^{klog(2n)-\frac{n\varepsilon^2}{4}}$$

Thus

$$2\mathcal{N}(\mathfrak{I}, 2n)e^{-\frac{n\varepsilon^2}{4}} \to 0$$

as

$$n \to \infty$$

6. Shattering Coefficient growth as 2^{2n}: In this case

$$2\mathcal{N}(\mathfrak{I}, 2n)e^{-\frac{n\varepsilon^2}{4}} = 2e^{n\left[2log2-\frac{\varepsilon^2}{4}\right]}$$

thus this does not support convergence. However, since the supremum limits we've worked on so far are very conservative, this does not indicate non-conformance either.

7. Necessary and Sufficient Conditions for ERM Convergence: As shown in Vapnik and Chervonenkis (1971, 1981), Devroye, Gyorfi, and Lugosi (1996), and Mendelson (2003), the necessary and sufficient condition for EMR convergence is

$$\frac{\log \mathcal{N}(\mathfrak{I}, 2n)}{n} \to 0$$

Empirical Risk Generalization Bound

1. Probabilistic Bounds: Expressing the probability bound the other way, with a probability of at least $1 - \delta$

$$f \in \Im$$

satisfies

$$R(f) \leq R_{Emp}(f) + \sqrt{\frac{4}{n}[\log \mathcal{N}(\Im, 2n) - \log \delta]}$$

For polynomial growth functions

$$\sqrt{\frac{\log \mathcal{N}(\Im, 2n)}{n}} \to 0$$

as

$$n \to \infty$$

therefore it converges. However, for

$$\mathcal{N}(\Im, 2n) \sim 2^{2n}$$

$$\sqrt{\frac{\log \mathcal{N}(\Im, 2n)}{n}} \to 2$$

therefore that DOES NOT converge.

2. Union Bound Contribution - Inversion: To recap, for a specific f, with probability at least $1 - \delta$

$$R(f) \leq R_n(f) + \sqrt{\frac{\log \frac{2}{\delta}}{2n}}$$

For all

$$f \in \Im$$

with probability of at least $1 - \delta$

$$R(f) \leq R_n(f) + \sqrt{\frac{\log N + \log \frac{2}{\delta}}{2n}}$$

The extra $\log N$ term contribution from the union bound may also be interpreted as the number of additional bits needed to encode the hypothesis space $\Im$. It turns out that this kind of coding interpretation of the generalization bounds is often possible, and can be used to obtain error estimates (von Luxburg, Bousquet, and Scholkopf (2004)).

3. Alternate Generalization Bounds: Several other types of bounds for the probability and ERM exist, and they differ in the constants (in front of the exponential and as its argument), as exponent of ε, and in the way they measure capacity (Devroye, Gyorfi, and Lugosi (1996), Vapnik (1998)).
4. Improvements over Classical Bounds: There are several things that can be improved:
    a. Hoeffding's inequality only uses the boundedness of the functions, not their variance.
    b. The union bound is as bad as if all the functions in the class were independent (i.e., if $f_1(\mathbb{Z})$ and $f_2(\mathbb{Z})$ were independent).
    c. The supremum over $\Im$ of $R(f) - R_n(f)$ is not necessarily what the algorithm would choose, so that upper bounding $R(f_n) - R_n(f_n)$ by the supremum might be loose.

Large Margin Bounds

1. Motivation: Large Margin Bounds are a specialized capacity measure of the function classes, where the sole purpose is to classify points in a 2D $\mathbb{R}^2$ data space into separate classes using a straight line. A generalization would be linear classifiers in $\mathbb{R}^d$.
2. Setup: Given a set of training points and a classifier f_n that would perfectly separate them, we define the *Margin of the Classifier* as the smallest distance of any training point to the separating line f_n. A similar margin can be defined in $\mathbb{R}^d$ for an arbitrary dimension d.
3. VC Dimension: The VC dimension of a class $\Im_\rho$ of linear classifiers with all having a margin of at least ρ can be essentially bounded by the ratio of the radius R of the smallest sphere enclosing the data points with the margin ρ, that is

$$\mathcal{VC}(\Im_\rho) \leq min\left\{d, \frac{4R^2}{\rho^2}\right\} + 1$$

   (Vapnik (1995)).
4. Capacity: Thus, larger the ρ, smaller is the VC dimension. Thus, one can use the margin of the classifiers as a capacity concept. SVM builds on this concept, and is one of the best-known classifiers (Scholkopf and Smola (2002)).
5. $\mathbb{R}^d$ Definition: Assume that the data lies in a ball of radius R in $\mathbb{R}^d$. Consider the set $\Im_\rho$ of linear classifiers with a margin of at least ρ. Assume that we are given n training examples. Use $\nu(f)$ to denote the fraction of the training examples with margin smaller than ρ, or which are wrongly classified by a classifier

$$f \in \Im$$

   Then, with probability of at least $1 - \delta$, the true error of any

$$f \in \Im_\rho$$

can be bounded by

$$R(f) \leq \nu(f) + \sqrt{\frac{c}{n}\left[\frac{R^2}{\rho^2}(\log n)^2 + \log\frac{1}{\delta}\right]}$$

where c is a universal constant (Scholkopf and Smola (2002)).

References

- Devroye, L., L. Gyorfi, and G. Lugosi (1996): *A Probabilistic Theory of Pattern Recognition* **Springer** New York.
- Mandelson, S. (2003): A few Notes on Statistical Learning Theory *Advanced Lectures on Machine Learning* **LNCS** 1-40 **Springer**.
- Scholkopf, B. and A. Smola (2002): *Learning with Kernels* **MIT Press** Cambridge, MA.
- Vapnik, V., and A. Chervonenkis (1971): On the Uniform Convergence of Relative Frequencies of Events to their Probabilities *Theory of Probability and its Applications* **16** 264-280.
- Vapnik, V., and A. Chervonenkis (1981): The Necessary and Sufficient Conditions for the Uniform Convergence of Averages to their Expected Values *Teoriya Veroyatnostei i Ee Primeneniya* **26 (3)** 543-564.
- Vapnik, V. (1995): *The Nature of Statistical Learning Theory* **Springer Verlag** New York.
- Vapnik, V. (1998): *Statistical Learning Theory* **Wiley** New York.
- Von Luxburg, U., O. Bousquet, and B. Scholkopf (2004): A Compression Approach to support Vector Model Selection *Journal of Machine Learning Research* **5** 293-323.

Rademacher Complexity

Setup and Definition

1. Motivation: Rademacher complexity is an alternate metric of the function space capacity. Compared to shattering coefficient and the VC dimension, using the Rademacher complexity in conjunction with the underlying distribution can produce much tighter bounds. The mathematical treatment can also become simpler (Bousquet, Boucheron, and Lugosi (2003), Mendelson (2003), Boucheron, Bousquet, and Lugosi (2005)).
2. Problem Space: Given a space Z and a fixed distribution $D_{|Z}$, let

$$S = \{z_1, \dots, z_m\}$$

be a set of samples drawn from Z. Let $f \in \Im$ be a class of functions $f: Z \to \mathbb{R}$.

3. Rademacher Complexity - Definition: The *Empirical Rademacher Complexity* of $\Im$ is defined to be

$$\hat{\mathcal{R}}_m(\Im) = E_\sigma \left[\sup_{f \in \Im} \frac{1}{m} \sum_{i=1}^{m} \sigma_i f(z_i) \right]$$

where $\sigma_1, \dots, \sigma_m$ are independent random variables chosen from $\{-1, +1\}$. We refer to these as Rademacher variables. The *Rademacher Complexity* is defined as

$$\mathcal{R}_m(\Im) = E_D\left[\hat{\mathcal{R}}_m(\Im)\right]$$

4. Empirical Rademacher Complexity - Intuition: The sup intuitively measures, for a given set S and the Rademacher vector $\sigma_1, \dots, \sigma_m$, the maximum correlation between $f(z_i)$ and σ_i over

all $f \epsilon \Im$. The expectation over σ, i.e., the empirical Rademacher complexity of $\Im$ measures the ability of functions from $\Im$ (when applied to the fixed set S) to fit random noise.

5. Rademacher Complexity – Intuition: The Rademacher complexity of $\Im$ measures the expected noise-fitting-ability of $\Im$ over all the data sets

$$S \epsilon Z^m$$

that could be drawn according to the distribution $D_{|z}$.

6. Sample Space Generalization: The Rademacher complexity can be defined even more generally on sets

$$A \subseteq \mathbb{R}^m$$

by making the sup over

$$a \epsilon A$$

in place of

$$f \epsilon \Im$$

and replacing each $f(z_i)$ with a_i. Of course, setting

$$A \equiv \Im(S) = \{f(z) | \, f \epsilon \Im, z \epsilon S\}$$

recovers the earlier definitions.

Rademacher-based Uniform Convergence

1. The Concept: The intention is to bound each function that is part of any class if functions by their empirical averages, the Rademacher complexity of the class, and an error term that depends on the probabilistic confidence interval and the sample size.
2. Rademacher Bounding - Steps: The Rademacher bounding typically involves extracting sequential bounds, and it typically employs the following steps:
    a. Use *concentration* to relate

$$\sup_{f \in \mathcal{F}} (P - P_n) f$$

    to its expectation.
    b. Use *symmetrization* to relate the expectation to the Rademacher average.
    c. Use *concentration* again to relate the unconditional Rademacher average to its conditional one.
3. Symbology and Steps: Denote the empirical average over a sample S as

$$\widehat{\mathbb{E}}_s[f(z)] = \frac{1}{|S|} \sum_{z \in S} f(z)$$

Thus, for a function f, the definition of sup leads to

$$\mathbb{E}_D[f(z)] - \widehat{\mathbb{E}}_s[f(z)] \leq \sup_{h \in \mathfrak{H}} \left\{ \mathbb{E}_D[f(z)] - \widehat{\mathbb{E}}_s[f(z)] \right\}$$

We set

$$\varphi(S) = \sup_{h \in \mathfrak{H}} \left\{ \mathbb{E}_D[f(z)] - \widehat{\mathbb{E}}_s[f(z)] \right\}$$

and try to bound it by using the McDiarmid inequality.

    a. McDiarmid Bounded Differences Inequality: Let $x_1, \ldots, x_n$ be independent random variables taking on values in set A, and let $c_1, \ldots, c_n$ be positive real constants. If

$$\varphi: A^n \to \mathbb{R}$$

satisfies

$$\sup_{x_1, \ldots, x_n, x'_i \in \mathfrak{J}} |\varphi(x_1, \ldots, x_i, \ldots, x_n) - \varphi(x_1, \ldots, x'_i, \ldots, x_n)| \leq c_i$$

then for

$$1 \leq i \leq n$$

$$Prob\{\varphi(x_1, \ldots, x_i, \ldots, x_n) - \mathbb{E}[\varphi(x_1, \ldots, x_i, \ldots, x_n)] \geq \varepsilon\} \geq e^{-\frac{2\varepsilon^2}{\sum_{i=1}^{n} {c_i}^2}}$$

4. Supremum Bounds: $\varphi(S)$ as defined above satisfies

$$\sup_{z_1, \ldots, z_m, z'_i \in \mathfrak{J}} |\varphi(z_1, \ldots, z'_i, \ldots, z_m) - \varphi(z_1, \ldots, z'_i, \ldots, z_m)| \leq \frac{1}{m}$$

where m is the number of data points.

   a. Proof Setup => Let

$$S = \{z_1, \ldots, z_i, \ldots, z_m\}$$

and

$$S' = \{z_1, \ldots, z'_i, \ldots, z_m\}$$

We then define

$$|\varphi(S) - \varphi(S')| = \left| \sup_{h \in \mathfrak{J}} \{\mathbb{E}_D[f(z)] - \widehat{\mathbb{E}}_s[f(z)]\} - \sup_{h \in \mathfrak{J}} \{\mathbb{E}_D[f(z)] - \widehat{\mathbb{E}}_{s'}[f(z)]\} \right|$$

b. S Optimizer Function => Letting $h^* \in \mathfrak{I}$ be the maximizing function for the supremum in $\varphi(S)$

$$|\varphi(S) - \varphi(S')| \leq \left|\mathbb{E}_D[h^*(z)] - \hat{\mathbb{E}}_s[h^*(z)] - \sup_{h \in \mathfrak{I}}\{\mathbb{E}_D[f(z)] - \hat{\mathbb{E}}_{s'}[f(z)]\}\right|$$

c. S' Optimizer Function => By definition of the supremum, h^* can also, at best, maximize $\varphi(S')$, so

$$\mathbb{E}_D[h^*(z)] - \hat{\mathbb{E}}_{s'}[h^*(z)] \leq \sup_{h \in \mathfrak{I}}\{\mathbb{E}_D[f(z)] - \hat{\mathbb{E}}_{s'}[f(z)]\}$$

d. h Value Realization => Using the above 2 inequalities, and noting the negative sign ahead of $\varphi(S')$

$$\begin{aligned}|\varphi(S) - \varphi(S')| &\leq \left|\mathbb{E}_D[h^*(z)] - \hat{\mathbb{E}}_s[h^*(z)] - \mathbb{E}_D[h^*(z)] + \hat{\mathbb{E}}_{s'}[h^*(z)]\right| \\ &= \left|\hat{\mathbb{E}}_{s'}[h^*(z)] - \hat{\mathbb{E}}_s[h^*(z)]\right| = \frac{1}{m}\left|\sum_{z \in S} h^*(z) - \sum_{z \in S'} h^*(z)\right|\end{aligned}$$

e. Bounds Estimator => Since S and S' differ in only element j, the above becomes

$$|\varphi(S) - \varphi(S')| \leq \frac{1}{m}\left|h^*(z_j) - h^*(z'_j)\right| \leq \frac{1}{m}$$

where the last step follows from the fact that

$$h^*: \mathbb{Z} \to [a, a+1]$$

so

$$\sup_{z_j, z'_j \in \mathbb{Z}}\left|h^*(z_j) - h^*(z'_j)\right| = 1$$

5. McDiarmid Bounds on the Inequality $\varphi(S)$: Using the bound

$$|\varphi(S) - \varphi(S')| \leq \frac{1}{m}$$

we apply McDiarmid's inequality to get

$$\mathbb{P}\{\varphi(S) - \mathbb{E}[\varphi(S')] \geq t\} \geq e^{-\frac{t^2}{\sum_{i=1}^{m}\left(\frac{1}{m}\right)^2}} = e^{-mt^2}$$

Setting this probability to be less than δ and solving for t, we get

$$t \geq \sqrt{\frac{\log\frac{1}{\delta}}{m}}$$

6. First ERM Bounds: Thus, with a probability of at least $1 - \delta$,

$$\mathbb{E}_D[h(z)] - \widehat{\mathbb{E}}_s[h(z)] \leq \mathbb{E}_S\left\{\sup_{h \in \mathfrak{J}}\left[\mathbb{E}_D[h(z)] - \widehat{\mathbb{E}}_s[h(z)]\right]\right\} + \sqrt{\frac{\log\frac{1}{\delta}}{m}}$$

To obtain further refinement/tightness to these bounds, we introduce the concept of ghost variables, and then estimate the bounds in terms of Rademacher Complexity.

a. Impact of McDiarmid Bounds => Effectively, applying the McDiarmid bounds on each of the sample point transforms the agnostic supremum over to an expectation of the supremum over S (plus a bound that depends on the probability), i.e.

$$\mathbb{E}_D[h(z)] - \widehat{\mathbb{E}}_S[h(z)] \leq \mathbb{E}_S\left\{\sup_{h \in \mathfrak{J}}\left[\mathbb{E}_D[h(z)] - \widehat{\mathbb{E}}_s[h(z)]\right]\right\} \rightarrow \mathbb{E}_D[h(z)] - \widehat{\mathbb{E}}_s[h(z)]$$

$$\leq \mathbb{E}_S\left\{\sup_{h \in \mathfrak{J}}\left[\mathbb{E}_D[h(z)] - \widehat{\mathbb{E}}_s[h(z)]\right]\right\} + \sqrt{\frac{\log\frac{1}{\delta}}{m}}$$

7. Estimation of the Sample Bound for the Supremum: To estimate

$$\mathbb{E}_S\left\{\sup_{h \in \mathfrak{I}}\left[\mathbb{E}_D[h(z)] - \widehat{\mathbb{E}}_s[h(z)]\right]\right\}$$

we do the following steps:

a. Draw a "ghost sample"

$$\tilde{S} = \{\tilde{z}_1, \dots, \tilde{z}_m\}$$

that is i.i.d. and independent of

$$S = \{z_1, \dots, z_m\}$$

b. Get $\mathbb{E}_D[h(z)]$ as an expectation over these ghost samples.

c. Observe that

$$\mathbb{E}_{\tilde{S}}\left[\widehat{\mathbb{E}}_{\tilde{S}}[h(z)]|S\right] = \mathbb{E}_D[h(z)]$$

and that

$$\mathbb{E}_{\tilde{S}}\left[\widehat{\mathbb{E}}_S[h(z)]|S\right] = \widehat{\mathbb{E}}_S[h(z)]$$

8. Sample Expectation of the Supremum:

$$\mathbb{E}_S\left\{\sup_{h \in \mathfrak{I}}\left[\mathbb{E}_D[h(z)] - \widehat{\mathbb{E}}_s[h(z)]\right]\right\} = \mathbb{E}_S\left\{\sup_{h \in \mathfrak{I}}\mathbb{E}_{\tilde{S}}\left[\widehat{\mathbb{E}}_{\tilde{S}}[h(z)] - \widehat{\mathbb{E}}_S[h(z)]|S\right]\right\}$$

$$= \mathbb{E}_S\left\{\sup_{h \in \mathfrak{I}}\mathbb{E}_{\tilde{S}}\left[\frac{1}{m}\sum_{i=1}^{m}[h(\tilde{z}_i) - h(z_i)|S]\right]\right\}$$

9. Convexity of the Supremum Function: Since it seeks out the supremum, by definition *sup* is a convex function. Thus, we can apply Jensen's inequality to push the *sup* inside the expectation n as:

$$\mathbb{E}_S\left\{\sup_{h\in\Im}\mathbb{E}_{\tilde{S}}\left[\frac{1}{m}\sum_{i=1}^{m}[h(\tilde{z}_i)-h(z_i)|S]\right]\right\}\leq\mathbb{E}_{S,\tilde{S}}\left\{\sup_{h\in\Im}\left[\frac{1}{m}\sum_{i=1}^{m}[h(\tilde{z}_i)-h(z_i)|S]\right]\right\}$$

Remember that

$$\mathbb{E}_{S,\tilde{S}}[\dots]\to\mathbb{E}_S\left[\mathbb{E}_{\tilde{S}}[\dots]\right]$$

since $S,\tilde{S}$ are independent.

10. Advantage of the Rademacher Formulation: There are 2 main advantages to introducing Rademacher variables:
    a. Multiplying each term in the summation $[h(\tilde{z}_i)-h(z_i)|S]$ by a Rademacher variable σ_i independent of $S,\tilde{S}$ does not alter the expectation;
    b. Negating a Rademacher variable uniformly does not alter its distribution.
11. Reduction using Rademacher Variables:

$$\begin{aligned}\mathbb{E}_{S,\tilde{S}}\left\{\sup_{h\in\Im}\left[\frac{1}{m}\sum_{i=1}^{m}[h(\tilde{z}_i)-h(z_i)|S]\right]\right\}&=\mathbb{E}_{\sigma,S,\tilde{S}}\left\{\sup_{h\in\Im}\left[\frac{1}{m}\sum_{i=1}^{m}\sigma_i[h(\tilde{z}_i)-h(z_i)|S]\right]\right\}\\&\leq\mathbb{E}_{\sigma,S,\tilde{S}}\left\{\sup_{h\in\Im}\left[\frac{1}{m}\sum_{i=1}^{m}-\sigma_i h(z_i)+\frac{1}{m}\sum_{i=1}^{m}\sigma_i h(\tilde{z}_i)\right]\right\}\\&=2\mathbb{E}_{\sigma,S}\left\{\sup_{h\in\Im}\frac{1}{m}\sum_{i=1}^{m}\sigma_i h(z_i)\right\}=2\mathcal{R}_m(\Im)\end{aligned}$$

12. Rademacher Uniform Convergence Bound:

$$\mathbb{E}_D[h(z)]-\widehat{\mathbb{E}}_s[h(z)]\leq\mathbb{E}_S\left\{\sup_{h\in\Im}\left[\mathbb{E}_D[h(z)]-\widehat{\mathbb{E}}_s[h(z)]\right]\right\}+\sqrt{\frac{\log\frac{1}{\delta}}{m}}\leq 2\mathcal{R}_m(\Im)+\sqrt{\frac{\log\frac{1}{\delta}}{m}}$$

13. Empirical Rademacher Bounds Supremum Estimation: For

$$\hat{\mathcal{R}}_m(\Im) = \mathbb{E}_\sigma \left[\sup_{f \in \Im} \frac{1}{m} \sum_{i=1}^{m} \sigma_i f(z_i) \right]$$

it is bounded precisely the same way as before using the McDiarmid bounded difference inequality as

$$\left| \hat{\mathcal{R}}_{m,\mathbb{z}}(\Im) - \hat{\mathcal{R}}_{m,\mathbb{z}'}(\Im) \right| \leq \frac{1}{m}$$

This is easy to show, since all the arguments for the earlier bounding of the supremum apply in the presence of Rademacher variable coefficients as well.

14. Rademacher Average Uniform Convergence Bound #2: Thus, the second application of the McDiarmid' inequality (now using confidence limits of $\frac{\delta}{2}$ for each) bound $\hat{\mathcal{R}}_m(\Im)$ in terms of its expectation $\mathcal{R}_m(\Im)$ gives

$$\mathbb{E}_D[h(z)] - \hat{\mathbb{E}}_s[h(z)] \leq 2\hat{\mathcal{R}}_m(\Im) + 3\sqrt{\frac{\log\frac{2}{\delta}}{m}}$$

15. Bounding the Rademacher Complexity: To set up the use of Hoeffding's inequality, we start by taking exponential of the Empirical Rademacher complexity, multiply by some positive constant s, apply Jensen's inequality, and optimize over s. In other words, to compute

$$\mathbb{E}_\sigma \left\{ \sup_{a \in A} \frac{1}{m} \sum_{i=1}^{m} \sigma_i a_i \right\}$$

start with

$$s\mathbb{E}_\sigma\left\{\sup_{a \in A}\frac{1}{m}\sum_{i=1}^{m}\sigma_i a_i\right\}$$

16. Bounding Rademacher Complexity – Steps:

$$e^{s\mathbb{E}_\sigma\left\{\sup_{a\in A}\Sigma_{i=1}^{m}\sigma_i a_i\right\}} \le \mathbb{E}_\sigma\left[e^{s\left\{\sup_{a\in A}\Sigma_{i=1}^{m}\sigma_i a_i\right\}}\right] = \mathbb{E}_\sigma\left[\sup_{a \in A} e^{s\{\Sigma_{i=1}^{m}\sigma_i a_i\}}\right] \le \sum_{a\in A}\mathbb{E}_\sigma\left[e^{s\{\Sigma_{i=1}^{m}\sigma_i a_i\}}\right]$$

$$= \sum_{a\in A}\mathbb{E}_\sigma\left[\prod_{i=1}^{m} e^{s\sigma_i a_i}\right] = \sum_{a\in A}\prod_{i=1}^{m}\mathbb{E}_\sigma[e^{s\sigma_i a_i}]$$

17. Hoeffding's Application Criterion: The last step above exploits the fact that the σ_i's are independent. Now we can apply Hoeffding's inequality since

$$\mathbb{E}_\sigma[\sigma_i a_i] = 0$$

and

$$\sigma_i a_i \in [\alpha, \beta]$$

where

$$\beta - \alpha = 2a_i$$

18. Application of Hoeffding's Inequality:

$$e^{s\mathbb{E}_\sigma\left\{\sup_{a\in A}\Sigma_{i=1}^{m}\sigma_i a_i\right\}} \le \sum_{a\in A}\prod_{i=1}^{m}\mathbb{E}_\sigma[e^{s\sigma_i a_i}] \le \sum_{a\in A}\prod_{i=1}^{m} e^{\frac{s^2(2a_i)^2}{8}} = \sum_{a\in A} e^{\frac{s^2}{2}\Sigma_{i=1}^{m}{a_i}^2} \le |A| e^{\frac{s^2}{2}R^2}$$

where $|A|$ is the cardinality of the space A, and

$$R^2 = \sup_{a \in A} \sum_{i=1}^{m} {a_i}^2$$

19. Optimization around s: Taking log of both sides above we get

$$\mathbb{E}_\sigma \left\{ \sup_{a \in A} \frac{1}{m} \sum_{i=1}^{m} \sigma_i a_i \right\} \leq \frac{\log|A|}{s} + s\frac{R^2}{2}$$

which has a maximum at

$$s = \frac{\sqrt{2\log|A|}}{R}$$

20. Rademacher Complexity Bounds: Substituting this value for s back into the previous bound, and dividing both sides by m gives

$$\hat{\mathcal{R}}_m(\Im) = \mathbb{E}_\sigma \left[\sup_{a \in A} \frac{1}{m} \sum_{i=1}^{m} \sigma_i a_i \right] \leq \frac{R\sqrt{2\log|A|}}{m}$$

21. Application to VC Theory Finite Concept Class: For any finite concept class

$$\mathcal{H} \subseteq \{h: X \to [-1, +1]\}$$

and data points

$$S = \{x_1, \ldots, x_m\}$$

we can take

$$A = \{h(x_1), \ldots, h(x_m) | h \in \mathcal{H}\}$$

Thus

$$|A| \to |\mathcal{H}|$$

and

$$R = \sqrt{\sup_{a \in A} \sum_{i=1}^{m} a_i{}^2} = \sqrt{m}$$

on setting each

$$a_i = 1$$

Thus, in this case

$$\hat{\mathcal{R}}_m(\mathcal{H}) \leq \sqrt{\frac{2\log|\mathcal{H}|}{m}}$$

22. Application to the VC Theory - The General Case: In general, irrespective of whether $\mathcal{H}$ is finite or not, we can take

$$|A| = \mathcal{H}[S]$$

the set of distinct labels of points in S using concepts in $\mathcal{H}$. Then

$$|A| = \mathcal{H}[m]$$

the shatter coefficient of $\mathcal{H}$ on m points, and

$$\hat{\mathcal{R}}_m(\mathcal{H}) \leq \sqrt{\frac{2\log \mathcal{H}[m]}{m}}$$

By Sauer's lemma

$$|\mathcal{H}[m]| \leq m^d$$

where d is the VC dimension of $\mathcal{H}$. Thus, the above may be further simplified into

$$\hat{\mathcal{R}}_m(\mathcal{H}) \leq \sqrt{\frac{2\log m}{m}}$$

23. Rademacher Average on the Loss Class vs. Hypothesis Class:

$$\mathcal{R}(\mathcal{L}) = \mathbb{E}_\sigma\left[\sup_{f \in \Im} \frac{1}{n}\sum_{i=1}^{n} \sigma_i 1_{f(X_i)\neq Y_i}\right] = \mathbb{E}_\sigma\left[\sup_{f \in \Im} \frac{1}{n}\sum_{i=1}^{n} \sigma_i \frac{1}{2}\{1 - Y_i f(X_i)\}\right]$$
$$= \frac{1}{2}\mathbb{E}_\sigma\left[\sup_{f \in \Im} \frac{1}{n}\sum_{i=1}^{n} \sigma_i Y_i f(X_i)\right] = \frac{1}{2}\mathcal{R}(\Im)$$

Using this we can relate the empirical loss bound to the empirical hypothesis bound.

VC Entropy

1. Distribution Dependent Bounds: The treatment so far has been distribution independent. We modify the treatment above to get a distribution dependent bound. We use the notation

$$\mathcal{N}(\mathcal{F}, \mathcal{Z}_1^n) \coloneqq \left|\mathcal{F}_{\mathcal{Z}_1,\ldots,\mathcal{Z}_n}\right|$$

2. VC Entropy Definition: The annealed VC entropy is defined as

$$\mathcal{H}_{\mathcal{F}}(n) = \log \mathbb{E}[\mathcal{N}(\mathcal{F}, \mathcal{Z}_1^n)]$$

Using this definition, for any δ, with probability at least $1 - \delta$, we now show that, by extending the VC theory

$$R(f) - R_n(f) \leq 2\sqrt{2\frac{\mathcal{H}_{\mathcal{F}}(n) + \log\frac{2}{\delta}}{n}}$$

3. Distribution Dependent Bounds Estimator: As before, we begin with the Symmetrization lemma, so we have to upper bound the quantity

$$\mathcal{I} = \mathbb{P}_{\mathcal{X}}\left[\sup_{f \in \mathcal{F}_{\mathcal{Z}_1^n, \mathcal{Z}'^n_1}} (P'_n - P_n)f \geq \frac{\varepsilon}{2}\right]$$

4. Distribution Dependent Rademacher Bounds: Using Rademacher variables σ_i, we note that

$$(P'_n - P_n)f$$

and

$$\frac{1}{n}\sum_{i=1}^{n} \sigma_i[f(\mathcal{Z}_i) - f(\mathcal{Z}'_i)]$$

have the same distribution, since changing one σ_i corresponds to exchanging $\mathcal{Z}_i$ and $\mathcal{Z}'_i$.

5. Distribution Dependent Bounds - Union Bounding: Applying Rademacher variables as shown above, we get

$$\mathcal{I} \leq \mathbb{E}_{\mathcal{X}}\left\{\mathbb{P}_{\sigma}\left[\sup_{f \in \mathcal{F}_{\mathcal{Z}_1^n, \mathcal{Z}'^n_1}} \frac{1}{n}\sum_{i=1}^{n} \sigma_i[f(\mathcal{Z}_i) - f(\mathcal{Z}'_i)] \geq \frac{\varepsilon}{2}\right]\right\}$$

and union bounding followed by the application of the distribution-dependent shattering leads to

$$\mathcal{I} \leq \mathbb{E}_{\mathcal{X}} \left\{ \mathcal{N}(\mathcal{F}, \mathcal{Z}_1^n, \mathcal{Z}'^n_1) \underset{f \in \mathcal{F}_{\mathcal{Z}_1^n, \mathcal{Z}'^n_1}}{sup} \mathbb{P}_\sigma \left[\frac{1}{n} \sum_{i=1}^{n} \sigma_i [f(\mathcal{Z}_i) - f(\mathcal{Z}'_i)] \geq \frac{\varepsilon}{2} \right] \right\}$$

6. Distribution Dependent Bounds - Final Step: Noting that

$$\sigma_i [f(\mathcal{Z}_i) - f(\mathcal{Z}'_i)] \in [-1, 1]$$

and applying Hoeffding's inequality finally gives

$$\mathcal{I} \leq \mathbb{E}_{\mathcal{X}} \left\{ \mathcal{N}(\mathcal{F}, \mathcal{Z}_1^n, \mathcal{Z}'^n_1) e^{-\frac{n\varepsilon^2}{8}} \right\}$$

Finally, observe that just as

$$\log[\mathbb{E}\{\mathcal{N}(\mathcal{F})\}] \rightarrow \mathcal{S}_{\mathcal{F}}(2n)$$

before we now have

$$\log[\mathbb{E}\{\mathcal{N}(\mathcal{F}, \mathcal{Z}_1^n, \mathcal{Z}'^n_1)\}] \rightarrow \mathcal{H}_{\mathcal{F}}(2n)$$

Applying this last step, we get the bounds above.

Chaining Technique

1. Introduction: Although the Rademacher bounding produces bounds that are comparable to the traditional VC theory, i.e.

$$\mathcal{R}(\mathcal{F}) \leq 2\sqrt{\frac{\log \mathcal{N}(\mathcal{F})}{n}}$$

the dependence on n can be improved by using the *Chaining* technique. The idea is to use the covering numbers at all scales to capture the geometry of the class in a better way than the VC entropy class.

2. Dudley-Haussler Entropy Bound: Using the chaining technique, one has the following so-called Dudley's entropy bound:

$$\mathcal{R}_n(\mathcal{F}) \leq 2\frac{\mathcal{C}}{\sqrt{n}}\int_0^{\infty}\sqrt{\log \mathcal{N}(\mathcal{F}, \varepsilon, n)}\, d\varepsilon$$

As a consequence, along with the Haussler's upper bound, we get the following result:

$$\mathcal{R}_n(\mathcal{F}) \leq \mathcal{K}\sqrt{\frac{h}{n}}$$

We can, with this approach, thus remove the unnecessary $\log n$ factor of the VC bound.

Literature

1. Bounded Differences Inequality - Martingale Methods: The bounded differences inequality was first formulated explicitly by McDiarmid (1989), who proved it using the Martingale method (McDiarmid (1989, 1998)).
2. Bounded Differences Inequality – Information Theoretic Methods: Concentration results closely related to martingale methods have been obtained using information theoretic methods (Ahlswede, Gacs, and Korner (1976), Marton (1986, 1996a, 1996b), Dembo (1997), Massart (1998), and Rio (2001)).

3. Bounded Differences Inequality – Talagrand's Induction Method: See Talagrand (1995, 1996a, 1996b), Panchenko (2001, 2002), McDiarmid (2002), Luczak and McDiarmid (2003), Panchenko (2003).
4. Bounded Differences Inequality – Entropy Methods: The *Entropy Method*, based on logarithmic Sobolev inequalities, was developed by Ledoux (1996, 1997). See also Bobkov and Ledoux (1997), Massart (2000), Boucheron, Lugosi, and Massart (2000), Rio (2001), Bousquet (2002), Boucheron, Lugosi, and Massart (2003), and Boucheron, Bousquet, Lugosi, and Massart (2004).
5. Rademacher Averages: The use of Rademacher averages in classification was first promoted by Koltchinskii (2001), and Bartlett, Boucheron, and Lugosi (2001). Details are available in additional surveys by Koltchinskii and Panchenko (2000, 2002), Bartlett and Mendelson (2002), Bartlett, Bousquet, and Mendelson (2002), Bousquet, Koltchinskii, and Panchenko (2002), and Antos, Kegl, Linder, and Lugosi (2002).

References

- Ahlswede, R., P. Gacs, and J. Korner (1976): Bounds on Conditional Probabilities with Applications in multi-user Communication *Zeitschrift fur Wahrscheinlichkeitstheorie und verwandte Gebiete* **34** 157-177 (with corrections in **39** 353-354 (1977)).
- Antos, A., B. Kegl, T. Linder, and G. Lugosi (2002): Data-dependent Margin-based Generalization Bounds for Classification *Journal of Machine Learning Research* **3** 73-98.
- Bartlett, P., S. Boucheron, and G. Lugosi (2001): Model Selection and Error Estimation *Machine Learning* **48** 85-113.
- Bartlett, P., O. Bousquet, and S. Mendelson (2002): Localized Rademacher Complexities, in: *Proceedings of the 15th Annual Conference on Computational Learning Theory* 44-48.
- Bartlett, P., and S. Mendelson (2002): Rademacher and Gaussian Complexities: Risk Bounds and Structural Results *Journal of Machine Learning Research* **3** 463-482.
- Bobkov, S., and M. Ledoux (1997): Poincare's Inequalities and Talagrand's Concentration Phenomenon for the Exponential Distribution *Probability Theory and Related Fields* **107** 383-400.

- Boucheron, S., G. Lugosi, and P. Massart (2000): A Sharp Concentration Inequality with Applications *Random Structures and Algorithms* **16** 277-292.
- Boucheron, S., G. Lugosi, and P. Massart (2003): Concentration Inequalities using the Entropy Method *Annals of Probability* **31** 1583-1614.
- Boucheron, S., O. Bousquet, G. Lugosi, and P. Massart (2004): Moment Inequalities for Functions of Independent Random Variables *Annals of Probability* **33 (2)** 514-560.
- Boucheron, S., O. Bousquet, and G. Lugosi (2005): Theory of Classification *ESAIM: Probability and Statistics* **9** 323-375.
- Bousquet, O. (2002): A Bennett Concentration Inequality and its Application to Suprema of Empirical Processes *C. R. Acad. Sci. Paris* **334** 495-500.
- Bousquet, O., V. Koltchinskii, and D. Panchenko (2002): Some Local Measures of Complexities of Convex Hulls and Generalization Bounds, in: *Proceedings of the 15th Annual Conference on Computational Learning Theory* **Springer** 59-73.
- Bousquet, O., S. Boucheron, and G. Lugosi (2003): Introduction to Statistical Learning Theory 169-207 *Advances in Machine Learning* (editors: Bousquet, O., U. von Luxburg, and G. Ratsch) **Springer** Berlin.
- Dembo, A. (1997): Information Inequalities and Concentration of Measure *Annals of Probability* **25** 927-939.
- Koltchinskii, V., and D. Panchenko (2000): Rademacher Processes and Bounding the Risk of Function Learning, in: *High Dimensional Probability II (editors: E. Gine, D. Mason, and J. Wellner)* 443-459.
- Koltchinskii, V. (2001): Rademacher Penalties and Structural Risk Minimization *IEEE Transactions on Information Theory* **47** 1902-1914.
- Koltchinskii, V. and D. Panchenko (2002): Empirical Margin Distribution and Bounding the Generalization Error of Combined Classifiers *Annals of Statistics* **30**.
- Ledoux, M. (1996): On Talagrand's Deviation Inequalities for Product Measures *ESAIM: Probability and Statistics* **1** 63-87.
- Ledoux, M. (1997): Isoperimetry and Gaussian Analysis *Lectures on Probability Theory and Statistics (editor: P. Bernard)* **Ecole d'Ete de Probabilites de St-Flour XXIV-1994** 165-294.

- Luczak, M. J., and C. McDiarmid (2003): Concentration for Locally Acting Permutations *Discrete Mathematics* **265** 159-171.
- Mandelson, S. (2003): A few Notes on Statistical Learning Theory *Advanced Lectures on Machine Learning* **LNCS** 1-40 **Springer**.
- Marton, K. (1986): A Simple Proof of the Blowing-up Lemma *IEEE Transactions on Information Theory* **32** 445-446.
- Marton, K. (1996a): Bounding d-distance by Informational Divergence: A Way to prove Measure Concentration *Annals of Probability* **24** 857-866.
- Marton, K. (1996b): A Measure Concentration Inequality for contracting Markov Chains *Geometric and Functional Analysis* **6** 556-571 (Erratum: **7** 609-613 (1997)).
- Massart, P. (1998): Optimal Constants for Hoeffding Type Inequalities *Technical Report **98.86**, Mathematiques* **Universite de Paris-Sud**.
- Massart, P. (2000): About the Constants in the Talagrand's Concentration Inequalities for Empirical Processes *Annals of Probability* **28** 863-884.
- McDiarmid, C. (1989): On the Method of Bounded Differences, in: *Surveys in Combinatorics* 148-188 **Cambridge University Press** Cambridge.
- McDiarmid, C. (1989): Concentration, in: *Probabilistic Methods for Algorithmic Discrete Mathematics (M. Habib, C. McDiarmid, J. Ramirez-Alfonsin, and B. Reed: editors)* 195-248 **Springer** New York.
- McDiarmid, C. (2002): Concentration for Independent Permutations *Combinatorics, Probability, and Computing* **2** 163-178.
- Panchenko, D. (2001): A Note on Talagrand's Concentration Inequality *Electronic Communications in Probability* **6**.
- Panchenko, D. (2002): Some Extensions of an Inequality of Vapnik and Chervonenkis *Electronic Communications in Probability* **7**.
- Panchenko, D. (2003): Symmetrization Approach to Concentration Inequalities for Empirical Processes *Annals of Probability* **31 (4)** 2068-2081.
- Rio, E. (2001): Inegalities de concentration pour les processus empiriques de classes de parties *Probability Theory and Related Fields* **119** 163-175.
- Talagrand, M. (1995): Concentration of Measures and Isoperimetric Inequalities in Product Spaces *Publications Mathematiques de l'I.H.E.S.* **81** 73-205.

- Talagrand, M. (1996a): A New Look at Independence *Annals of Probability* **24** 1-34 (Special Invited Paper).
- Talagrand, M. (1996b): New Concentration Inequalities in Product Spaces *Inventiones Mathematicae* **126** 505-563.

Local Rademacher Averages

Introduction

1. Definition: Local Rademacher averages refers to Rademacher averages of subsets of function class determined by a condition on the variance of the function. Formally, the local Rademacher average at a radius

$$r \geq 0$$

for the class $\mathfrak{I}$ is defined as

$$\mathcal{R}_n(\mathcal{F}, r) = \mathbb{E}_\sigma \left[\sup_{f \in \mathcal{F}: Pf^2 \leq r} R_n(f) \right]$$

2. Importance of the Local Variance Based Families: The rationale for the above definition/construction is that, as just seen, the crucial ingredient for better rates of convergence is to use the variance of the function. Localizing the Rademacher average allows us to focus on that part of the function class where the fast rate phenomenon occurs, i.e., the functions with small variance.

Star-Hull and Sub-root Functions

1. Sub-root Function: A function

$$\psi : \mathbb{R} \to \mathbb{R}$$

is a sub-root if:

   a. ψ is non-decreasing
   b. ψ is non-negative, and
   c. $\frac{\psi(r)}{\sqrt{r}}$ is non-increasing.

   These characteristics ensure that the sub-root function is continuous, and has a unique (non-zero) fixed-point r^* satisfying

$$\psi(r^*) = r^*$$

2. Star-Hull Definition: Let $\mathcal{F}$ be a set of functions. Its star-Hull is defined as

$$* \mathcal{F} = \{\alpha f : f \in \mathcal{F}, \alpha \in [0,1]\}$$

3. Motivation Behind the Star-Hull Construction: By taking the star-Hull of a class of functions, we are guaranteed that the local Rademacher average behaves as a sub-root function, and thus has a unique fixed-point. This fixed point is a key quantity in relative error bounds. Formally, we start that, for any class of functions $\mathcal{F}$, $\mathcal{R}_n(* \mathcal{F}, r)$ is a sub-root.
4. Star-Hull Hypothesis Space Expansion: One way to estimate the enlargement of the size of the function class space is to compare the metric entropy (i.e., the log of the covering numbers) of $\mathcal{F}$ and $* \mathcal{F}$. It is possible to see that the metric entropy increases only by a logarithmic factor, which is essentially negligible.

Local Rademacher Averages and Fixed Point

1. Basic Theorem: Let $\mathcal{F}$ be a class of bounded functions, i.e.

$$f \in [-1, 1]$$

and r^* be the fixed point of $\mathcal{R}_n(* \mathcal{F}, r)$. There exists a constant

$$c > 0$$

such that, with probability at least $1 - \delta$

$$\forall f \in \mathcal{F}\ Pf - P_n f \leq c \left[\sqrt{r^*\, Var\, f} + \frac{\log \frac{1}{\delta} + \log \log n}{n} \right]$$

If, in addition, the functions in $\mathcal{F}$ satisfy

$$Var\, f \leq c(Pf)^{\alpha}$$

then one obtains, with probability at least $1 - \delta$

$$\forall f \in \mathcal{F}\ Pf \leq c \left[P_n f + (r^*)^{\frac{1}{2-\alpha}} + \frac{\log \frac{1}{\delta} + \log \log n}{n} \right]$$

2. Proof Step #1 - Talagrand's Inequality for Empirical Processes: The starting point is the Talagrand's inequality for empirical processes, a generalization of the McDiarmid's inequality of the Bernstein's type (i.e., it includes the variance). This inequality tells us, that with high probability

$$\sup_{f \in \mathcal{F}} Pf - P_n f \leq \mathbb{E}\left[\sup_{f \in \mathcal{F}} Pf - P_n f\right] + c\sqrt{\sup_{f \in \mathcal{F}} \frac{Var\, f}{n}} + \frac{c'}{n}$$

for some c, c'.

3. Proof Step #2 - Split into Variance sub-groups: The second step consists in "peeling" the class, that is, splitting the classes into sub-classes according to the variance of the functions

$$\mathcal{F}_k = \{f : Var\, f \in [x_k, x_{k+1}]\}$$

Note that although $Pf - P_n f$ depends on both the suprema across the expectation as well as the variance, here we classify the function space only on the variance.

4. Proof Step #3 - Apply Talagrand's Inequality to each sub-group: We then apply the Talagrand inequality to each sub-group separately to obtain, with high probability

$$\sup_{f \in \mathcal{F}_k} Pf - P_n f \leq \mathbb{E}\left[\sup_{f \in \mathcal{F}_k} Pf - P_n f\right] + c\sqrt{\sup_{f \in \mathcal{F}_k} \frac{Var\, f}{n}} + \frac{c'}{n}$$

5. Proof Step #4 - Rademacher Bounding to the Variance Sub-group: Now use the symmetrization lemma to introduce local Rademacher averages. We then get, with high probability

$$Pf - P_n f \leq 2\mathcal{R}_n\left(\mathcal{F}, \sup_{f \in \mathcal{F}_k} Var\, f\right) + c\sqrt{\sup_{f \in \mathcal{F}_k} \frac{Var\, f}{n}} + \frac{c'}{n}$$

6. Proof Step #5 - Express Local Rademacher Average in terms of the Fixed Point: We then "solve" the above inequality for $Pf - P_n f$. Things are simple if

$$\mathcal{R}_n\left(\mathcal{F}, \sup_{f \in \mathcal{F}_k} Var\, f\right)$$

behaves like a square root function, since we can upper bound the local Rademacher average by its fixed-point value. With high probability, we then obtain

$$Pf - P_n f \leq 2\sqrt{r^* Var\, f} + c\sqrt{\sup_{f \in \mathcal{F}_k} \frac{Var\, f}{n}} + \frac{c'}{n}$$

7. Proof Step #6 - Relation Between the Variance and the Expectation: Finally, we obtain the relationship between the variance and the expectation

$$Var\, f \leq c(Pf)^{\alpha}$$

and "solve" the inequality in Pf to get the original result.

Local Rademacher Average – Consequences

1. Fast Rate: An important example in this case is where the class $\mathcal{F}$ is of finite VC dimension h. In that case, one has

$$\mathcal{R}(\mathcal{F}, r) \leq \mathcal{C} \sqrt{\frac{rh \log n}{n}}$$

so that

$$r^* \leq \mathcal{C}' \frac{rh \log n}{n}$$

As a consequence, under the Tsybakov noise condition, we obtain a rate of convergence of $P_n f$ to Pt as

$$\mathcal{O}\left(\frac{1}{n^{\frac{1}{2-\alpha}}}\right)$$

It is important to note that, in this case, the rate of convergence of $P_n f$ to Pf is

$$\mathcal{O}\left(\frac{1}{\sqrt{n}}\right)$$

Thus, we obtain a fast rate by looking at the relative error. These fast rates can be obtained provided

$$t \in \mathcal{F}$$

but it is not necessary that

$$R^* = 0$$

This requirement can be removed if one uses structural risk minimization or regularization.

2. Conditional Data Dependent Rademacher Averages: Another related result is that, as in the global case, one can obtain a bound with data-dependent (i.e., conditional) local Rademacher averages for

$$\mathcal{R}_n(\mathcal{F}, r) = \mathbb{E}_\sigma \left[\sup_{f \in \mathcal{F}: Pf^2 \leq r} R_n(f) \right]$$

The result is the same as before (but with different constants) under the same conditions. With probability at least $1 - \delta$

$$Pf \leq \mathcal{D} \left[P_n f + (r_n^*)^{\frac{1}{2-\alpha}} + \frac{\log \frac{1}{\delta} + \log \log n}{n} \right]$$

where r_n^* is the fixed-point of a sub-root upper bound of $\mathcal{R}_n(\mathcal{F}, r)$.

3. Conditional Local Rademacher Averages - Asymptotic Noise Behavior: Hence, we get improved rates when the noise is well-behaved, and these rates interpolate between $n^{-\frac{1}{2}}$ and n^{-1}. However, in general it is not possible to estimate the parameters $\mathcal{D}$ and α that enter in the noise condition.
4. Local Rademacher Averages - Local Capacity Measure: Although the capacity measure used seems "local", it does depend on all the functions in the class (through the variance), but each of them is appropriately re-scaled. Indeed, in $\mathcal{R}_n(* \mathcal{F}, r)$, each function

$$f \in \mathcal{F}$$

with

$$Pf^2 \geq r$$

is considered at a scale $\frac{r}{Pf^2}$.

Normalized ERM

Background

1. The Concept: The main idea is to consider the ratio

$$\frac{Pf - P_n f}{\sqrt{Pf}}$$

where

$$f \in \{0, 1\}$$

and

$$Var\, f \leq Pf^2, Pf$$

2. Motivation: The motivation for considering the above normalized variant is that the fluctuations of the variants from $\Im$ are more "uniform". Hence, the supremum in

$$\sup_{f \in \Im} \frac{Pf - P_n f}{\sqrt{Pf}}$$

is not necessarily attained at those functions whose variance is large, as in the previous case. Moreover, since we know that our goal is to find functions with small error Pf (hence small variance), the normalized supremum takes this into account.

Computing the Normalized Empirical Risk Bounds

1. Statement: See Vapnik and Chervonenkis (1971) and Bousquet, Boucheron, and Lugosi (2003). For

$$\delta > 0$$

with probability at least $1 - \delta$

$$\forall f \in \Im \frac{Pf - P_n f}{\sqrt{Pf}} \leq 2\sqrt{\frac{\log \mathcal{S}_{\mathcal{F}}(2n) + \log\frac{4}{\delta}}{n}}$$

and also with probability at least $1 - \delta$

$$\forall f \in \Im \frac{Pf - P_n f}{\sqrt{P_n f}} \leq 2\sqrt{\frac{\log \mathcal{S}_{\mathcal{F}}(2n) + \log\frac{4}{\delta}}{n}}$$

2. Proof Step #1 - Using Symmetrization: The first step uses a variant of the symmetrization lemma:

$$\mathbb{P}\left[\sup_{f \in \Im} \frac{Pf - P_n f}{\sqrt{Pf}} \geq \varepsilon\right] \leq 2\mathbb{P}\left[\sup_{f \in \Im} \frac{P'_n f - P_n f}{\sqrt{\frac{P'_n f + P_n f}{2}}} \geq \varepsilon\right]$$

3. Proof Step #2 - Use of Rademacher Variables: In this step we use randomization using Rademacher variables, i.e.

$$\mathbb{P}\left[\sup_{f \in \Im} \frac{P'_n f - P_n f}{\sqrt{\frac{P'_n f + P_n f}{2}}} \geq \varepsilon\right] = \mathbb{E}\left\{\mathbb{P}_\sigma\left[\sup_{f \in \Im} \frac{\frac{1}{n}\sum_{i=1}^{n} \sigma_i[f(\mathbb{Z}_i) - f(\mathbb{Z}'_i)]}{\sqrt{\frac{P'_n f + P_n f}{2}}} \geq \varepsilon\right]\right\}$$

4. Proof Step #3 - Bernstein + Union Bounding: Finally, one applies the union bound in conjunction with a tail bound of the Bernstein type.

Denormalized Bounds

1. Denormalized Bound Estimation: From the fact that for positive, A, B, and C,

$$A \leq B + C\sqrt{A} \rightarrow A \leq B + C^2 + \sqrt{B}C$$

we get

$$\forall f \in \Im\, Pf - P_n f \leq 2\sqrt{P_n f \frac{\log \mathcal{S}_{\mathcal{F}}(2n) + \log\frac{4}{\delta}}{n}} + 4\frac{\log \mathcal{S}_{\mathcal{F}}(2n) + \log\frac{4}{\delta}}{n}$$

2. Ideal, Noiseless Minimizer: In such a situation, there is no noise, i.e.

$$Y = t(X)$$

almost surely, and

$$t \in \mathfrak{I}_n$$

Denoting by f_n the empirical minimizer, we have

$$R^* = 0$$

as well as

$$R_n(f_n) = 0$$

in this case. Thus, in the situation where $\mathfrak{I}$ is a function space with VC dimension h we get

$$R(f_n) \approx \mathcal{O}\left[\frac{h \log n}{n}\right]$$

This clearly demonstrates that we de facto interpolate between the best rate of convergence

$$\mathcal{O}\left[\frac{h \log n}{n}\right]$$

and the worst rate of convergence

$$\mathcal{O}\left[\sqrt{\frac{h \log n}{n}}\right]$$

(the $\log n$ factor cannot be removed in this case).

3. Corresponding ERM Bound: Casting

$$Pf \to R(f)$$

in the approach above, we get

$$R(f_n) \leq R^* + 2\sqrt{R^* \frac{\log \mathcal{S}_{\mathcal{F}}(2n) + \log\frac{4}{\delta}}{n}} + 4\frac{\log \mathcal{S}_{\mathcal{F}}(2n) + \log\frac{4}{\delta}}{n}$$

thus when

$$R^* = 0$$

$$t \in \Im$$

and the convergence is $\frac{1}{n}$, while if

$$R^* > 0$$

the rate contains contribution from the $\frac{1}{\sqrt{n}}$ term. Therefore, it is not possible to obtain a rate with a power of n between $-\frac{1}{2}$ and -1.

4. Challenges with Denormalized Bounding: The main challenge with the Denormalized ERM approach is that the factor of the square root term R^* (which arises out of the denormalization term $\sqrt{Pf}$) is not a convenient entity to use here, since it does not vary with n. However, if we use $R(f_n) - R^*$ as the corresponding entity under the square root, this converges to zero with increasing n (see later on the relative error classes). However, the denormalized approach cannot be used for $f - f^*$, so we need another approach.

References

- Bousquet, O., S. Boucheron, and G. Lugosi (2003): Introduction to Statistical Learning Theory 169-207 *Advances in Machine Learning* (editors: Bousquet, O., U. von Luxburg, and G. Ratsch) **Springer** Berlin.
- Vapnik, V., and A. Chervonenkis (1971): On the Uniform Convergence of Relative Frequencies of Events to their Probabilities *Theory of Probability and its Applications* **16** 264-280.

Noise Conditions

SLT Analysis Metrics

1. Regression Function: Note that

$$\mathcal{P}(x, y) = \mathcal{P}(x) \times \mathcal{P}(y \mid x)$$

We define the regression function as

$$\eta(x) = \mathbb{E}[Y|X = x] = 2\mathbb{P}[Y = 1|X = x] - 1$$

and the target function (i.e., the Bayes' classifier) as

$$t(x) = sign[\eta(x)]$$

2. Noise Level: We define the noise level $s(x)$ as

$$s(x) = \min\{\mathbb{P}[Y = 1|X = x], 1 - \mathbb{P}[Y = 1|X = x]\}$$

In practice, a more useful definition (one that does not involve $\mathbb{P}$) is

$$s(x) = \frac{1 - \eta(x)}{2}$$

In the deterministic case

$$s(x) = 0$$

almost surely. Thus, the Bayes' risk is

$$\mathcal{R}^* = \mathbb{E}[s(x)]$$

3. Noise Condition Motivation: To improve the treatment/results above, we seek a refinement that requires us to make assumptions about the noise function $s(x)$ - in the ideal case

$$s(x) = 0$$

everywhere, thus

$$R^* = 0$$

and

$$Y = t(X)$$

Thus, we use the above-mentioned metrics to quantify how well-behaved the noise function is.

4. Range of the Noise and the Regression Function: From a classification point-of-view, the favorable classifier situation occurs when $\eta(x)$ is not too close to zero. Indeed

$$\eta(x) = 0$$

means that the noise is a maximum at that x and the corresponding

$$s(x) = \frac{1}{2}$$

Here the label is completely undetermined, as any prediction results in an error with a probability of $\frac{1}{2}$.

Types of Noise Conditions

1. Type 1 - Massart Noise Condition: For some c, assume

$$|\eta(x)| > \frac{1}{c}$$

almost surely (c may be viewed as a measure of the "odds against"). This condition implies that there is no region where the decision is completely random, i.e., that the noise is bounded away from $\frac{1}{2}$.

2. Type 2 - Tsybakov Noise Condition: Let

$$\alpha \in [0, 1]$$

and assume that one of the following equivalent conditions is satisfied:

a.

$$\exists\, c > 0, \forall f \in \{-1, 1\}^{\mathcal{X}}, \mathbb{P}[f(X)\eta(X) \leq 0] \leq c[R(f) - R^*]^{\alpha}$$

b.

$$\exists\, c > 0, \forall \mathcal{A} \subset \mathcal{X}, \int_{\mathcal{A}}^{\mathcal{A}} dP(x) \leq c\left[\int_{\mathcal{A}}^{\mathcal{A}} |\eta(x)| dP(x)\right]^{\alpha}$$

c.

$$\exists\, B > 0, \forall \varepsilon \geq 0, \mathbb{P}[|\eta(X)| \leq \varepsilon] \leq B\varepsilon^{\frac{\alpha}{1-\alpha}}$$

This condition is the easiest to interpret, as it indicates that $\eta(x)$ is close to the critical value 0 with low probability.

3. Equivalence of the Tsybakov Conditions:
   a. Equivalence between a) and c) => It is easy to check that

$$R(f) - R^* = \mathbb{E}\left[|\eta(x)| 1_{f\eta \leq 0}\right]$$

For each function f, there exists a set $\mathcal{A}$ such that

$$1_{\mathcal{A}} = 1_{f\eta \leq 0}$$

   b. Equivalence between b) and c) => Let

$$\mathcal{A} = \{x : |\eta(x)| \leq \varepsilon\}$$

$$\mathbb{P}[|\eta(x)| \leq \varepsilon] = \int_{\mathcal{A}}^{\mathcal{A}} dP(x) \leq c\left[\int_{\mathcal{A}}^{\mathcal{A}} |\eta(x)| dP(x)\right]^{\alpha} \leq c\varepsilon^{\alpha}\left[\int_{\mathcal{A}} dP(x)\right]^{\alpha}$$
$$\Longrightarrow \mathbb{P}[|\eta(x)| \leq \varepsilon] \leq c^{\frac{1}{1-\alpha}} \varepsilon^{\frac{\alpha}{1-\alpha}} = B\varepsilon^{\frac{\alpha}{1-\alpha}}$$

where

$$B = c^{\frac{1}{1-\alpha}}$$

   c. Equivalence between Tsybakov Conditions a) and c) => We write

$$\begin{aligned} R(f) - R^* = \mathbb{E}\left[|\eta(x)| 1_{f\eta \leq 0}\right] &\geq \varepsilon \mathbb{E}\left[1_{f\eta \leq 0} 1_{|\eta(x)| > \varepsilon}\right] \\ &= \varepsilon \mathbb{P}[|\eta(x)| > \varepsilon] - \varepsilon \mathbb{E}\left[1_{f\eta > 0} 1_{|\eta(x)| > \varepsilon}\right] \\ &\geq \varepsilon\left[1 - B\varepsilon^{\frac{\alpha}{1-\alpha}}\right] - \varepsilon \mathbb{P}[f\eta > 0] = \varepsilon\left[\mathbb{P}[f\eta \leq 0] - B\varepsilon^{\frac{\alpha}{1-\alpha}}\right] \end{aligned}$$

Setting

$$\varepsilon = \left\{\frac{(1-\alpha)\mathbb{P}[f\eta \leq 0]}{B}\right\}^{\frac{\alpha}{1-\alpha}}$$

finally gives

$$\mathbb{P}[f\eta \leq 0] \leq \frac{B^{1-\alpha}}{(1-\alpha)\alpha^{\alpha}}[R(f) - R^*]^{\alpha}$$

and setting

$$c = \frac{B^{1-\alpha}}{(1-\alpha)\alpha^{\alpha}}$$

recovers a).

4. Restriction on the Range of α: The parameter α has to be in the range $[0, 1]$. Otherwise we end up with the opposite inequality

$$R(f) - R^* = \mathbb{E}\left[|\eta(x)|1_{f\eta \leq 0}\right] \leq \mathbb{E}\left[1_{f\eta \leq 0}\right] = \mathbb{P}[f\eta \leq 0]$$

which is incompatible with condition a) if

$$\alpha > 1$$

(i.e., this is a special case).

5. Equivalence of the Tsybakov and the Massart Cases: When

$$\alpha = 0$$

the Tsybakov condition becomes invalid, and when

$$\alpha = 1$$

it is equivalent to the Massart's condition.

Relative Loss Class

1. Motivation: The conditions above that we impose on the noise yield a crucial relationship between the variance and the expectation of functions within the so-called relative loss class (i.e., the difference between the sample-specific error and the empirical Bayes' error) defined as

$$\tilde{\mathfrak{I}} = \{(x, y) \mapsto f(x, y) - 1_{t(x)\neq y} : f \in \mathfrak{I}\}$$

   This relationship will allow exploiting the Bernstein-type inequalities applied to this latter class.
2. Massart and Tsybakov Noise Relative Error: Under Massart's condition, one has

$$\mathbb{E}\left[\left(1_{f(x)\neq y} - 1_{t(x)\neq y}\right)^2\right] \leq c[R(f) - R^*]$$

   or equivalently, for

$$f \in \tilde{\mathfrak{I}}$$

$$Var\, f \leq Pf^2 \leq cPf$$

Under Tsybakov's condition, this becomes, for

$$f \in \Im$$

$$\mathbb{E}\left[\left(1_{f(x)\neq y} - 1_{t(x)\neq y}\right)^2\right] \leq c[R(f) - R^*]^\alpha$$

and for

$$f \in \widetilde{\Im}$$

$$Var\, f \leq Pf^2 \leq c(Pf)^\alpha$$

3. Bernstein Inequality on Tsybakov Condition: In the finite case, with

$$|\Im| = N$$

one can apply the Bernstein inequality to $\widetilde{\Im}$ and get (in conjunction with the finite union bound), with a probability of at least $1 - \delta$, for all

$$f \in \Im$$

$$R(f) - R^* \leq R_n(f) - R_n(t) + \sqrt{\frac{8c[R(f) - R^*]^\alpha \log\frac{N}{\delta}}{n}} + \frac{4\log\frac{N}{\delta}}{3n}$$

4. Bernstein-Tsybakov Bound - $t \in \mathfrak{J}$ Case: When

$$t \in \mathfrak{J}$$

f_n is the minimizer of the empirical error, and hence

$$R_n(f) \leq R_n(t)$$

one has

$$R(f) - R^* \leq c \left[\frac{\log \frac{N}{\delta}}{n} \right]^{\frac{1}{2-\alpha}}$$

which is always better than $n^{-\frac{1}{2}}$ for

$$\alpha > 0$$

and is valid even if

$$R^* > 0$$

VC Theory and VC Dimension

Introduction

1. Purpose: A form of the computational learning theory, VC theory aims to explain the learning process from a statistical point of view by applying the empirical processes independent of the probability distribution (VC Theory (Wiki)).
2. Components of the VC Theory: As detailed in Vapnik (1989, 2000), the VC theory covers 4 main parts:
    a. Theory of consistency of the learning process => What are the necessary and sufficient conditions for the consistency of a learning process based on empirical risk minimization?
    b. Non-asymptotic theory of the rate of convergence of a learning process => How fast is the rate of convergence of the learning process?
    c. Theory of control of the generalization ability of a learning process => How can one control the rate of convergence (i.e., the generalization ability) of the learning process?
    d. Theory of construction learning machines => How can one construct the algorithms that control the generalization ability?

Empirical Processes

1. Background: Let $X_1, \dots, X_n$ be the random elements defined on a measurable space (X, A). Define the empirical measure

$$\mathbb{P} = \frac{1}{n}\sum_{i=1}^{n} \delta(X_i)$$

where δ stands for the Dirac notation. Further, using the notation of van der Vaart and Wellner (2000), denote

$$\mathbb{Q}f = \int f d\mathbb{Q}$$

for any probability measure $\mathbb{Q}$.

2. The Empirical Measure Map: Let $\mathfrak{I}$ be a class of measurable functions

$$f: X \to \mathbb{R}$$

The empirical measure above induces a map from $\mathfrak{I}$ to $\mathbb{R}$ given by

$$f \to \mathbb{P}_n f$$

Notice the similarity between the success counting measure $\mathbb{P}_n$ and the $0-1$ empirical loss function.

3. Empirical Process Theory: Let

$$\|\mathbb{Q}\|_{\mathfrak{I}} = sup\{|\mathbb{Q}f|: f \epsilon \mathfrak{I}\}$$

where sup is the supremum operator on the set, representing the least upper bound of $|\mathbb{Q}f|$. Empirical process theory aims at identifying the classes $\mathfrak{I}$ for which the following statements hold. In both cases we assume that the underlying distribution of the data $\mathbb{P}$ is unknown in practice.

   a.

$$\|\mathbb{P}_n - \mathbb{P}\|_{\mathfrak{I}} \to 0$$

      i.e., this is the uniform law of large numbers

   b.

$$\mathbb{G}_n = \sqrt{n}(\mathbb{P}_n - \mathbb{P}) \to \mathbb{G} \in l^\infty(\mathfrak{I})$$

i.e., this is the generalized uniform central theorem limit.

4. Glivenko-Cantelli and $\mathbb{P}$-Donsker Classes: If the law of large numbers can be explicitly established across all elements of $\mathfrak{I}$, the class $\mathfrak{I}$ is called Glivenko-Cantelli. If the validity of the central limit theorem under the assumption

$$\sup_{f \in \mathfrak{I}} |f(x) - \mathbb{P}f| < \infty \; \forall \, x$$

can be established, then the class $\mathfrak{I}$ is called $\mathbb{P}$-Donsker. Obviously a $\mathbb{P}$-Donsker class is Glivenko-Cantelli in probability by the application of the Slutsky's theorem.

5. Determination of $\mathfrak{I}$: The main challenge here is to determine $\mathfrak{I}$ that satisfies the LLN and the CLT criteria bove (under regularity conditions) for all

$$f \in \mathfrak{I}$$

Intuitively, $\mathfrak{I}$ cannot be too large, and the geometry of $\mathfrak{I}$ will play an important role.

6. Size of $\mathfrak{I}$: One way of measuring how big $\mathfrak{I}$ is is by using covering numbers. The covering number $N(\varepsilon, \mathfrak{I}, \|.\|)$ is the minimum number of balls

$$\{g : \|g - f\| < \varepsilon\}$$

needed to cover $\mathfrak{I}$ fully (obviously assuming there is an underlying norm on $\mathfrak{I}$). The entropy number is the logarithm of the covering number.

7. Sufficiency conditions for $\mathfrak{I}$ using Glivenko-Cantelli: A class $\mathfrak{I}$ is Glivenko-Cantelli if it is $\mathbb{P}$-measurable inside of an envelope $\mathbb{F}$ such that

$$\mathbb{P}^*\mathbb{F} < \infty$$

and satisfies

$$\sup_{\mathbb{Q}} \left[\varepsilon \|\mathbb{F}\|_{\mathbb{Q}}, \Im, L_1(\mathbb{Q})\right] < \infty$$

for every

$$\varepsilon > 0$$

8. Sufficiency Conditions for $\Im$ being $\mathbb{P}$-Donsker: This criterion is a version of the Dudley's theorem. If $\Im$ belongs to the class of functions such that

$$\int_0^\infty \sup_{\mathbb{Q}} \sqrt{logN\left[\varepsilon \sqrt{\int |\mathbb{F}|^2 d\mathbb{P}}, \Im, L_2(\mathbb{Q})\right]} d\varepsilon < \infty$$

then $\Im$ is $\mathbb{P}$-Donsker for every probability measure $\mathbb{P}$ such that

$$\mathbb{P}^* \mathbb{F}^2 < \infty$$

Bounding the Empirical Loss Function

1. Symmetrization: The majority of treatments on how to bound empirical processes rely on symmetrization, application of the maximal concentration inequalities, and chaining. Symmetrization is usually the first step in these proofs, and since it is used in many machine learning proofs on bounding empirical loss functions (including the proof of VC inequality) we treat it in some detail.
2. The Empirical Process and its Symmetrized Counterpart: Consider the empirical process

$$(\mathbb{P}_n - \mathbb{P})f = \frac{1}{n}\sum_{i=1}^{n} [f(X_i) - \mathbb{P}f]$$

Here we establish the connection between the empirical processes above and its symmetrized equivalent

$$\mathbb{P}_n^0 f = \frac{1}{n}\sum_{i=1}^{n} \varepsilon_i f(X_i)$$

where ε_i is an independent random variable that can be ± 1 with a probability 0.5 each. This symmetrized process, therefore, is a Rademacher process, conditional on the data X_i. Therefore this process is a sub-Gaussian process by the Hoeffding's inequality.

3. The Symmetrization Inequality: The Symmetrization bounds the LLN criterion of the empirical process theory. For every non-decreasing convex

$$\Phi : \mathbb{R} \to \mathbb{R}$$

and a class of measurable functions $\Im$

$$\mathbb{E}[\Phi(\|\mathbb{P}_n - \mathbb{P}\|_{\Im})] \leq \mathbb{E}[\Phi(2\|\mathbb{P}_n^0\|_{\Im})]$$

4. Conceptual idea behind the proof by Symmetrization: The proof of the symmetrization lemma lies on introducing independent copies of the original variables X_i (sometimes referred to as the ghost sample) and replacing the inner expectation of $\Phi(\|\mathbb{P}_n - \mathbb{P}\|_{\Im})$ with these copies. After an application of Jensen's inequality, different signs could be introduced (hence the name symmetrization) without changing the expectation.
5. Proof Steps:
    a. Introduction of the Ghost Sample => The empirical measure $\mathbb{P}f$ is replaced by $\mathbb{E}[f(Y_i)]$ – thereby providing the motivation for introducing the ghost sample variables $Y_1, \ldots, Y_n$ as independent copies of $X_1, \ldots, X_n$. For fixed values of $X_1, \ldots, X_n$ one has

$$\|\mathbb{P}_n - \mathbb{P}\|_{\Im} = \sup_{f \in \Im} \frac{1}{n} \left| \sum_{i=1}^{n} [f(X_i) - \mathbb{E}[f(Y_i)]] \right|$$
$$\leq \mathbb{E}_Y \left[\sup_{f \in \Im} \frac{1}{n} \left| \sum_{i=1}^{n} [f(X_i) - f(Y_i)] \right| \right]$$

b. Apply Jensen's Inequality => Apply the convex operator Φ to both sides:

$$\Phi(\|\mathbb{P}_n - \mathbb{P}\|_{\Im}) \leq \mathbb{E}_Y \left[\Phi \left\{ \frac{1}{n} \left\| \sum_{i=1}^{n} [f(X_i) - f(Y_i)] \right\|_{\Im} \right\} \right]$$

We are able to switch the locations of E_Y and Φ, because Φ is convex, and we have applied Jensen's inequality to convert this to an inequality.

c. Take expectation with respect to X:

$$\mathbb{E}[\Phi(\|\mathbb{P}_n - \mathbb{P}\|_{\Im})] \leq \mathbb{E}_X \left[\mathbb{E}_Y \left[\Phi \left\{ \frac{1}{n} \left\| \sum_{i=1}^{n} [f(X_i) - f(Y_i)] \right\|_{\Im} \right\} \right] \right]$$

Given that only X is the variate on the RHS, the expectation simply reduces to $\mathbb{E}_X$ on RHS (and $\mathbb{E}$ on LHS).

d. Sign Perturbation Step => Note that adding a minus sign ahead of $[f(X_i) - f(Y_i)]$ does not alter the RHS, because it is a symmetric function of $f(X_i)$ and $f(Y_i)$. Thus, the RHS remains the same under sign perturbation:

$$\mathbb{E}_X \left[\mathbb{E}_Y \left[\Phi \left\{ \frac{1}{n} \left\| \sum_{i=1}^{n} e_i [f(X_i) - f(Y_i)] \right\|_{\Im} \right\} \right] \right]$$

for a random Rademacher sequence e_i given as

$$(e_1, e_2, \dots, e_n) \in \{-1, +1\}$$

Therefore

$$\mathbb{E}[\Phi(\|\mathbb{P}_n - \mathbb{P}\|_{\mathfrak{I}})] \leq \mathbb{E}_\varepsilon \left[\mathbb{E} \left[\Phi \left\{ \frac{1}{n} \left\| \sum_{i=1}^{n} \varepsilon_i [f(X_i) - f(Y_i)] \right\|_{\mathfrak{I}} \right\} \right] \right]$$

e. Triangle Inequality and Convexity of Φ =>

$$\mathbb{E}\Phi(\|\mathbb{P}_n - \mathbb{P}\|_{\mathfrak{I}}) \leq \frac{1}{2}\mathbb{E}_\varepsilon \mathbb{E}\Phi \left\{ 2 \left\| \frac{1}{n} \sum_{i=1}^{n} \varepsilon_i f(X_i) \right\|_{\mathfrak{I}} \right\} + \frac{1}{2}\mathbb{E}_\varepsilon \mathbb{E}\Phi \left\{ 2 \left\| \frac{1}{n} \sum_{i=1}^{n} \varepsilon_i f(Y_i) \right\|_{\mathfrak{I}} \right\}$$

Since the 2 right hand terms are equal, we get

$$\mathbb{E}\Phi(\|\mathbb{P}_n - \mathbb{P}\|_{\mathfrak{I}}) \leq \mathbb{E}_\varepsilon \mathbb{E}\Phi \left\{ 2 \left\| \frac{1}{n} \sum_{i=1}^{n} \varepsilon_i f(X_i) \right\|_{\mathfrak{I}} \right\} \leq \mathbb{E}\Phi(2\|\mathbb{P}_n^0\|_{\mathfrak{I}})$$

6. Proving Empirical CLT's Using Symmetrization: Proof for the CLT's proceed analogously to the LNN. First use symmetrization to pass the process to $\mathbb{P}_n^0$, and then argue conditionally on the data using the fact that Rademacher processes are simple processes with nice properties.

VC Dimension - Introduction

1. Motivation: Although we have formulated the generalization bound in terms of the shattering coefficient, shattering coefficients in themselves are hard to evaluate. Another alternate capacity measure - the VC dimension – can be used to characterize the growth behavior of the shattering coefficient using a single number.
2. The VC Dimension: The VC Dimension of $\mathfrak{I}$, $\mathcal{VC}(\mathfrak{I})$, is defined as the largest number n such that there exists a sample of size n which is shattered by $\mathfrak{I}$; i.e.

$$\mathcal{VC}(\mathfrak{I}) = max\{n \in \mathbb{N} : \left|\mathfrak{I}_{\mathbb{Z}_n}\right| = 2^n \; for \; some \; \mathbb{Z}_n\}$$

If the maximum does not exist, the VC dimension is defined to be infinite. For other studies and many of the properties of the VC Dimension please refer to Cover (1965), Steele (1978), Dudley (1979), Wenocur and Dudley (1981), Assouad (1983), Khovanskii (1991), Anthony and Biggs (1992), MacIntyre and Sontag (1993), Kearns and Vazirani (1994), Goldberg and Jerrum (1995), Karpinski and MacIntyre (1997), Koiran and Sontag (1997), Anthony and Bartlett (1999), and Dudley (1999) demonstrate VC dimensions for several classes of functions.

VC Dimension - Formal Definition

1. Problem Space: Let C be a concept class over an infinite space X, i.e., a set of functions from X to $\{0,1\}$ (both X and C may be infinite). For any

$$S \subseteq X$$

$C(S)$ denotes the set of all labels or dichotomies on S that are induced or realized by C, i.e., if

$$S = \{x_1, \ldots, x_m\}$$

then

$$C(S) \subseteq \{0,1\}^m$$

therefore

$$C(S) = \{h(x_1), \dots, h(x_m) | c \in C\}$$

For any natural number m, we consider $C[m]$ to be the number of ways to split m points using concepts in C, that is

$$C[m] = max\{|C(S)|;\ |S| = m;\ S \subseteq X\}$$

2. Shattering of C by S: If

$$|C(S)| = 2^{|S|}$$

then S is said to be *shattered* by C.

3. VC Dimension of C: The *Vapnik-Chervonenkis* dimension of C, denoted as $\mathcal{VCDim}(C)$, is the cardinality of the largest set S shattered by C. If arbitrarily large finite sets can be shattered by C, then

$$\mathcal{VCDim}(C) = \infty$$

VC Dimension Examples

1. Computing the VC Dimension: In order to show that the VC Dimension of a class is at least d, we must find some shattered set of size d. In order to show that the VC Dimension is at most d, we must show that no set of size $d + 1$ is shattered.
2. VC Dimension Examples:
    a. Thresholds on a Number Line => Let C be the concept class of thresholds on the number line. Clearly samples of size 1 can be shattered by this class. However, no

sample of size 2 can be shattered, since it is impossible to choose a threshold such that x_1 is labeled positive and x_2 is labeled negative. Hence

$$\mathcal{VCDim}(C) = 1$$

b. Intervals on a Real Line => Here a sample of size 2 is shattered, but no sample of size 3 is shattered, since no concept can satisfy a sample whose middle point is negative and the outer points are positive. Hence the

$$\mathcal{VCDim}(C) = 2$$

c. k Non-intersecting Intervals => Let C be the concept class of k non-intersecting intervals on the real line. A sample of size $2k$ shatters (simply treating each pair of points as a separate case of the previous example), but no sample of size $2k + 1$ shatters, since if the sample points are alternated negative/positive, starting with a positive point, the positive points cannot all be covered by only k intervals. Hence

$$\mathcal{VCDim}(C) = 2k$$

d. Linear Separators in $\mathbb{R}^2$ => Here 3 points can be shattered, but not 4, so

$$\mathcal{VCDim}(C) = 3$$

In general, one can show that the $\mathcal{VCDim}(C)$ of the class of linear separators in $\mathbb{R}^n$ is $n + 1$.

e. Axis-aligned Rectangles => The class of axis-aligned rectangles in a plane has

$$\mathcal{VCDim}(C) = 4$$

The trick here is to note that for any collection of 5 points, at least one of them must be interior to, or on the boundary of any rectangle bounded by the other 4; hence if the bounding points are positive, the interior points cannot be made negative.

VC Dimension vs Popper's Dimension

1. The Comparison: Corfield, Scholkopf, and Vapnik (2005) pointed out that the VC dimension is related to Popper's notion of the dimension of a theory. They also highlight the differences between these approaches, including characterizing complexity using the number of parameters (e.g., the example of the class of thresholded sine waves in $\mathbb{R}$, Vapnik (1995)).
2. Popper's Dimension of a Theory: As stated in Popper (1959), if there exists, for a theory t, a field of singular (but not necessarily basic) statements such that, for some number d, the theory cannot be falsified for any d-tuple of the field, although it can be falsified by certain $(d+1)$-tuples, then we call d the characteristic number of the theory with respect to that field. All statements of the of the field whose degree of composition is less than or equal to d, are then compatible with the theory, and permitted by it, irrespective of the content.

References

- Anthony, M., and N. Biggs (1992): *Computational Learning Theory* **Cambridge University Press**.
- Anthony, M., and P. Bartlett (1999): *Neural Network Learning: Theoretical Foundations* **Cambridge University Press** Cambridge.
- Assouad, P. (1983): Densite et Dimension *Annales de l'Institut Fourier* **33** 233-282.
- Corfield, D., B. Scholkopf, and V. Vapnik (2005): Popper, Falsification, and VC Dimension *Technical Report TR-145* **Max Planck Institute for Biological Cybernetics**.

- Cover, T. (1965): Geometric and Statistical Properties of Systems of Linear Inequalities with Applications in Pattern Recognition *IEEE Transactions on Electronic Computers* **14** 326-334.
- Dudley, R. (1979): Balls in $\mathbb{R}^k$ do not cut all sub-sets of $k+2$ points *Advances in Mathematics* **31 (3)** 306-308.
- Dudley, R. (1999): *Uniform Central Limit Theorems* **Cambridge University Press** Cambridge.
- Goldberg, P., and M. Jerrum (1995): Bounding the Vapnik-Chervonenkis Dimension of Concept Classes Parametrized by Real Numbers *Machine Learning* **18** 131-148.
- Karpinski, M., and A. MacIntyre (1997): Polynomial Bounds for VC Dimension of Sigmoidal and General Pfaffian Neural Networks *Journal of Computer and System Science* **54**.
- Kearns, M., and U. Vazirani (1994): *An Introduction to Computational Learning Theory* **MIT Press** Cambridge, MA.
- Khovanskii, A. G. (1991): Fewnomials *Translations of Mathematical Monographs* **88** American Mathematical Society.
- Koiran, P., and E. Sontag (1997): Neural networks with Quadratic VC Dimension *Journal of Computer and System Science* **54**.
- MacIntyre, A., and E. Sontag (1993): Finiteness Results for Sigmoidal Neural Networks, in: *Proceedings of the 25th Annual ACM Symposium on the Theory of* Computing 325-334 **Association of Computing Machinery** New York.
- Popper, K. (1959): *The Logic of Scientific Discovery* (Hutchinson, Translation of *Logik der Forschung (1934)*).
- Steele, J. (1978): Existence of sub-matrices with all possible Columns *Journal of Combinatorial Theory* **A28** 84-88.
- Van der Waart, A. W., and J. A. Wellner (2000): *Weak Convergence and Empirical Processes with Applications to Statistics 2nd Edition* **Springer**.
- Vapnik, V. N. (1989): *Statistical Learning Theory* **Wiley Interscience**.
- Vapnik, V. N. (1995): *The Nature of Statistical Learning Theory* **Springer Verlag** New York.

- Vapnik, V. N. (2000): *The Nature of Statistical Learning Theory, 2nd Edition* **Springer Verlag**.
- Vapnik Chervonenkis Theory (Wiki): *Wikipedia Entry for Vapnik Chervonenkis Theory*.
- Wenocur, R., and R. Dudley (1981): Some special Vapnik-Chervonenkis Classes *Discrete Mathematics* **33** 313-318.

Sauer Lemma and VC Classifier Framework

Motivation

1. Growth Behavior of the VC Dimension: A combinatorial result proved simultaneously by Vapnik and Chervonenkis (1971), Sauer (1972), and Shelah (1972) characterizes the growth behavior of the shattering coefficient and relates it to the VC dimension – this is called the Sauer lemma. Related combinatorial results are available in Frankl (1983), Haussler (1995), Alekser (1997), Alon, Ben-David, Cesa-Bianchi, and Haussler (1997), Szarek and Talagrand (1997), and Cesa-Bianchi and Haussler (1998).
2. Sauer Lemma: Let $\mathfrak{I}$ be a function class with a finite VC dimension d. Then

$$\mathcal{N}(\mathfrak{I}, n) \leq \sum_{i=0}^{d} \binom{n}{i}$$

for all

$$n \in \mathbb{N}$$

In particular for all

$$n \geq d$$

we have

$$\mathcal{N}(\mathfrak{I}, n) \leq \left(\frac{en}{d}\right)^{d}$$

3. Nature of $\mathcal{N}(\mathfrak{I}, n)$: $\mathcal{N}(\mathfrak{I}, n)$ is very similar to the concept of covering numbers (in all their variants), and Sauer's lemma is an estimate of the bound of the shattering coefficient in terms of the sample size n, and the VC dimension.
4. Implication of Sauer's Lemma: Thus, for

$$n \geq d$$

the shattering coefficient behaves like a polynomial in the sample size n. Once we know that the VC dimension is finite, the shattering coefficients grow polynomially, and this implies ERM consistency. The converse is true too.

5. ERM Consistency Statement: ERM is consistent with respect to $\mathfrak{I}$ if and only if $\mathcal{VC}(\mathfrak{I})$ is finite.

Derivation of Sauer Lemma Bounds

1. Statement: If

$$\mathcal{VCD}im(C) = d$$

then for all m

$$C[m] \leq \Phi_d(m)$$

where

$$\Phi_d(m) = \sum_{i=0}^{d} \binom{m}{i}$$

2. Shattering Cardinality Polynomial Limit: We estimate $\Phi_d(m)$ in the limit of d. Note that

$$m > d$$

so

$$0 \leq \frac{d}{m} < 1$$

Thus

$$\left(\frac{d}{m}\right)^d \sum_{i=0}^{d} \binom{m}{i} \leq \sum_{i=0}^{d} \binom{m}{i} \left(\frac{d}{m}\right)^i \leq \sum_{i=0}^{m} \binom{m}{i} \left(\frac{d}{m}\right)^i = \left[1 + \frac{d}{m}\right]^m \leq e^d$$

Thus for

$$m > d$$

we have

$$\Phi_d(m) \leq \left(\frac{em}{d}\right)^d$$

3. General Sauer Complexity Results: Sauer's lemma can be used to get closed form expressions on sample complexity. For the treatment below, C is an arbitrary hypothesis space, D an arbitrary unknown fixed probability distribution, and c^* an arbitrary unknown target function.
4. Shattering Cardinality Bound - Recap: For any

$$\varepsilon, \delta > 0$$

if we draw a sample S from D of size m satisfying

$$2C[2m]2^{-\frac{m\varepsilon}{2}} \leq \delta$$

then with probability of at least $1-\delta$, all hypothesis in C with

$$err_D(h) > \varepsilon$$

are inconsistent with the data, i.e.

$$err_S(h) \neq 0$$

5. Sample Size Bound: For any

$$\varepsilon, \delta > 0$$

if we draw a sample S from D of size m satisfying

$$m \geq \frac{8}{\varepsilon}\left[d \log \frac{16}{\varepsilon} + \log \frac{2}{\delta}\right]$$

then with probability of at least $1-\delta$, all hypothesis in C with

$$err_D(h) > \varepsilon$$

are inconsistent with the data, i.e.

$$err_S(h) \neq 0$$

   a. Step #1 => From the shatter cardinality bound we've just seen, we get

$$2C[2m]2^{-\frac{m\varepsilon}{2}} \leq \delta \rightarrow 2^{\frac{m\varepsilon}{2}} \geq \frac{2C[2m]}{\delta} \rightarrow m \geq \frac{4}{\varepsilon}\log\frac{2C[2m]}{\delta}$$

b. Step #2 => Applying Sauer's lemma, and recognizing that

$$C[2m] \leq \left(\frac{em}{d}\right)^{d}$$

we can reduce the inequality for m to

$$m \geq \frac{4}{\varepsilon}\left[d\log\frac{2me}{d} + \log\frac{2}{\delta}\right] = \frac{4}{\varepsilon}\left[d\log m + d\log\frac{2e}{d} + \log\frac{2}{\delta}\right]$$

c. Step #3 => We use Taylor expansion to estimate the bound $\log m$. For

$$\alpha, x > 0$$

we get

$$\log x \leq \alpha x - \log\alpha - 1$$

so

$$\frac{4d}{\varepsilon}\log m \leq \frac{4d}{\varepsilon}\left[\frac{\varepsilon}{8d}m + \log\frac{8d}{\varepsilon} - 1\right] = \frac{m}{2} + \frac{4d}{\varepsilon}\log\frac{8d}{e\varepsilon}$$

d. Step #4 => Re-cast the above bounding for $\log m$ to get

$$m \geq \frac{m}{2} + \frac{4d}{\varepsilon}\log\frac{8d}{e\varepsilon} + \frac{4d}{\varepsilon}\log\frac{2e}{d} + \frac{4}{\varepsilon}\log\frac{2}{\delta} \rightarrow m \geq \frac{8}{\varepsilon}\left[d\log\frac{16}{\varepsilon} + \log\frac{2}{\delta}\right]$$

Sauer Lemma ERM Bounds

1. Growth Function - Definition: The growth function is the maximum number of ways into which n points can be classified by the function class $\mathfrak{I}$:

$$S_{\mathfrak{I}}(n) = \sup_{z_1, \ldots, z_n} \left|\mathfrak{I}_{z_1,\ldots,z_n}\right|$$

2. Risk Bound Using Growth Function: For any δ, with probability at least $1 - \delta$

$$R(f) - R_n(f) \leq 2\sqrt{2\frac{\log S_{\mathfrak{I}}(2n) + \log\frac{2}{\delta}}{n}}$$

3. Bounding with Sauer's Lemma: Let $\mathfrak{I}$ be a class of functions with a finite VC dimension h. Then, for all

$$n \in \mathbb{N}$$

$$S_{\mathfrak{I}}(n) \leq \sum_{i=0}^{h} \binom{n}{i}$$

and for all

$$n \geq h$$

$$S_{\mathfrak{I}}(n) \leq \left(\frac{en}{h}\right)^{h}$$

This implies, that with probability at least $1 - \delta$

$$\forall f \in \Im\, R(f) - R_n(f) \leq 2\sqrt{2\frac{h\log\frac{en}{h} + \log\frac{2}{\delta}}{n}}$$

Thus, this indicates that the difference between the TRUE and the EMPIRICAL risk is at most of the order of

$$\sqrt{\frac{h\log n}{n}}$$

4. Behavior of the Growth Function: The growth function, which is exponential in the sample size n as long as

$$n \leq h$$

where h is the VC Dimension, becomes polynomial in n for

$$n > h$$

VC Index

1. Motivation and Setup: Consider a collection $\mathbb{C}$ of subsets of the sample space χ. The collection of sets $\mathbb{C}$ is said to pick out a certain subset of finite set

$$S = \{x_1, \ldots, x_n\} \in \chi$$

if

$$S = S \cap C$$

from

$$C \in \mathbb{C}$$

$\mathbb{C}$ is said to shatter S if it picks out each of its 2^n subsets. The VC-index $V(\mathbb{C})$ of $\mathbb{C}$ is the smallest number n for which NO SET of size n is shattered by $\mathbb{C}$ (in fact, the VC-index is similar the $VC_{Dimension} + 1$ for an appropriately chosen classifier set).

2. Bounds on the VC Class Subset Number: Sauer's lemma then states that the number $\Delta_n(\mathbb{C}, x_1, \ldots, x_n)$ of subsets picked out by a VC class $\mathbb{C}$ satisfies

$$\max_{x_1, \ldots, x_n} \Delta_n(\mathbb{C}, x_1, \ldots, x_n) \leq \sum_{j=0}^{V(\mathbb{C})-1} \binom{n}{j} \leq \left(\frac{ne}{V(\mathbb{C}) - 1}\right)^{V(\mathbb{C})-1}$$

This is polynomial in the number of subsets (i.e.,$n^{V(\mathbb{C})-1}$) rather than exponential. Intuitively, this means that finite VC index implies that $\mathbb{C}$ has an apparent simplistic structure.

3. VC Sub-graph Bounds: A similar bound can be shown (for a different constant, same polynomial order) for the so-called VC subgraph classes. For a function

$$f: X \to \mathbb{R}$$

the sub-graph is a subset of $\chi \times \mathbb{R}$ such that

$$\{(x, t): t < f(x)\}$$

A collection of is called a VC sub-graph class if all the sub-graphs form a VC-class.

4. Covering Number for the $0 - 1$ Loss Function: Consider a set of indicator functions

$$I_C = \{1_C: C \in \mathbb{C}\}$$

in $L_1(\mathbb{Q})$ for the discrete empirical measure $\mathbb{Q}$ (or any type of probability measure $\mathbb{Q}$ - discrete or not). It can be shown that for any

$$r \geq 1$$

$$N\left(\varepsilon, I_C, L_r(\mathbb{Q})\right) \leq k\, V(\mathbb{C})(4e)^{V(\mathbb{C})}\left(\frac{1}{\varepsilon}\right)^{r(V(\mathbb{C})-1)}$$

5. Entropy Number for the Symmetric Convex Hypotheses Hull: Next consider the symmetric convex hull of a set $\mathfrak{I}_{CONVEX}$, which is a collection of functions of the form

$$\sum_{i=1}^{m} \alpha_i f_i$$

with

$$\sum_{i=1}^{m} |\alpha_i| \leq 1$$

Then

$$N\left[\varepsilon \sqrt{\int |\mathbb{F}|^2 d\mathbb{P}}, \mathfrak{I}_{CONVEX}, L_2(\mathbb{Q})\right] < C\left(\frac{1}{\varepsilon}\right)^{\frac{2V}{V+2}}$$

6. Convergence of the Convex Hull: The most important consequence of the above is that the power of $\frac{1}{\varepsilon}$ - $\frac{2V}{V+2}$ is strictly less than 2, which is just enough so that entropy integral is guaranteed to converge, therefore the class $\mathfrak{I}_{CONVEX}$ is going to be $\mathbb{P}$-Donsker.
7. VC Index of a Sub-graph Class: Any finite dimensional vector space $\mathfrak{I}$ of measurable functions

$$f: X \to \mathbb{R}$$

is a VC-subgraph of index smaller than or equal to $dim(\Im) + 2$.

8. Generalizations of VC Sub-graph: There are generalizations of the notion of VC subgraph class, e.g., there is the notion of pseudo-dimension (Bousquet, Boucheron, and Lugosi (2004)).

VC Classifier Framework

1. Introduction: Let χ be a feature space and

$$\Upsilon = \{0, 1\}$$

The function

$$f: X \to \Upsilon$$

is called the classifier. Similar to before, we define the shattering coefficient (also referred to as growth coefficient) as

$$S(\Im, \mathrm{n}) = \max_{x_1, \ldots, x_n} |\{f(x_1), f(x_2), \ldots, f(x_n)\}, f \in \Im|$$

2. Shattering Coefficient Relation: In terms of the previous section the shattering coefficient is precisely

$$\max_{x_1, \ldots, x_n} \Delta_n(\mathbb{C}, x_1, \ldots, x_n)$$

with $\mathbb{C}$ being a collection of all the sets as laid out above. Further, using Sauer's lemma, it can be shown that $S(\mathfrak{I}, \mathrm{n})$ is going to be a polynomial in n provided that class $\mathfrak{I}$ has a finite VC dimension, or equivalently, the collection $\mathbb{C}$ has a finite VC index.

3. VC Indicator Empirical Risk: Let

$$D_n = \{(X_1, Y_1), \dots, (X_n, Y_n)\}$$

be the data set. As always, we assume that P_{XY} is unknown, thus we work on bounding the $0-1$ indicator empirical risk:

$$\hat{R}_n(f) = \frac{1}{n}\sum_{i=1}^{n} I[f(X_n) \neq Y_n]$$

4. VC Inequality Bounds: For binary classification and $0-1$ loss function, we have the following generalization bounds:

   a.

$$\mathbb{P}\left(\sup_{f \in \mathfrak{I}} \left|\hat{R}_n(f) - R(f)\right| > \varepsilon\right) \leq 8\, S(\mathfrak{I}, \mathrm{n}) e^{-\frac{n\varepsilon^2}{32}}$$

   b.

$$\mathbb{E}\left(\sup_{f \in \mathfrak{I}} \left|\hat{R}_n(f) - R(f)\right|\right) \leq 2\sqrt{\frac{log S(\mathfrak{I}, \mathrm{n}) + log 2}{n}}$$

5. Implication of the VC Inequality Bounds: The impact of the VC inequality relation is that, as the sample size increases, provided $\mathfrak{I}$ has a finite VC dimension, the empirical $0-1$ risk becomes a good proxy for the expected $0-1$ risk. Note that both RHS of the two inequalities will converge to 0 provided $S(\mathfrak{I}, \mathrm{n})$ grows polynomially in n.
6. Connection between the Classifier Framework and the Empirical Process Framework: With the empirical classifier framework, on deals with the modified empirical process $\left|\hat{R}_n(f) - R(f)\right|_{\mathfrak{I}}$, but the other components are the same. As before the proof of the first VC

inequality relies on symmetrization, and then argue conditionally using the data with the concentration inequalities – in particular the Hoeffding's inequality (Bousquet, Boucheron, and Lugosi (2004)).

References

- Alekser, S. (1997): A Remark on the Szarek-Talagrand Theorem *Combinatorics, Probability, and Computing* **6** 139-144.
- Alon, N., S. Ben-David, N. Cesa-Bianchi, and D. Haussler (1997): Scale-sensitive Dimension, Uniform Convergence, and Learnability *Journal of the ACM* **44** 615-631.
- Bousquet, O., S. Boucheron, and G. Lugosi (2004): Introduction to Statistical Learning Theory *Advanced Lectures in Machine Learning Lecture Notes in Artificial Intelligence - Bousquet, von Luxburg, and Ratsch (editors)* **3176** 169-207.
- Cesa-Bianchi, N., and D. Haussler (1998): A Graph-Theoretic Generalization of the Sauer-Shelah Lemma *Discrete Applied Mathematics* **86** 27-35.
- Frankl, P. (1983): On the Trace of Finite Sets *Journal of Combinatorial Theory* **A34** 41-45.
- Haussler, D. (1995): Sphere-packing Numbers for the Boolean n-cube with bounded Vapnik-Chervonenkis Dimension *Journal of Combinatorial Theory* **A69** 217-232.
- Sauer, N. (1972): On the Density of Families of Sets *Journal of Combinatorial Theory (A)* **13** 145-147.
- Shelah, S. (1972): A Combinatorial Problem: Stability and Orders for Models and Theories in Infinitary Languages *Pacific Journal of Mathematics* **41** 247-261.
- Szarek, S., and M. Talagrand (1997): On the Convexified Sauer-Shelah Theorem *Journal of Combinatorial Theory* **B69** 183-192.
- Vapnik, V., and A. Chervonenkis (1971): On the Uniform Convergence of Relative Frequencies of Events to their Probabilities *Theory of Probability and its Applications* **16** 264-280.

Covering and Entropy Numbers

Motivation

1. Alternate Generalization Bounds: In the literature there exist many more capacity concepts, and they all assume the following form: with a probability of at least $1 - \delta$

$$R(f) \leq R_{emp}(f) + Capacity\ (\Im) + Confidence\ (\delta)$$

   In the simplest case, the capacity term only depends on $\Im$ and the confidence term on the underlying probability. Obviously, as is to be expected, these bounds are worst case bounds on *bad behavior*.
2. Shortcoming of the VC Measure: Of the three capacity measures seen so far – the VC dimension, growth function, and the VC entropy, all three are usually hard/impossible to compute. There are other measures which not only give sharper estimates, but also have properties that make their computation possible from data only.
3. Capacity/Complexity Measures: Some capacity/complexity measures are:
   a. Covering Numbers (all variants)
   b. VC Dimension
   c. Rademacher Complexity

Nomenclature - Normed Spaces

1. l_p^d Spaces: For

$$d \in \mathbb{N}$$

$\mathbb{R}^d$ denotes the d-dimensional space of vectors

$$\vec{x} = (x_1, \ldots, x_d)^{\mathrm{T}}$$

We define spaces l_p^d as follows: as vector spaces, they are identical to $\mathbb{R}^d$, in addition, they are endowed with p-norms; for

$$0 < p < \infty$$

$$\|\vec{x}\|_{l_p^d} \doteqdot \|\vec{x}\|_p = \left[\sum_{j=1}^{d} |x_j|^p\right]^{\frac{1}{p}}$$

and for

$$p = \infty$$

$$\|\vec{x}\|_{l_\infty^d} \doteqdot \|\vec{x}\|_\infty = \underset{j\ =\ 1, \ldots, d}{max} |x_j|$$

Note that a different normalization of the l_p^d norm is used in some papers in learning theory (e.g., Talagrand (1996)). For

$$0 < p \leq \infty$$

$$l_p \doteqdot l_p^\infty$$

2. $\underline{l_\infty^d \text{ of } f \in \mathcal{F} \text{ with respect to } \vec{X}_m}$: Given m points

$$\vec{x}_1, \ldots, \vec{x}_m \in l_p^d$$

we use the short-hand

$$\vec{X}_m = (\vec{x}_1, \ldots, \vec{x}_m)$$

Suppose $\mathcal{F}$ is a class of functions

$$f : \mathbb{R}^d \to \mathbb{R}$$

The l_∞^d norm with respect to $\vec{X}_m$ of

$$f \in \mathcal{F}$$

is defined as

$$\|f\|_{l_\infty^{\vec{X}_m}} \doteq \max_{i = 1, \ldots, m} |f(\vec{x}_i)|$$

Likewise

$$\|f\|_{l_p^{\vec{X}_m}} = \|[f(\vec{x}_1), \ldots, f(\vec{x}_m)]\|_{l_p^m}$$

3. Integral Norm over a σ-algebra Space: Given some set $\mathcal{X}$ with a σ-algebra, a measure μ on $\mathcal{X}$, some

$$0 < p < \infty$$

and a function

$$f: \mathcal{X} \to \mathbb{R}$$

we define

$$\|f\|_{\mathcal{L}_p(\mathcal{X},\mathbb{R})} \doteq \left[\int |f(x)|^p d\mu(x)\right]^{\frac{1}{p}}$$

if the integral exists, and

$$\|f\|_{\mathcal{L}_\infty(\mathcal{X},\mathbb{R})} \doteq ess \sup_{x \in \mathcal{X}} f(x)$$

For

$$0 < p < \infty$$

we let

$$\mathcal{L}_p(\mathcal{X},\mathbb{R}) \doteq \left\{f : \mathcal{X} \to \mathbb{R} : \|f\|_{\mathcal{L}_p(\mathcal{X},\mathbb{R})} < \infty\right\}$$

We also let

$$\mathcal{L}_p(\mathcal{X}) \doteq \mathcal{L}_p(\mathcal{X},\mathbb{R})$$

Covering, Entropy, and Dyadic Numbers

1. ϵ-Covering Numbers: If $\mathcal{S}$ is a set and d is a metric on $\mathcal{S}$, then the ϵ-covering number of

$$M \subset \mathcal{S}$$

with respect to the metric d denoted $\mathcal{N}(\epsilon, \mathcal{F}, d)$ is the smallest number of elements of an ϵ-cover for $\mathcal{F}$ using the metric d.

2. n^{th} Entropy Number of a Set $M \subset \mathcal{S}$: Given a metric space

$$\mathcal{E} = (\mathcal{S}, d)$$

we also write the covering number as $\mathcal{N}(\epsilon, \mathcal{F}, \mathcal{E})$. The n^{th} entropy number of a set

$$M \subset \mathcal{E}$$

for

$$n \in \mathbb{N}$$

is

$$\epsilon_n(M) \doteqdot \inf\{\epsilon > 0;\ \mathcal{N}(\epsilon, \mathcal{F}, \mathcal{E}) \leq n\}$$

3. Operator Norm: Let $\mathfrak{L}(\mathcal{E}, \mathcal{F})$ be the set of all bounded operators $\mathcal{T}$ between the normed spaces $(\mathcal{E}, \|\cdot\|_{\mathcal{E}})$ and $(\mathcal{F}, \|\cdot\|_{\mathcal{F}})$, i.e., operators such that the image of the closed unit ball

$$\mathcal{U}_{\mathcal{E}}\{x \in \mathcal{E} : \|x\|_{\mathcal{E}} \leq 1\}$$

is bounded. The smallest such bound is called the *Operator Norm*.
4. Entropy Numbers of an Operator: The *Entropy Numbers of an Operator*

$$\mathcal{T} \in \mathfrak{L}(\mathcal{E}, \mathcal{F})$$

are defined as

$$\epsilon_n(\mathcal{T}) = \epsilon_n\big(\mathcal{T}(\mathcal{U}_{\mathcal{E}})\big)$$

Note that

$$\epsilon_n(\mathcal{T}) = \|\mathcal{T}\|$$

and that $\epsilon_n(\mathcal{T})$ is certainly well-defined for all

$$n \in \mathbb{N}$$

if $\mathcal{T}$ is a compact operator, i.e., for any

$$\epsilon > 0$$

there exists a finite cover of $\mathcal{T}(\mathcal{U}_\mathcal{E})$ with open ϵ balls over $\mathcal{X}$.

5. Dyadic Entropy Numbers of an Operator: The dyadic entropy numbers of an operator are defined by

$$e_n(\mathcal{T}) = \epsilon_{2^{n-1}}(\mathcal{T})$$

$$n \in \mathbb{N}$$

Similarly, the dyadic entropy numbers of a set are defined from its entropy numbers. A very nice introduction to entropy numbers of operators is found in Carl and Stephani (1990).

6. Banach Spaces: $\mathcal{E}$ and $\mathcal{F}$ will always be Banach spaces - for instance, l_p^d spaces with

$$p \leq 1$$

In some cases they may be *Hilbert Spaces* $\mathcal{H}$, Banach spaces endowed with a dot-product $\langle .,. \rangle_\mathcal{H}$ giving rise to its norm via

$$\|\vec{x}\|_\mathcal{H} = \sqrt{\langle \vec{x}, \vec{x} \rangle_\mathcal{H}}$$

Background and Overview of Basic Results

1. Setup: We start by endowing a function class $\mathfrak{I}$ with the following random metric:

$$d_n(f, f') = \frac{1}{n}|\{f(\mathcal{Z}_i) - f'(\mathcal{Z}_i); i = 1, \dots, n\}|$$

This is the normalized Hamming distance of the projections on the sample. Given such a metric, we say that $f_1, \dots, f_n$ *covers* $\mathfrak{I}$ at a radius ε if

$$\mathfrak{I} \subset \bigcup_{i=1}^{N} \mathcal{B}(f_i, \varepsilon)$$

The term *cover* is used analogous to the term *space* in function space.

2. Definition: The covering number of $\mathfrak{I}$ at a radius ε with respect to d_n, denoted by $\mathcal{N}(\mathfrak{I}, \varepsilon, N)$, is the minimum size of a cover of radius ε. Note that it does not matter if we apply this definition to the original class $\mathfrak{I}$ or the loss class $\mathcal{L}$, since

$$\mathcal{N}(\mathfrak{I}, \varepsilon, N) = \mathcal{N}(\mathcal{L}, \varepsilon, N)$$

4. Growth of the Covering Function: The covering numbers characterize the size of the function class measured by the metric d_n. The rate of growth of the logarithm of $\mathcal{N}(\mathfrak{I}, \varepsilon, N)$, called the metric entropy, is related to the classical concept of the vector dimension. Indeed, if $\mathfrak{I}$ is a compact set in a d-dimensional Euclidean space

$$\mathcal{N}(\mathfrak{I}, \varepsilon, N) \approx \varepsilon^{-d}$$

5. Finite Covering Numbers: When the covering numbers are finite, it is possible to approximate the class $\mathfrak{I}$ by a finite set of functions (which cover $\mathfrak{I}$). This again allows the use of finite union bound, provided we can relate the behavior of all functions in $\mathfrak{I}$ to that of the functions in the cover.
6. Finite Covering Number Sample Bound:

$$\mathbb{P}[\exists\, f \in \Im : R(f) - R_n(f) > \varepsilon] \leq 8\mathbb{E}[\mathcal{N}(\Im, \varepsilon, n)]e^{-\frac{n\varepsilon^2}{128}}$$

7. Covering Numbers for Real Valued Functions: Covering Numbers can also be defined for real-valued functions – see treatment below.
8. Covering Numbers and the VC Dimension: Notice that, if the functions in $\Im$ can only take 2 values, for all

$$\varepsilon > 0$$

$$\mathcal{N}(\Im, \varepsilon, n) \leq \left|\Im_{\mathbb{Z}_1^n}\right| = \mathcal{N}(\Im, \mathbb{Z}_1^n)$$

Hence the VC entropy corresponds to the log covering number at the minimal scale, which implies

$$\mathcal{N}(\Im, \varepsilon, n) \leq h \log \frac{en}{h}$$

but one can have a considerably better result.

9. Haussler's Bound: Let $\Im$ be a class of VC dimension h. Then for all

$$\varepsilon > 0$$

all n, and any sample

$$\mathcal{N}(\Im, \varepsilon, n) \approx Ch(4e)^h \varepsilon^{-h}$$

The interesting aspect of this bound is that it does not depend on the sample size n.

10. Principle behind Covering Bound Improvement: The covering bound is a generalization of the VC entropy bound where the scale is adapted to the error. The result can be considerably improved by considering all scales.

References

- Carl, B., and I. Stephani (1990): *Entropy, Compactness, and the Approximation of Operators* **Cambridge University Press** Cambridge UK.
- Talagrand, M. (1996): The Glivenko-Cantelli Problem, 10 years later *Journal of Theoretical Probability* **9 (2)** 371-384.

Covering Numbers for Real-Valued Function Classes

Introduction

1. Literature: Covering numbers for functions have been extensively studied in a variety of literature dating way back to Kolmogorov (1956), Kolmogorov and Tihomirov (1961)). They play a central role in a number of areas of information theory and statistics, including density estimation, empirical processes, and machine learning (Pollard (1984), Birge (1987), Haussler (1992)).
2. Definition: Let $\mathcal{F}$ be a sub-set of a metric space $(\mathcal{X}, \rho)$. For a given

$$\epsilon > 0$$

   the metric covering number $\mathcal{N}(\epsilon, \mathcal{F}, \rho)$ is defined as the smallest number of sets/balls of radius ϵ whose union contains $\mathcal{F}$. In what follows, ρ is omitted if the context is clear.
3. Functions of Bounded Variation under the $\mathcal{L}_1$ Metric: Here our intention is to find bounds on the covering numbers of functions of bounded variations under the $\mathcal{L}_1$ metric. Specifically, let $\mathcal{F}_1$ be the set of all functions on $[0, T]$ taking values in $\left[-\frac{V}{2}, +\frac{V}{2}\right]$ with total variation of at most

$$V > 0$$

   It is natural to use the same V for both the range and the variation, since a bound on the variation of a function implies a bound on its range.
4. Tight Bounds on $\mathcal{N}(\epsilon, \mathcal{F}_1, \mathcal{L}_1)$: Our goal is to find tight bounds on $\mathcal{N}(\epsilon, \mathcal{F}_1, \mathcal{L}_1)$ in terms of the relevant constants. This is related to the problem of density estimation, where attention has been given to the problem of finding covering numbers for classes of densities that are unimodal or non-decreasing (Groeneboom (1986), Birge (1987)).

5. Density Estimation using Covering Numbers under Constraints: Treatment here is a super-set of the treatment in Birge (1987), since we do not impose a density constraint on the class, which accounts for the difference in function behavior as a function of the parameters. In fact, it is this density constraint that accounts for the $\log VT$ constant in Birge (1987) rather than VT that we obtain here.
6. Covering Numbers for General Classes of Real-valued Functions: Here we also investigate the metric covering numbers for the general classes of real-valued functions under the family of $\mathcal{L}_1(dP)$ metrics, where P is a probability distribution. Upper bounds in terms of the Vapnik-Chervonenkis dimension or the pseudo-dimension of the function class were first obtained by Dudley (1978), improved in Pollard (1984), and further by Haussler (1992, 1995). Various lower bounds have also been obtained (e.g., Kulkarni, Mitter, and Tsitsiklis (1993)).
7. Scale-Sensitive Covering Number Bounds: The importance of scale-sensitive versions of several combinatorial dimensions in learning problems may be seen from Alon, Ben-David, Cesa-Bianchi, and Haussler (1993), and Bartlett and Long (1995). Using techniques due to Haussler and results from Alon, Ben-David, Cesa-Bianchi, and Haussler (1993), Lee, Bartlett, and Williamson (1995) proved an upper bound on

$$\max_P \log \mathcal{N}\left(\epsilon, \mathcal{F}, \mathcal{L}_1(dP)\right)$$

in terms of the scale-sensitive dimension of the function class. The treatment here improves the result and provides a lower bound. As will be shown, in general the bounds presented here cannot be significantly improved.

Functions of Bounded Variation

1. Upper/Lower Bounds - Statement: For all

$$\epsilon \leq \frac{VT}{12}$$

$$\frac{VT}{54\epsilon} \leq \log \mathcal{N}(\epsilon, \mathcal{F}_1, \mathcal{L}_1) \leq \frac{12VT}{\epsilon} \log 2$$

(Bartlett, Kulkarni, and Posner (1997)).

2. Special Case of the $\mathcal{L}_\infty$ Metric: Certainly, under the $\mathcal{L}_\infty$ metric, the classes of functions of bounded variation (or even the subset of functions under this class that are continuous) are not compact, hence

$$\mathcal{N}(\epsilon, \mathcal{F}_1, \mathcal{L}_\infty) = \infty$$

However, it has been established that the class of function of bounded variation that are also Lipschitz-smooth have covering numbers of the order of $\left(\frac{1}{\epsilon}\right)^{\frac{1}{\epsilon}}$ in the $\mathcal{L}_\infty$ metric (Lorentz (1966)).

3. Function Bound vs. Class Bound: Let $\mathcal{F}_2$ be all functions of variation at most V that map from $[0, T]$ to $[-B, +B]$ for some

$$B > \frac{V}{2}$$

The theorem above can be extended to $\mathcal{F}_2$ as

$$\frac{VT}{54\epsilon} \log 2 + \frac{eBT}{6\epsilon} \leq \log \mathcal{N}(\epsilon, \mathcal{F}_2, \mathcal{L}_1) \leq \frac{18VT}{\epsilon} \log 2 + \frac{3(2B - V)T}{8\epsilon}$$

(Bartlett, Kulkarni, and Posner (1997)). Both upper and lower bounds are obtained by considering vertical shifts in $\mathcal{F}_1$ and using the proof approach outlined below.

4. Distribution-Dependent Bounds: The upper bound (with T set to 1) can be extended to give the upper bound

$$\log \mathcal{N}\left(\epsilon, \mathcal{F}_1, \mathcal{L}_1(dP)\right) \leq \frac{12V}{\epsilon}$$

(Bartlett, Kulkarni, and Posner (1997)), i.e., a uniform bound on the covering numbers for all weighted $\mathcal{L}_1$ norms where P is an arbitrary probability distribution in $[0, T]$. The proof for this is obtained by modifying the proof presented below for an equi-probable partition of $[0, T]$ rather than a partition of equal size.

Functions of Bounded Variation – Upper Bound Proof

1. $\frac{\epsilon}{2}$-Covering of the Function Class: Define $\mathcal{I}$ to be the class of all non-decreasing functions on $[0, T]$ taking values in $[0, V]$. Let $\mathcal{G}\left(\frac{\epsilon}{2}\right)$ be any $\frac{\epsilon}{2}$-covering of $\mathcal{I}$. It is well-known that for any

$$f \in \mathcal{F}_1$$

there exist

$$g, h \in \mathcal{I}$$

such that

$$f = g - h$$

(e.g., Kolmogorov and Fomin (1970)).

2. Range of the $\frac{\epsilon}{2}$-cover Class: More precisely, if $v(x)$ denotes the total variation over the interval $[0, x]$, then g and h can be taken to be

$$g(x) = \frac{v(x) + f(x)}{2} + \frac{V}{4}$$

and

$$h(x) = \frac{v(x) - f(x)}{2} + \frac{V}{4}$$

It is easy to show that both g and h are non-decreasing. Also, if f takes values in $\left[-\frac{V}{2}, +\frac{V}{2}\right]$, and has total variation bounded by V, then both g and h take values only in the range $[0, V]$.

3. Metric Covering Number of the $\frac{\epsilon}{2}$-cover Set: By then definition of the cover of a set, there exist

$$\phi_1, \phi_2 \in \mathcal{G}\left(\frac{\epsilon}{2}\right)$$

such that

$$\|g - \phi_1\|_{\mathcal{L}_1} \leq \frac{\epsilon}{2}$$

$$\|h - \phi_2\|_{\mathcal{L}_1} \leq \frac{\epsilon}{2}$$

This gives

$$\|f - (\phi_1 - \phi_2)\|_{\mathcal{L}_1} = \|g - h - (\phi_1 - \phi_2)\|_{\mathcal{L}_1} \leq \|g - \phi_1\|_{\mathcal{L}_1} + \|h - \phi_2\|_{\mathcal{L}_1} \leq \epsilon$$

Thus, we can produce ϵ-covering of $\mathcal{F}_1$ by taking all pairs from

$$\mathcal{G}\left(\frac{\epsilon}{2}\right) \times \mathcal{G}\left(\frac{\epsilon}{2}\right)$$

Hence

$$\mathcal{N}(\epsilon, \mathcal{F}_1, \mathcal{L}_1) \leq \left[\mathcal{N}\left(\frac{\epsilon}{2}, \mathcal{I}, \mathcal{L}_1\right)\right]^2$$

4. Cardinality of the Equi-Partition Set: We now find an upper bound on $\mathcal{N}(\epsilon, \mathcal{I}, \mathcal{L}_1)$ by producing an ϵ-covering of $\mathcal{I}$. Partition $[0, T]$ into

$$n_1 = \frac{T}{h_1} \geq 1$$

sub-intervals of length h_1, i.e.,

$$[0, h_1), [h_1, 2h_1), \dots, , [(n_1 - 1)h_1, 1)$$

Let $\Phi(n_1, n_2)$ be the set of all functions that are constant on these sub-intervals, non-decreasing, and taking values only in the set

$$\left\{\left(j - \frac{1}{2}\right) h_2 | j = 1, \dots, n_2\right\}$$

where

$$h_2 = \frac{V}{n_2}$$

It is easy to see that cardinality of $\Phi(n_1, n_2)$, denoted by $|\Phi(n_1, n_2)|$, satisfies

$$|\Phi(n_1, n_2)| = \binom{n_1 + n_2}{n_1} \leq 2^{n_1 + n_2}$$

5. Metric Covering Number - Upper Bound: Now, note that for any

$$f \in \mathcal{I}$$

there exists a

$$\phi \in \Phi(n_1, n_2)$$

such that

$$\|f - \phi\|_{\mathcal{L}_1} \leq h_1 V + \frac{h_2 T}{2}$$

To see this, consider the error to the best constant approximation to f on a sub-interval, and the additional error introduced by quantizing the range. Choosing

$$h_1 = \frac{\epsilon}{2V}$$

and

$$h_2 = \frac{\epsilon}{T}$$

gives

$$\|f - \phi\|_{\mathcal{L}_1} \leq \epsilon$$

Hence, with this choice we get

$$\mathcal{N}(\epsilon, \mathcal{I}, \mathcal{L}_1) \leq |\Phi(n_1, n_2)| \leq 2^{n_1 + n_2} = 2^{\frac{3VT}{\epsilon}}$$

for

$$\epsilon < VT$$

By using the ϵ-covering bound of $\mathcal{F}_1$ from $\mathcal{G}\left(\frac{\epsilon}{2}\right)$, we get

$$\mathcal{N}(\epsilon, \mathcal{F}_1, \mathcal{L}_1) \leq \left[\mathcal{N}\left(\frac{\epsilon}{2}, \mathcal{I}, \mathcal{L}_1\right)\right]^2 \leq 2^{\frac{12VT}{\epsilon}}$$

Functions of Bounded Variation – Lower Bound Proof

1. Partition of $[0, T]$: Partition $[0, T]$ into n segments. Take the set Γ of all the binary $\{0, h\}$-valued functions constant over each segment. If

$$n \geq 2$$

the bounded variation and the boundedness constraints impose

$$h \leq \frac{V}{n}$$

2. The Difference Function: There are 2^n functions in the set Γ. For any two functions

$$\gamma_i, \gamma_j \in \Gamma$$

define $d(i, j)$ as the number of segments on which the 2 functions have different values. It is then easy to see that

$$\left\|\gamma_i - \gamma_j\right\|_{\mathcal{L}_1} = d(i, j)\frac{hT}{n}$$

3. Bounding the Differences Function: Thus

$$\gamma_i, \gamma_j \in \Gamma$$

are close if

$$d(i,j) \leq \frac{n\epsilon}{hT}$$

For an arbitrary

$$\gamma \in \Gamma$$

let $C(\epsilon)$ be the number of functions in Γ that are ϵ-close to Γ. Then

$$C(\epsilon) = \sum_{l=0}^{[\![\frac{n\epsilon}{hT}]\!]} \binom{n}{l}$$

where $[\![\alpha]\!]$ denotes the largest number no bigger than α. The Chernoff-Okamoto inequality (Dudley (1978)) is

$$\sum_{l=0}^{m} \binom{n}{l} p^l (1-p)^{n-l} \leq e^{-\frac{(np-m)^2}{2np(1-p)}}$$

for

$$p \leq \frac{1}{2}$$

and

$$m \leq np$$

Letting

$$p = \frac{1}{2}$$

we get that

$$\sum_{l=0}^{\left[\left[\frac{n\epsilon}{hT}\right]\right]} \binom{n}{l} \leq 2^n e^{-\frac{2\left(\frac{n}{2}-\left[\left[\frac{n\epsilon}{hT}\right]\right]\right)^2}{n}} \leq 2^n e^{-\frac{n}{2}\left(1-\frac{2\epsilon}{hT}\right)^2}$$

The same result can also be obtained by using Hoeffding's inequality.

4. Optimal Lower Bound: Since the cardinality of Γ is 2^n, it is clear that we need at least $\frac{2^n}{C(\epsilon)}$ functions for a ϵ-cover, i.e.

$$\mathcal{N}(\epsilon, \mathcal{F}_1, \mathcal{L}_1) \geq \frac{2^n}{C(\epsilon)} \geq e^{-\frac{n}{2}\left(1-\frac{2\epsilon}{hT}\right)^2}$$

To obtain a large lower bound we want to maximize this expression subject to

$$h \leq \frac{V}{n}$$

Equivalently

$$\max_{n \geq 2, h \leq \frac{V}{n}} n\left(1-\frac{2\epsilon}{hT}\right)^2 \geq \max_{n \geq 2} \left(1-\frac{2\epsilon n}{VT}\right)^2$$

If

$$\epsilon \leq \frac{VT}{12}$$

we may choose

$$n = \left[\!\left[\frac{VT}{6\epsilon}\right]\!\right]$$

to show that this maximum is at least

$$\frac{2VT}{27\epsilon} - \frac{4}{9} \geq \frac{VT}{27\epsilon}$$

Thus, the lower bound on the covering number is

$$\mathcal{N}(\epsilon, \mathcal{F}_1, \mathcal{L}_1) \geq e^{\frac{VT}{54\epsilon}} \; \forall \; \epsilon \leq \frac{VT}{12}$$

General Function Classes

1. Introduction: Here we provide lower and upper bounds on the quantity

$$\max_P \log \mathcal{N}\left(\epsilon, \mathcal{F}, \mathcal{L}_1(dP)\right)$$

for the general classes $\mathcal{F}$ of $\{0, 1\}$-valued functions defined on the set X, where the max is taken over all the probability distributions P on X. The bounds are given in terms of the scale-sensitive dimension $\mathcal{F}$ (Alon, Ben-David, Cesa-Bianchi, and Haussler (1993)).

2. Fat-Shattering Coefficient: Define

$$fat_{\mathcal{F}}(\epsilon) = max\{n: some\ x \in X^n\ is\ \epsilon - shattered\ by\ \mathcal{F}\}$$

where a sequence

$$x \in X^n$$

is ϵ-shattered by $\mathcal{F}$ if there is a sequence

$$r \in [0,1]^n$$

such that, for all

$$b \in [0,1]^n$$

there is an

$$f \in \mathcal{F}$$

with

$$f(x_i) = \begin{cases} \geq r_i + \epsilon & if\ b_i = 1 \\ \leq r_i - \epsilon & otherwise \end{cases}$$

For example, it is easy to show that, bounded variation functions $\mathcal{F}_1$ have

$$fat_{\mathcal{F}_1}(\epsilon) = \left[\!\left[\frac{V}{2\epsilon}\right]\!\right]$$

To see the upper bound, consider an ϵ-shattered sequence, and for a suitable sequence r, find the sequence b for which the corresponding function has maximum variation. An easy construction with

$$r = 0$$

gives the lower bound.

3. Upper/Lower Bounds - Statement: There are constants c_1 and c_2 such that, for any permissible (i.e., benign measurability condition (Pollard (1984))) class $\mathcal{F}$ of $\{0,1\}$-valued functions defined on a set X

$$\frac{fat_{\mathcal{F}}(4\epsilon)}{32} \leq \max_{P} \log \mathcal{N}\left(\epsilon, \mathcal{F}, \mathcal{L}_1(dP)\right) \leq c_1 fat_{\mathcal{F}}(c_2\epsilon)\left[log_2 \frac{1}{\epsilon}\right]^2$$

(Bartlett, Kulkarni, and Posner (1997)).

4. Gap Between Upper and Lower Bounds: There is a gap between the lower and the upper bounds. The class $\mathcal{F}_1$ of bounded variation functions shows that the lower bound in tight within a constant factor. The following example shows that some gap between the lower and the upper bounds is essential. For any positive integers n, d, let $\mathcal{F}_{d,n}$ be the class of all functions from $\{1, 2, \ldots, d\}$ to $\left\{0, \frac{1}{n}, \ldots, 1\right\}$, and let P be the uniform distribution on $\{1, 2, \ldots, d\}$. Then

$$fat_{\mathcal{F}_{d,n}}(\epsilon) = d$$

for

$$\epsilon \leq \frac{1}{2}$$

yet for

$$\epsilon = \frac{1}{2nd}$$

we have

$$\log \mathcal{N}\left(\epsilon, \mathcal{F}_{d,n}, \mathcal{L}_1(dP)\right) > dlog_2 \frac{1}{2d\epsilon}$$

5. Function Range Scaling: Clearly, the statement above can be trivially extended to classes of functions that map to an arbitrary interval $[a, b]$ by scaling ϵ by a factor $b - a$.

6. Generalization of the Statement: Cesa-Bianchi and Haussler (1993) proved a version of the above statement in a considerably more general setting. Their results provide bounds on covering numbers of a class of functions that take values in an arbitrary totally bounded metric space, in terms of a scale-sensitive dimension of the class. In the special case of real-valued functions, their scale-sensitive dimension is different from that considered here, although the lemmas in Anthony and Bartlett (1995) show that the two quantities are within log factors of each other.

General Function Class Bounds – Proof Sketch

1. Lemma #1 of 3 - Bounds on $fat_{\mathcal{F}}$: In the first step, we show that a bound on $fat_{\mathcal{F}}$ implies that, for any finite sequence

$$x = (x_1, \ldots, x_m)$$

from X, there is a small subset of $\mathcal{F}$ that is a cover of the restriction of $\mathcal{F}$ to x, denoted

$$\mathcal{F}_{|x} = \{[f(x_1), \ldots, f(x_m)] : f \in \mathcal{F}\} \subseteq \mathcal{R}^m$$

In defining the cover and the covering numbers $\mathcal{N}\left(\epsilon, \mathcal{F}_{|x}\right)$, we use the scaled $\mathbb{l}_1$ metric on $\mathcal{R}^m$ defined by

$$\rho(a, b) = \frac{1}{m}\sum_{i=1}^{m}|a_i - b_i|$$

We can also consider $\mathcal{F}_{|x}$ as a class of functions from $\{1, \ldots, m\}$ to $\mathcal{R}$; this is how we define $fat_{\mathcal{F}_{|x}}$.

2. Lemma #2 of 3 - Intra-Class Cover: In the second step, we use the results of the first lemma to deduce that there is a small cover for the set of absolute differences between the functions in $fat_{\mathcal{F}_{|x}}$.
3. Lemma #3 of 3 - Uniform Convergence: The third lemma implies that the above 2 indicate a uniform convergence result for this set. We use this result to show that, for some sequences of positive probability P, the cover for the restriction of $\mathcal{F}$ to the sequence induces a cover for $\mathcal{F}$.

General Function Class Bounds – Lemmas

1. Lemma #1 - Statement: Suppose

$$x \in X^m$$

and $\mathcal{F}$ is a set of $\{0,1\}$-valued functions on X. Let

$$d = fat_{\mathcal{F}_{|x}}\left(\frac{\epsilon}{4}\right)$$

where

$$\epsilon > 0$$

If

$$n \geq 2dlog_2 \frac{64e^2}{\epsilon \log 2}$$

there is a subset $\mathcal{T}$ of $\mathcal{F}$ for which $\mathcal{T}_{|x}$ is an ϵ-cover of $\mathcal{F}_{|x}$, and

$$|\mathcal{T}| \leq 2\left(\frac{16}{\epsilon}\right)^{6d\log_2\frac{32en}{d\epsilon}}$$

2. Lemma #1 - Proof: This lemma is implicit in Bartlett and Long (1995), which provides a slightly better bound - by a factor of $\log d$ - than that given in Alon, Ben-David, Cesa-Bianchi, and Haussler (1993).
3. Lemma #2 - Statement: For a class $\mathcal{F}$ of $\{0,1\}$-valued functions, let

   $$|\mathcal{F}-\mathcal{F}| = \{|f_1 - f_2| : f_1, f_2 \in \mathcal{F}\}$$

   Then, with the metric $\mathcal{L}_1(dP)$ on $\mathcal{F}$

   $$\mathcal{N}\left(\epsilon, |\mathcal{F}-\mathcal{F}|, \mathcal{L}_1(dP)\right) \leq \left[\mathcal{N}\left(\frac{\epsilon}{2}, \mathcal{F}, \mathcal{L}_1(dP)\right)\right]^2$$

4. Lemma #2 - Proof: Take an $\frac{\epsilon}{2}$-cover $\mathcal{T}$ for $\mathcal{F}$. Then for all

   $$f_1, f_2 \in \mathcal{F}$$

   pick

   $$t_1, t_2 \in \mathcal{T}$$

   within $\frac{\epsilon}{2}$ of f_1 and f_2, respectively. We have

   $$\| |f_1 - f_2| - |t_1 - t_2| \|_{\mathcal{L}_1(dP)} \leq \| |f_1 - t_1| + |f_2 - t_2| \|_{\mathcal{L}_1(dP)} \leq \epsilon$$

   It follows that

   $$\{|t_1 - t_2| : t_1, t_2 \in \mathcal{T}\}$$

is an ϵ-cover for $\mathcal{F}$.

5. Lemma #3 – Statement: For a permissible class $\mathcal{G}$ of $\{0,1\}$-valued functions on a set X, a probability distribution P on X, and an

$$\epsilon > 0$$

$$P_m\left\{x \in X^m : \exists\, g \in \mathcal{G}, \left|\frac{1}{m}\sum_{i=1}^{m} g(x_i) - \mathbb{E}[g]\right| > \epsilon\right\} \leq 4 \max_{x \in X^m} \mathcal{N}\left(\frac{\epsilon}{16}, \mathcal{G}_{|x}\right) e^{-\frac{m\epsilon^2}{128}}$$

6. Lemma #3 - Consequence: This lemma, which provides the uniform convergence property, is due to Pollard (1984). Haussler (1992) provides a related result with different constants. The class $\mathcal{G}$ that we will consider is $|\mathcal{F} - \mathcal{F}|$.

General Function Class – Upper Bounds

1. Reduction of the Probability Bound: We start by establishing the following chain of inequalities:
   a.

$$P_m\left\{x \in X^m : \exists\, f_1, f_2 \in \mathcal{F}, \left|\frac{1}{m}\sum_{i=1}^{m} |f_1(x_i) - f_2(x_i)| - \mathbb{E}[|f_1 - f_2|]\right| > \frac{\epsilon}{2}\right\}$$

   b. Use Lemma #3 to get

$$P_m\left\{x \in X^m : \exists\, f_1, f_2 \in \mathcal{F}, \left|\frac{1}{m}\sum_{i=1}^{m} |f_1(x_i) - f_2(x_i)| - \mathbb{E}[|f_1 - f_2|]\right| > \frac{\epsilon}{2}\right\}$$
$$\leq 4 \max_{x \in X^m} \mathcal{N}\left(\frac{\epsilon}{32}, |\mathcal{F} - \mathcal{F}|_{|x}\right) e^{-\frac{m\epsilon^2}{128\times 4}}$$

c. Use Lemma #2 to get

$$P_m\left\{x \in X^m : \exists\, f_1, f_2 \in \mathcal{F}, \left|\frac{1}{m}\sum_{i=1}^{m}|f_1(x_i) - f_2(x_i)| - \mathbb{E}[|f_1 - f_2|]\right| > \frac{\epsilon}{2}\right\}$$
$$\leq 4 \max_{x \in X^m} \mathcal{N}\left(\frac{\epsilon}{64}, \mathcal{F}_{|x}\right) e^{-\frac{m\epsilon^2}{512}}$$

d. Finally use Lemma #1 for

$$d = fat_{\mathcal{F}_{|x}}\left(\frac{\epsilon}{256}\right)$$

and

$$m \geq 2d\log_2 \frac{64e^2}{\epsilon \log 2}$$

to get

$$P_m\left\{x \in X^m : \exists\, f_1, f_2 \in \mathcal{F}, \left|\frac{1}{m}\sum_{i=1}^{m}|f_1(x_i) - f_2(x_i)| - \mathbb{E}[|f_1 - f_2|]\right| > \frac{\epsilon}{2}\right\}$$
$$\leq 16\left(\frac{1024}{\epsilon}\right)^{12d\log_2\frac{2048me}{\epsilon d}} e^{-\frac{m\epsilon^2}{512}}$$

2. Sample Size Bounds: It is easy to show that the probability P_m is less than 1 for

$$m \geq \frac{k_1 d}{\epsilon^3}$$

where k_1 is a constant. In fact, a more detailed estimation will show that

$$m \geq \frac{k_1 d}{\epsilon^2}\left[\log\log\frac{1}{\epsilon}\right]$$

will be sufficient.

3. Cover for the Restriction of $\mathcal{F}$: For the above sample size, it follows that there is a

$$x \in X^m$$

such that any

$$\mathcal{T} \subseteq \mathcal{F}$$

for which $\mathcal{T}_{|x}$ is an $\frac{\epsilon}{2}$–cover for $\mathcal{F}_{|x}$ is also an ϵ-cover for $\mathcal{F}$. To see this, notice that for all

$$f \in \mathcal{F}$$

there is a

$$t \in \mathcal{T}$$

with

$$\frac{1}{m}\sum_{i=1}^{m}|t(x_i) - f(x_i)| \leq \frac{\epsilon}{2}$$

and if x is chosen in the complement of the set that generates P_m above, then

$$\left|\frac{1}{m}\sum_{i=1}^{m}|t(x_i) - f(x_i)| - \mathbb{E}[|t - f|]\right| \leq \frac{\epsilon}{2}$$

so that

$$\mathbb{E}[|t - f|] \leq \frac{\epsilon}{2}$$

In fact, for sufficiently large m, a proper cover for the restriction of $\mathcal{F}$ to almost any m-sequence induces a cover of $\mathcal{F}$.

4. The Upper Bound - Final Form: Together with Lemma #1, the above statement shows that some

$$\mathcal{T} \subseteq \mathcal{F}$$

satisfies

$$log_2 \mathcal{N}\left(\epsilon, \mathcal{F}, \mathcal{L}_1(dP)\right) \leq log_2 |\mathcal{T}| \leq 1 + c_1 fat_{\mathcal{F}}(c_2 \epsilon) \left[log_2 \frac{1}{\epsilon}\right]^2$$

from which the result follows.

General Function Class – Lower Bounds

1. The Lower Bound: Suppose

$$fat_{\mathcal{F}}(4\epsilon) \geq d$$

Then consider the uniform distribution on a 4ϵ-shattered set of size d, and consider the restriction of the class $\mathcal{F}$ to this set. The same argument as the proof for the lower bound for the functions of bounded variation gives

$$\mathcal{N}\left(\epsilon, \mathcal{F}, \mathcal{L}_1(dP)\right) \geq e^{\frac{d}{32}}$$

References

- Alon, N., S. Ben-David, N. Cesa-Bianchi, and D. Haussler (1993): Scale-sensitive Dimensions, Uniform Convergence, and Learnability *Proceedings of the ACM Symposium on Foundations of Computer Science*.
- Anthony, M., and P. L. Bartlett (1995): Function Learning from Interpolation *Computational Learning Theory: Proceedings of the 2nd European Conference, EuroCOLT '95* 211-221 **Springer**.
- Anthony, M. (1997): Probabilistic Analysis of Learning in Artificial Neural Networks: The PAC Model and its Variants *Neural Computational Survey* **1** 1-47.
- Bartlett, P. L., and P. Long (1995): More Theorems about Scale-sensitive Dimensions and Learnability *Proceedings of the 8th Annual Conference on Computational Learning Theory* 392-401 **ACM Press**.
- Bartlett, P. L., S. R. Kulkarni, and S. E. Posner (1997): Covering Numbers for Real-Valued Function Classes *IEEE Transactions on Information Theory* **43 (5)** 1721-1724.
- Birge, L. (1987): Estimating a Density under Order Restrictions: Non-asymptotic Minimax Risk *Annals of Statistics* **15** 995-1012.
- Cesa-Bianchi, N., and D. Haussler (1996): A Graph-theoretic Generalization of the Sauer-Shelah Lemma *Internal Report 170 96* **DSI Universita di Milano**.
- Dudley, R. M. (1978): Central Limit Theorems for Empirical Measures *Annals of Probability* **6 (6)** 899-929.
- Groeneboom, P. (1986): *Some Current Developments in Density Estimation* **CWI Monographs** North-Holland.
- Haussler, D. (1992): Decision-theoretic Generalizations of the PAC Model for Neural Net and other Learning Applications *Information and Computation* **100** 78-150.
- Haussler, D. (1995): Sphere Packing Numbers for the Subsets of the Boolean n-cube with Bounded Vapnik-Chervonenkis Dimension *Journal of Combinatorial Theory A* **69 (2)** 217.
- Kolmogorov, A. N. (1956): Asymptotic Characteristics of some Completely Bounded Metric Spaces *Dokl. Acad. Nauk. SSSR* **108** 585-589.

- Kolmogorov, A. N., and V. M. Tihomirov (1961): ϵ-entropy and ϵ -capacity of Sets in Function Spaces *Amer. Math. Soc. Transl. (2)* **17** 277-364.
- Kolmogorov, A. N., and S. V. Fomin (1970): *Introductory Real Analysis* **Dover** New York.
- Kulkarni, S. R., S. K. Mitter, and J. N. Tsitsiklis (1993): Active Learning using Binary-valued Queries *Machine Learning* **11** 23-35.
- Lee, W. S., P. L. Bartlett, and R. C. Williamson (1995): On Efficient Agnostic Learning of Linear Combinations of Basis Functions *Proceedings of the 8th Annual Conference on Computational Learning Theory* 369-376 **ACM Press**.
- Lorentz, G. G. (1966): Metric Entropy and Approximation *Bull. Amer. Math. Soc.* **72** 903-937.
- Pollard, D. (1984): *Convergence of Stochastic Processes* **Springer** New York.

Operator Theory Methods for Entropy Numbers

Introduction and Setup

1. Classical Statistical Theory Viewpoint: Under the traditional viewpoint of the statistical learning theory, one is given a class of functions $\mathcal{F}$, and the generalization performance attainable using $\mathcal{F}$ is determined via the covering numbers of $\mathcal{F}$.
2. Uniform Covering Numbers of $\mathcal{F}$: More precisely, for some set $\mathcal{X}$, and

$$\vec{x}_i \in \mathcal{X}$$

for

$$i = 1, \ldots, m$$

define *Uniform Covering Numbers* of the function class $\mathcal{F}$ on $\mathcal{X}$ by

$$\mathcal{N}_m(\epsilon, \mathcal{F}) \doteq \sup_{\vec{x}_1, \ldots, \vec{x}_m \in \mathcal{X}} \mathcal{N}_m\left(\epsilon, \mathcal{F}, l_\infty^{\vec{X}_m}\right)$$

where $\mathcal{N}_m(\epsilon, \mathcal{F})$ is the ϵ-covering number of $\mathcal{F}$ with respect to $l_\infty^{\vec{X}_m}$. Recall

$$\vec{X}_m = (\vec{x}_1, \ldots, \vec{x}_m)$$

Many generalization bounds can be expressed in terms of $\mathcal{N}_m(\epsilon, \mathcal{F})$.

3. Combinatorial Dimension Bounding of Entropy Numbers: Traditionally, lemmas such as the Sauer lemma have been used in function learning to extract dimensions such as VC-

dimension of real-valued functions, a variation due to Pollard called pseudo-dimension, or a scale-sensitive dimension thereof.

4. Bounding Covering Numbers: These dimensions reduce the computation of $\mathcal{N}_m(\epsilon, \mathcal{F})$ to that of a single "dimension" like quantity independent of m. An overview of these various dimensions, details of their history, and examples of their computation can be found in Anthony (1997), and Anthony and Bartlett (1999).
5. Information Theory Viewpoint:
    a. The Kernel Operator => An alternate view is that of a function class $\mathcal{F}$ being induced by an operator T_k that depends on some kernel function k - thus $\mathcal{F}$ is the image of a base class $\mathcal{G}$ under T_k.
    b. Covering Number in terms of T_k Properties => Thus, the determination of $\mathcal{N}_m(\epsilon, \mathcal{F})$ can be done in terms of the properties of the operator T_k, as the latter plays a constructive role in controlling the complexity of $\mathcal{F}$, and hence the difficulty of the learning task (Smola, Williamson, Mika, and Scholkopf (1999), Williamson, Shawe-Taylor, Scholkopf, and Smola (1999), Smola, Elisseff, Scholkopf, and Williamson (2000), Williamson, Smola, and Scholkopf (2000, 2001). Often the determination is done in terms of packing numbers in place of covering numbers – however, they are equivalent up to a factor of 2 (Anthony and Bartlett (1999)).

Literature, Approaches, and Result

1. Concept of Metric Entropy: The concept of metric entropy of a set has been around for some time. It seems to have been introduced by Pontriagin and Schnirelmann (1932) and was studied in detail by Kolmogorov and others (Kolmogorov and Tihomirov (1961), Lorentz, van Golitschek, and Makovoz (1996)).
2. Entropy Numbers and Linear Operator Spectrum: The use of metric entropy to say something about linear operators was carried out independently by several people. Prosser (1966) seems to have been the first to make the idea explicit.
3. Entropy Rates and Eigenvalue Asymptotics: In particular, Prosser (1966) proved a number of results concerning the asymptotic rate of decrease of the entropy numbers in terms of the

asymptotic behavior of the eigenvalues. A similar result is implicit in Shannon's famous paper (Shannon (1948)) where he considered the effect of different convolutional operators on the entropy of an ensemble. Prosser's work led to several studies on convolutional operators (Prosser and Root (1968), Jagerman (1969), Akashi (1990), Koski, Persson, and Peetre (1994)).

4. ϵ-entropies of Linear Operators and Stochastic Processes: The connection between Prosser's ϵ-entropy of an operator and Kolmogorov's ϵ-entropy of stochastic processes was established in Akashi (1986). Additional work studied covering numbers (Triebel (1970), Carl and Stephani (1990)) and entropy numbers in the context of operator ideals (Pietsch (1980)), Carl (1981)).
5. Banach Spaces and Uniform Convergence: Connections between the local theory of Banach spaces and uniform convergence of empirical means has been noted before (e.g., Pajor (1985)). Gurvits (1997) related the Rademacher-type of a Banach space to the fat-shattering dimensions of linear functionals on that space, and hence via the key result in Alon, Ben-David, Cesa-Bianchi, and Haussler (1997) to the covering number of the induced class (Williamson, Scholkopf, and Smola (1999)).
6. Operator Type and Entropy Number Decay: The equivalence of the type of an operator (or of the space that it maps to), and the rate of decay of its entropy numbers has been established independently by Koltchinskii (1988, 1991), Defant and Junge (1993), and Junge and Defant (1993) (albeit under different formulations – Koltchinskii's work is motivated by probabilistic considerations).

References

- Akashi, S. (1986): Characterization of ϵ-entropy in Gaussian Processes *Kodai Mathematical Journal* **9** 58-67.
- Akashi, S. (1990): The Asymptotic Behavior of ϵ-entropy of a Compact Positive Operator *Journal of Mathematical Analysis and Applications* **153** 250-257.

- Alon, N., S. Ben-David, N. Cesa-Bianchi, and D. Haussler (1997): Scale-sensitive Dimensions, Uniform Convergence, and Learnability *Journal of the Association of the Computing Machinery* **44 (4)** 615-631.
- Anthony, M. (1997): Probabilistic Analysis of Learning in Artificial Neural Networks: The PAC Model and its Variants *Neural Computational Survey* **1** 1-47.
- Anthony, M., and P. L. Bartlett (1999): *Artificial Neural Network Learning: Theoretical Foundations* **Cambridge University Press** Cambridge UK.
- Carl, B. (1981): Entropy Numbers of Diagonal Operators with an Application to the Eigenvalue Problems *Journal of Approximation Theory* **32** 135-150.
- Carl, B., and I. Stephani (1990): *Entropy, Compactness, and the Approximation of Operators* **Cambridge University Press** Cambridge UK.
- Defant, M., and M. Junge (1993): Characterization of Weak Type by the Entropy Distribution of r-nuclear Operators *Studia Math.* **107 (1)** 1-14.
- Gurvits, L. (1997): A Note on Scale-sensitive Dimension of Linear Bounded Functionals in Banach Spaces, in: *Algorithmic Learning Theory ALT-97 (Lecture Notes in Artificial Intelligence)* (M. Li and A. Marouka, editors) **1316** 352-363 **Springer-Verlag** Berlin, Germany.
- Jagerman, D. (1969): ϵ-entropy and Approximation of Band-limited Functions *SIAM Journal of Applied Mathematics* **17 (2)** 362-377.
- Junge, M., and M. Defant (1993): Some Estimates of Entropy Numbers *Israeli Journal of Mathematics* **84** 417-433.
- Kolmogorov, A. N., and V. M. Tihomirov (1961): ϵ-entropy and ϵ -capacity of Sets in Function Spaces *Amer. Math. Soc. Transl. (2)* **17** 277-364.
- Koltchinskii (1988): Operators of Type p and Metric Entropy *Teoriya Veroyatnosteyi Matematicheskaya Statistika* **38** 69-76, 135.
- Koltchinskii (1991): Entropic Order of Operators in Banach Spaces and the Central Limit Theorem *Theory of Probability and Its Applications* **36 (2)** 303-315.
- Koski, T., L. E. Persson, and J. Peetre (1994): ϵ-entropy, ϵ-rate, and Interpolation Spaces revisited with an Application to Linear Communication Channels *Journal of Mathematical Analysis and Applications* **186** 265-276.

- Lorentz, G. G., M. van Golitschek, and Y. Makovoz (1996): *Constructive Approximation: Advanced Problems* **Springer-Verlag** Berlin Germany.
- Pajor, A. (1985): *Sous-Espaces l_n^1 des Espaces de Banach* **Hermann** Paris, France.
- Pontriagin, L. S., and L. G. Schnirelmann (1932): Sur une Propriete Metrique de la Dimension *Annals of Mathematics* **33** 156-162.
- Pietsch, A. (1980): *Operator Ideals* **North-Holland** Amsterdam.
- Prosser, R. T. (1966): The ϵ-entropy and ϵ-capacity of certain Time Varying Channels *Journal of Mathematical Analysis and Applications* **16** 553-573.
- Prosser, R. T., and W. L. Root (1968): The ϵ-entropy and ϵ-capacity of certain Time Invariant Channels *Journal of Mathematical Analysis and Applications* **21** 233-241.
- Shannon, C. E. (1948): A Mathematical Theory of Communication *Bell System Technical Journal* **27** 379-423, 623-656.
- Smola, A. J., R. C. Williamson, S. Mika, and B. Scholkopf (1999): Regularized Principal Manifolds, in: *Proceedings of the 4th European Workshop on Computational Learning Theory (EUROCOLT '99)* 214-229.
- Smola, A. J., A. Elisseff, B. Scholkopf, and R. C. Williamson (2000): Entropy Numbers for Convex Combinations and mlps, in: *Advances in large Margin Classifiers* (A. Smola, P. Bartlett, B. Scholkopf, and D. Schuurmans, editors) **MIT Press** Cambridge MA.
- Triebel, H. (1970): Interpolationseigenschaften von Entropie- und Durchmesseridealen Kompakter Operatoren *Studia Math.* **34** 89-107.
- Williamson, R. C., B. Scholkopf, and A. J. Smola (1999): A Maximum Margin Miscellany.
- Williamson, R. C., J. Shawe-Taylor, B. Scholkopf, and A. J. Smola (1999): Sample-based Generalization Bounds *NeuroCOLT2 Technical Report Series* **NC-TR-1999-055**.
- Williamson, R. C., A. J. Smola, and B. Scholkopf (2000): Entropy Numbers of Linear Function Classes, in: *Proceedings of the 13th Annual Conference on Computational Learning Theory (N. Cesa-Bianchi and S. Goldman, editors)* **ACM** New York.
- Williamson, R. C., A. J. Smola, and B. Scholkopf (2001): Generalization Performance of Regularization Networks and Support Vector Machines via Entropy Numbers of Compact Operators *IEEE Transactions on Information Theory* **47 (6)** 2516-2532.

Generalization Bounds via Uniform Convergence

Basic Uniform Convergence Bounds

1. Generalization Performance of Learning Machines: The generalization performance of learning machines can be bounded via uniform convergence results presented in Vapnik and Chervonenkis (1981) and Vapnik (1982) (Anthony (1997) and Kulkarni, Lugosi, and Venkatesh (1998) contain reviews). The capacity of the hypothesis class is often expressed in terms of the covering numbers.
2. Classification and Regression Covering Numbers: Results for both classification and regression are known. For the sake of concreteness, below is a quote suitable for regression, proved in Alon, Ben-David, Cesa-Bianchi, and Haussler (1997). Bartlett and Shawe-Taylor (1999) contain results of classifier performance in terms of covering numbers. We first set

$$\mathcal{P}_m(f) \doteqdot \frac{1}{m}\sum_{i=1}^{m} f(\vec{x}_i)$$

   to denote the *empirical means* of f on the sample $\vec{x}_1, \ldots, \vec{x}_m$.
3. Regression Uniform Convergence Bound: Let $\mathcal{F}$ be a class of functions from $\mathcal{X}$ to $[0, 1]$, and let $\mathcal{P}$ be a distribution over $\mathcal{X}$. Then for all

$$\epsilon > 0$$

   and

$$m \geq \frac{2}{\epsilon^2}$$

$$\mathbb{P}\left\{\sup_{f \in \mathcal{F}} |\mathcal{P}_m(f) - \mathcal{P}(f)| > \epsilon\right\} \leq 12m.\mathbb{E}\left[\mathcal{N}\left(\frac{\epsilon}{6}, \mathcal{F}, l_\infty^{\vec{X}_{2m}}\right)\right] e^{-\frac{m\epsilon^2}{36}}$$

where $\mathbb{P}$ denotes the probability with respect to the sample $\vec{x}_1, \ldots, \vec{x}_m$ drawn i.i.d. from $\mathcal{P}$, and $\mathbb{E}$ the expectation with respect to the first, and a second sample also drawn i.i.d. from $\mathcal{P}$, i.e.

$$\vec{X}_{2m} = \{\vec{x}_1, \ldots, \vec{x}_{2m}\}$$

(Alon, Ben-David, Cesa-Bianchi, and Haussler (1997)).

4. Use of the Regression Bound Lemma: To be able to use the above lemma, one usually makes use of the fact that for any $\mathcal{P}$

$$\mathbb{E}\left[\mathcal{N}\left(\epsilon, \mathcal{F}, l_\infty^{\vec{X}_{2m}}\right)\right] < \sup_{\vec{x}_1, \ldots, \vec{x}_m \in \mathcal{X}} \mathcal{N}\left(\epsilon, \mathcal{F}, l_\infty^{\vec{X}_{2m}}\right) < \mathcal{N}_m(\epsilon, \mathcal{F})$$

Loss Function Induced Classes

1. Application to Loss Function: The above result can be used to give a generalization error result by applying it to the loss-function-induced class. The following Lipschitz-condition lemmas on loss function, which are an improved version of the lemmas in Bartlett, Long, and Williamson (1996), is useful in this regard (a similar result appears in Anthony and Bartlett (1999)).
2. Loss Function Lemmas - Background: Let $\mathcal{F}$ be a class of functions from $\mathcal{X}$ to $[a, b]$, with

$$a < b$$

$$a, b \in \mathbb{R}$$

and

$$l: \mathbb{R} \to [0, \infty)$$

a loss function. Let the following conditions/definitions hold:

$$\vec{X}_m = (\vec{x}_1, \dots, \vec{x}_m)$$

$$\vec{z}_i \doteqdot (\vec{x}_i, y_i)$$

$$\vec{Z}_m = (\vec{z}_1, \dots, \vec{z}_m)$$

$$l_{f|\vec{z}_j} \doteqdot l\left(f\left(\vec{x}_j\right) - y_j\right)$$

$$l_{f|\vec{Z}_m} \doteqdot \left(l_{\mathcal{F}|\vec{z}_j}\right)_{j=1}^{m}$$

$$l_{\mathcal{F}|\vec{Z}_m} \doteqdot \left\{l_{f|\vec{z}_j} : f \in \mathcal{F}\right\}$$

$$\mathcal{N}\left(\epsilon, l_{|\vec{Z}_m}\right) \doteqdot \mathcal{N}\left(\epsilon, l_{\mathcal{F}|\vec{Z}_m}, l_{\infty}^{\vec{Z}_m}\right)$$

3. Lipschitz Loss Condition Lemma: Suppose l satisfies the Lipschitz condition

$$l(\xi) - l(\xi') \leq C|\xi - \xi'|$$

for all

$$\xi, \xi' \in [a - b, b - a]$$

Then for all

$$\epsilon > 0$$

$$\max_{\vec{Z}_m \in \{\mathcal{X} \times [a,b]\}^m} \mathcal{N}\left(\epsilon, l_{|\vec{Z}_m}\right) \leq \max_{\vec{X}_m \in \mathcal{X}_m} \mathcal{N}\left(\frac{\epsilon}{C}, \mathcal{F}_{|\vec{X}_m}, l_\infty^m\right)$$

and

$$\max_{\vec{Z}_m \in \{\mathcal{X} \times [a,b]\}^m} \mathcal{N}\left(\epsilon, l_{|\vec{Z}_m}\right) \leq \max_{\vec{X}_m \in \mathcal{X}_m} \mathcal{N}\left(\frac{\epsilon m}{C}, \mathcal{F}_{|\vec{X}_m}, l_\infty^1\right)$$

4. Approximate Lipschitz Loss Condition Lemma: Suppose that for some

$$C, \tilde{C} > 0$$

l satisfies the "approximate Lipschitz condition"

$$l(\xi) - l(\xi') \leq max\left(C|\xi - \xi'|, \tilde{C}\right)$$

for all

$$\xi, \xi' \in [a - b, b - a]$$

then for all

$$\epsilon \geq \frac{\tilde{C}}{C}$$

$$\max_{\vec{Z}_m \in \{\mathcal{X} \times [a,b]\}^m} \mathcal{N}\left(\epsilon, l_{|\vec{Z}_m}\right) \leq \max_{\vec{X}_m \in \mathcal{X}_m} \mathcal{N}\left(\frac{\epsilon}{C}, \mathcal{F}_{|\vec{X}_m}, l_\infty^m\right)$$

5. Lipschitz Loss Lemma Proof: We show that, for any sequence $\vec{Z}_m$ of $(\vec{x}, y)$ pairs in

$$\mathcal{X} \times [a, b]$$

and any functions f and g, if the restrictions of f and g to $\vec{X}_m$ are close, then the restrictions of l_f and l_g to $\vec{Z}_m$ are closer. Thus, given a cover of $\mathcal{F}_{|\vec{X}_m}$ we can construct a cover of $l_{\mathcal{F}|\vec{Z}_m}$ that is no bigger.

6. Lipschitz Loss Condition Bound:

$$\frac{1}{m}\left|\sum_{j=1}^{m}\{l[g(\vec{x}_j) - y_j] - l[f(\vec{x}_j) - y_j]\}\right| \leq \frac{1}{m}\left\{\sum_{j=1}^{m}\left|l[g(\vec{x}_j) - y_j] - l[f(\vec{x}_j) - y_j]\right|\right\}$$
$$\leq \frac{1}{m}\sum_{j=1}^{m} C|g(\vec{x}_j) - f(\vec{x}_j)| = \frac{C}{m}\left\|g(\vec{X}_m) - f(\vec{X}_m)\right\|_{l_\infty^1}$$
$$\leq C\left\|g(\vec{X}_m) - f(\vec{X}_m)\right\|_{l_\infty^m}$$

7. Approximate Lipschitz Loss Condition Bound: This condition is useful when the exact form of the loss function is unknown, happens to be discontinuous, or is badly behaved in some other way

$$\frac{1}{m}\left|\sum_{j=1}^{m}\{l[g(\vec{x}_j) - y_j] - l[f(\vec{x}_j) - y_j]\}\right| \leq \frac{C}{m}\sum_{j=1}^{m} max\left[g(\vec{x}_j) - f(\vec{x}_j), \frac{\tilde{C}}{C}\right] \leq C\epsilon$$

for

$$\epsilon \geq \frac{\tilde{C}}{C}$$

8. Polynomial Loss Function Bound: For either of the Lipschitz loss conditions above, if we assume a loss function of the type

$$l(\eta) = \frac{1}{p}\eta^p$$

with

$$p > 1$$

we have

$$\mathcal{C} = (b-a)^{p-1}$$

in particular

$$\mathcal{C} = b - a$$

for

$$p = 2$$

and therefore

$$\max_{\vec{Z}_m \in \{\mathcal{X} \times [a,b]\}^m} \mathcal{N}\left(\epsilon, l_{|\vec{Z}}\right) \leq \max_{\vec{x} \in \mathcal{X}_m} \mathcal{N}\left(\frac{\epsilon}{(b-a)^{p-1}}, \mathcal{F}_{|\vec{x}}\right)$$

Standard Form of Uniform Convergence

1. The Standard Form: One can readily combine the uniform convergence results with the above results to get overall bounds on generalization performance. A typical uniform convergence result takes the form

$$\mathcal{P}_m\left\{\sup_{f \in \mathcal{F}} \left|\mathcal{R}_{Emp}(f) - \mathcal{R}(f)\right| > \epsilon\right\} \leq c_1(m)\mathcal{N}_m(\epsilon, \mathcal{F})e^{-\frac{m\epsilon^\beta}{c_2}}$$

where $\mathcal{R}_{Emp}(f)$ is the empirical risk, and $\mathcal{R}(f)$ is the expected risk of

$$f \in \mathcal{F}$$

(Vapnik (1998), Anthony and Bartlett (1999)).

2. Exponents in the Standard Form: The exponent β depends on the setting. For regression, β can be set to 1. However, in agnostic learning in general

$$\beta = 2$$

(Kearns, Schapire, and Sellie (1994)), except if the class is convex, in which case β can be set to 1 (Lee, Bartlett, and Williamson (1998)).

3. Sample Complexity: The generalization bounds produced above are typically used by setting the right-hand side to δ, and solving for

$$m \equiv m(\epsilon, \delta)$$

m is called the sample complexity.

4. Learning Curve Bound: Another way to use the above result is as a learning curve bound $\bar{\epsilon}(\delta, m)$, where

$$\mathcal{P}_m\left\{\sup_{f \in \mathcal{F}} \left|\mathcal{R}_{Emp}(f) - \mathcal{R}(f)\right| > \bar{\epsilon}(\delta, m)\right\} \leq \delta$$

It is to be noted that the determination of $\bar{\epsilon}(\delta, m)$ is quite convenient in terms of e_n, the dyadic entropy number associated with the covering number $\mathcal{N}(\epsilon, \mathcal{F})$.

5. The Dyadic Entropy Number: Setting the right-hand side in the standard uniform convergence form to δ, we have

$$\delta = c_1(m)\mathcal{N}_m(\epsilon, \mathcal{F})e^{-\frac{m\epsilon^\beta}{c_2}} \rightarrow \log\left[\frac{\delta}{c_1(m)}\right] + \frac{m\epsilon^\beta}{c_2} = \log \mathcal{N}_m(\epsilon, \mathcal{F}) \rightarrow \epsilon$$

$$\geq e^{\left\lfloor \log\left[\frac{\delta}{c_1(m)}\right] + \frac{m\epsilon^\beta}{c_2} + 1 \right\rfloor}$$

Thus

$$\bar{\epsilon}(\delta, m) = \min\left\{\epsilon : \epsilon \geq e^{\left\lfloor \log\left[\frac{\delta}{c_1(m)}\right] + \frac{m\epsilon^\beta}{c_2} + 1 \right\rfloor}\right\}$$

holds. Therefore, the use of e_n or ϵ_n is a convenient thing to do for locating learning curves.

References

- Alon, N., S. Ben-David, N. Cesa-Bianchi, and D. Haussler (1997): Scale-sensitive Dimensions, Uniform Convergence, and Learnability *Journal of the Association of the Computing Machinery* **44 (4)** 615-631.
- Anthony, M. (1997): Probabilistic Analysis of Learning in Artificial Neural Networks: The PAC Model and its Variants *Neural Computational Survey* **1** 1-47.
- Anthony, M., and P. L. Bartlett (1999): *Artificial Neural Network Learning: Theoretical Foundations* **Cambridge University Press** Cambridge UK.
- Bartlett, P., and J. Shawe-Taylor (1999): Generalization Performance of Support Vector Machines and other Pattern Classifiers, in: *Advances in Kernel Methods – Support vector Learning* (B. Scholkopf, C. J. C. Burges, and A. J. Smola (editors)) 43-54 **MIT Press** Cambridge MA.
- Bartlett, P., P. Long, and R. C. Williamson (1999): Fat-shattering and the Learnability of Real-valued Functions *Journal of Computers and System Science* **52 (3)** 434-452.
- Kearns, M. J., R. E. Schapire, and L. M. Sellie (1994): Towards Efficient Agnostic Learning *Machine Learning* **17 (2)** 115-141.

- Kulkarni, S. R., G. Lugosi, and S. S. Venkatesh (1998): Learning Pattern Classification – A Survey *IEEE Transactions on Information Theory* **44** 2178-2206.
- Lee, W. S., P. L. Bartlett, and R. C. Williamson (1998): The Importance of Convexity in Learning with Squared Loss *IEEE Transactions on Information Theory* **44** 1974-1980.
- Vapnik, V. N., and A. Y. Chervonenkis (1981): Necessary and Sufficient Conditions for the Uniform Convergence of the Means to their Expectations *Theory of Probability and its Applications* **26 (3)** 532-553.
- Vapnik, V. N. (1982): *Estimation of Dependencies from Empirical Data* **Springer-Verlag** New York.
- Vapnik, V. N. (1998): *Statistical learning Theory* **Wiley** New York.

Kernel Machines

Introduction

1. Definition: Kernel machines perform a mapping from an input space onto a feature space (Aizerman, Braverman, and Rozonoer (1964)), construct regression functions and/or decision boundaries based on this mapping, and use constraints in the feature space for capacity control (Nilsson (1965)).
2. Support Vector Machines: Support Vector Machines, which have evolved as a full new class of learning algorithms, solving problems in pattern recognition, regression estimation, and operator inversion (Vapnik (1995)), are a well-known example of kernel machines.
3. Feature Maps: SV machines, like most kernel-based methods, possess the nice property of defining the feature map in a manner that allows its computation implicitly at little additional cost.
4. Generalization Performance of Kernel Machines: Williamson, Smola, and Scholkopf (2001) show how bounds on covering numbers can be obtained by employing relatively standard methods, and hence estimate a bound on their generalization performance. Similar approach may be applied to bound regularization networks (Girosi, Jones, and Poggio (1993)), or certain unsupervised algorithms (Scholkopf, B., A. Smola, and K. R. Muller (1998)).

SVM – Capacity Control

1. Maximum Margin of Separation: In order to perform pattern recognition using linear hyper-planes, often a maximum margin of separation between the classes is sought, as this leads to good generalization ability independent of dimensionality (Vapnik and Chervonenkis (1974), Vapnik (1995), Shawe-Taylor, Bartlett, Williamson, and Anthony (1996)).
2. Maximum Margin Classification: It can be shown that for a separable training data

$$(\vec{x}_1, y_1), \cdots, (\vec{x}_m, y_m) \in \mathbb{R}^d \times \{\pm 1\}$$

the maximum margins are achieved by minimizing $\|\vec{w}\|_2$ subject to the constraints

$$y_j\left(\langle \vec{w}, \vec{x}_j \rangle + b\right) \geq 1$$

for

$$j = 1, \ldots, m$$

The decision functions then take on the form

$$f(\vec{x}) = sgn\left(\langle \vec{w}, \vec{x} \rangle + b\right)$$

3. Maximum Margin Linear Regression: Likewise, a linear regression

$$f(\vec{x}) = \langle \vec{w}. \vec{x} \rangle + b$$

can be estimated from the data

$$(\vec{x}_1, y_1), \cdots, (\vec{x}_m, y_m) \in \mathbb{R}^d \times \mathbb{R}$$

by locating the flattest function that approximates the data with some margin of error. In this case one minimizes $\|\vec{w}\|_2$ subject to

$$\left|f\left(\vec{x}_j\right) - y_j\right| \leq \epsilon$$

where the parameter

$$\epsilon > 0$$

plays the role of the margin, although not in the space of inputs $\vec{x}$, but in the space of the outputs y.

4. Maximum Margins - Generalization: In both classification and regression, generalizations for the non-separable and the non-realizable cases exist, using various types of cost functions (Cortes and Vapnik (1995), Vapnik (1995), Smola and Scholkopf (1998)).

Non-linear Kernels

1. Dot-Products in High-Dimensional Spaces: To be able to apply the above technique to a general class of *non-linear* functions, one can use kernels computing dot products in high dimensional spaces non-linearly related to input (Aizerman, Braverman, and Rozonoer (1964)).
2. The Kernel Trick: Under certain conditions on the kernel k (these are listed later), there exists a non-linear map Φ into a reproducing kernel Hilbert space F (e.g., Saitoh (1988)) such that k computes the dot product in F, i.e.

$$k(\vec{x}, \vec{y}) = \langle \Phi(\vec{x}), \Phi(\vec{y}) \rangle_F$$

3. The Kernel Algorithm and its Applications: Thus, given any algorithm that can be expressed in terms of dot products exclusively, one can construct a non-linear version of it by substituting a kernel for the dot-product. Examples of such machines include SV pattern recognition (Boser, Guyon, and Vapnik (1992)), SV regression estimation (Vapnik (1995)), and kernel principal component analysis (Shawe-Taylor, Bartlett, Williamson, and Anthony (1996)).
4. Extension of the Maximum Margin Philosophy: By using the kernel trick for SV machines, the maximum margin idea is thus extended to a large variety of functions classes (e.g., radial basis function networks, polynomial networks, neural networks), which, in the case of regression estimation, comprise functions written as kernel expansions

$$f(\vec{x}) = \sum_{j=1}^{m} \alpha_j k_j(\overrightarrow{x_j}, \vec{x}) + b$$

where

$$\alpha_j \in \mathbb{R}, j = 1, \ldots, m$$

5. Regularization Properties: It has been noticed that the different kernels can be characterized by their regularization properties (Smola, Scholkopf, and Muller (1998)). SV machines are regularization networks minimizing the regularized risk

$$R_{Reg}[f] = R_{Emp}[f] + \frac{\lambda}{2}\|\mathcal{P}f\|^2$$

with a regularization parameter

$$\lambda \geq 0$$

and a regularization operator $\mathcal{P}$ over the set of functions of the form $f(\vec{x})$ shown above, provided that k and $\mathcal{P}$ are inter-related through

$$k(\overrightarrow{x_s}, \overrightarrow{x_t}) = \langle (\mathcal{P}k)(\overrightarrow{x_s}, \cdot), (\mathcal{P}k)(\overrightarrow{x_t}, \cdot) \rangle$$

To this end, k is chosen as the Green's function of $\mathcal{P}^*\mathcal{P}$ where $\mathcal{P}^*$ is the adjoint of $\mathcal{P}$.

6. Kernel Selection: While the analysis above provides insight into the regularization properties of the SV kernels, it does not settle the issue of how to select a kernel for a given learning problem, and how using a specific kernel might influence the performance of an SV machine.

Generalization Performance of Regularization Networks

1. Introduction: Williamson, Smola, and Scholkopf (2001) show that the properties of the spectrum of the kernel can be used to estimate the generalization error of the associated class of the learning machines.
2. Tuning the Generalization Error: The kernel is not just a means of broadening the class of functions alone, e.g., by rendering the non-separable data separable in a feature space non-linearly related to the input space. It is a constructive handle by which to control the generalization error.
3. Direct Bounding of the Covering Number: Williamson, Smola, and Scholkopf (2001) show how to directly bound the covering number of interest rather than make use of a combinatorial dimension (such as the Vapnik Chervonenkis (VC) dimension or the fat-shattering dimension) and subsequently applying a general result relating such dimensions to covering numbers.
4. Covering Numbers Construction: The covering numbers can be bound directly by viewing the values induced by the relevant class of functions as the image of a unit ball under a particular compact operator.

Covering Number Determination Steps

1. Step #1 - Entropy Numbers and Kernel Spectrum: First, formulate the generalization error in terms of the Entropy Numbers. In particular, it is important to relate the bounding entropy numbers in terms of the spectrum of a given kernel.
2. Step #2 - Function Class Covering Numbers: Identify the relation between the covering numbers of function classes and the entropy number of suitably defined operators. In particular, establish an upper bound on the entropy numbers in terms of the size of the weight vector in the feature space and eigenvalues of the kernel used. Apply standard techniques in case of kernels such as

$$k(x, y) = e^{-(x-y)^2}$$

which do not have a discrete spectrum.

3. Step #3 - Eigenvalue Decay Rate: Well-established results on the entropy numbers obtained for given rates of decay of eigenvalues (and their extension to multiple dimensions) can be employed.
4. Step #4 - Bringing it all together: Williamson, Smola, and Scholkopf (2001) show how the steps above along with the results may be glued in together to obtain overall bounds on the generalization error. While they deal with the eigenvalues of translation-invariant (i.e., convolutional) kernels, the general theory is not restricted to them.

Challenges Presenting Master Generalization Error

1. Problem Specificity: The particular statistical result one needs to use may be very specific to the problem at hand.
2. SRM Weakness: While SRM helps establish the existence of good generalization bounds those are necessary and sufficient (Vapnik (1982)), many of the results obtained are in a form which, while quite amenable to ready computation on a computer, do not provide much observational insight, except in an asymptotic sense. More explicit formulas suitable for SRM have been developed in Guo, Bartlett, Shawe-Taylor, and Williamson (1999).
3. Additional Extraneous Inputs: Applications such as classification may need margins to be estimated in a data-dependent fashion, thereby requiring additional "luckiness" arguments (Shawe-Taylor, Bartlett, Williamson, and Anthony (1998)) to apply bounds.

References

- Aizerman, M. A., F. M. Braverman, and L. I. Rozonoer (1964): Theoretical Foundations of the Potential Function Method in Pattern Recognition Learning *Automation Remote Control* **25** 821-837.
- Boser, B. E., I. M. Guyon, and V. N. Vapnik (1992): A Training Algorithm for Optimal Margin Classifiers, in: *5th Annual ACM Workshop on COLT (D. Haussler, editor)* **ACM Press** Pittsburgh PA.

- Cortes, C., and V. N. Vapnik (1995): Support Vector Networks *Machine Learning* **20** 273-297.
- Girosi, F., M. Jones, and T. Poggio (1993): Prior, Stabilizers, and Basis Functions: From Regularization to Radial, Tensor, and Additive Splines *A. I. Memo 1430* **Massachusetts Institute of Technology** Cambridge MA.
- Guo, Y., P. L. Bartlett, J. Shawe-Taylor, and R. C. Williamson (1999): Covering Numbers for Support Vector Machines, in: *Proceedings of the 12th Annual Conference on Computational Learning Theory* 267-277 **ACM** New York.
- Nilsson, N. J. (1965): *Learning Machines: Foundations of Trainable Pattern Classifying Systems* **McGraw-Hill** New York.
- Saitoh, S. (1988): *Theory of Reproducing Kernels and its Applications* **Longman** Harlow UK.
- Scholkopf, B., A. Smola, and K. R. Muller (1998): Non-linear Component Analysis as a Kernel Eigenvalue Problem *Neural Computation* **10** 1299-1319.
- Shawe-Taylor, J., P. L. Bartlett, R. C. Williamson, and M. Anthony (1996): A Framework for Structural Risk Minimization, in: *Proceedings of the 9th Annual Conference on the Computational Learning Theory* 68-76 **ACM** New York.
- Shawe-Taylor, J., P. L. Bartlett, R. C. Williamson, and M. Anthony (1998): Structural Risk Minimization over Data-dependent Hierarchies *IEEE Transactions on Information Theory* **44** 1926-1940.
- Smola, A. J., and B. Scholkopf (1998): On a Kernel-based Method for Pattern Recognition, Regression, Approximation, and Operator Inversion *Algorithmica* **22** 211-231.
- Vapnik, V. N., and A. Chervonenkis (1974): *Theory of Pattern Recognition (in Russian)* **Nauka** Moscow.
- Vapnik, V. N. (1982): *Estimation of Dependencies from Empirical Data* **Springer-Verlag** New York.
- Vapnik, V. N. (1995): *The Nature of Statistical Learning Theory* **Springer-Verlag** New York.
- Williamson, R. C., A. J. Smola, and B. Scholkopf (2001): Generalization Performance of Regularization Networks and Support Vector Machines via Entropy Numbers of Compact Operators *IEEE Transactions on Information Theory* **47 (6)** 2516-2532.

Entropy Numbers for Kernel Machines

Mercer Kernels

1. Introduction: Here we will mainly consider machines whose mapping into the feature space is defined by Mercer kernels $k(\vec{x}, \vec{y})$ as they are easier to deal with using functional analytics methods. More general kernels are considered in Smola, Elisseff, Scholkopf, and Williamson (2000). Such machines have become very popular due to the success of SV machines.
2. Goals: Our goal is to make statements of the shape of the image of the input space $\mathcal{X}$ under the feature map $\Phi(.)$. The version of the Mercer's theorem we use here is a special case of the theorem proven in Konig (1986). In what follows, we assume $(\mathcal{X}, \mu)$ is a finite measure space, i.e.

$$\mu(\mathcal{X}) < \infty$$

3. Mercer Theorem - Notation: Suppose

$$k \in L_\infty(\mathcal{X}, \mathcal{X})$$

is a symmetric kernel, i.e.

$$k(x, x') = k(x', x)$$

such that the integral operator

$$T_k : L_2(\mathcal{X}) \to L_2(\mathcal{X}) \Longrightarrow T_k[f(\cdot)] \doteq \int_{\mathcal{X}}^{\mathcal{X}} k(\cdot, \vec{y})\, f(\vec{y}) d\mu(\vec{y})$$

is positive. Let

$$\psi_j \in L_\infty(\mathcal{X})$$

be the eigen-function of T_k associated with the eigenvalue

$$\lambda_j \neq 0$$

and normalized by

$$\|\psi_j\|_{L_2} = 1$$

Finally, suppose ψ_j is continuous for all

$$j \in \mathbb{N}$$

4. Mercer's Condition: Using the notation above, we can specify Mercer's conditions:
   a.

$$\left(\lambda_j(T)\right)_j \in l_1$$

   for

$$j = 1, 2, \ldots$$

   b.

$$\psi_j \in L_\infty(\mathcal{X})$$

   and

$$\sup_j \|\psi_j\|_{L_\infty} < \infty$$

c.

$$k(\vec{x}, \vec{y}) = \sum_{j \in \mathbb{N}} \lambda_j \psi_j(\vec{x}) \psi_j(\vec{y})$$

holds for all

$$(\vec{x}, \vec{y}) \in \mathcal{X} \times \mathcal{X}$$

d. Finally, the series $\lambda_j \psi_j(\vec{x}) \psi_j(\vec{y})$ converges absolutely and uniformly for all

$$(\vec{x}, \vec{y}) \in \mathcal{X} \times \mathcal{X}$$

5. Assumption of Continuity of ψ_j: Note that if $\mathcal{X}$ is compact and k is continuous, then ψ_j is continuous (Ash (1965)). Alternatively, if k is translation-invariant, then ψ_j's are scaled cosine functions, and thus continuous. Therefore, the assumption that ψ_j are continuous is not very restrictive.
6. Impact of Mercer's Second Condition: From the second condition it follows that there exists some constant

$$C_k \in \mathbb{R}^+$$

depending on $k(\cdot,\cdot)$ such that

$$\left|\psi_j(\vec{x})\right| \leq C_k$$

for all

$$j \in \mathbb{N}$$

and

$$\vec{x} \in \mathcal{X}$$

7. Impact of Mercer's Third Condition: From the third condition it follows that $k(\vec{x}, \vec{y})$ corresponds to a dot-product in l_2, i.e.

$$k(\vec{x}, \vec{y}) = \langle \Phi(\vec{x}), \Phi(\vec{y}) \rangle_{l_2}$$

with

$$\Phi: \mathcal{X} \to l_2 \Longrightarrow \Phi: \vec{x} \mapsto \left(\phi_j(\vec{x})\right)_j \doteq \left(\sqrt{\lambda_j}\psi_j(\vec{x})\right)_j$$

8. Spatial Span of $\Phi(\mathcal{X})$: From the above argument one can see that $\Phi(\mathcal{X})$ lives not only in l_2, but in an axis parallel parallelepiped with lengths $2C_k\sqrt{\lambda_j}$. Also, we assume, without loss of generality, that the sequence $\left(\lambda_j\right)_j$ is sorted in a non-increasing order.

Equivalent Kernels

1. The Equivalent Kernel Lemma: Denote by $\mathcal{X}$ a compact set, and by

$$k: \mathcal{X}^2 \to \mathbb{R}$$

a Mercer kernel. Then for any $\mathcal{X}'$ and a surjective map

$$\chi: \mathcal{X} \to \mathcal{X}'$$

the kernel

$$k(x, x') = k[\chi(x), \chi(x')]$$

also satisfies Mercer's condition, and, moreover, the eigenvalues $\lambda_i{}'$ and the coefficient $C_{k'}$ of the integral operator

$$T_{k'}[f(x)] \doteqdot \int_{\mathcal{X}'}^{\mathcal{X}'} k'(x, x')\, f(x') dx'$$

can be used equivalently in any application of k.

2. Proof of the Lemma: The first part of the claim, namely that k' also satisfies Mercer's condition, follows immediately from the construction of k'. For the second claim, note that due to the fact that χ is surjective for any distribution $p(x)$ on $\mathcal{X}$ there must exist an equivalent distribution $p'(x)$ on $\mathcal{X}'$. Therefore, we can always consider the problem as being one on $\mathcal{X}'$ from the start.
3. Impact of the Lemma: Note that since $\mathcal{X}$ and $\mathcal{X}'$ were chosen arbitrarily, we can optimize over them. In particular, this means that we could construct diffeomorphisms

$$\chi : \mathcal{X} \to \mathcal{X}'$$

and look for the function χ such that the eigenvalues $\lambda_i{}'$ and $C_{k'}$ are as small as possible.

4. Dependence on μ: Note that the measure μ need have nothing to do with the distribution of our sample. However, the Equivalent Kernel lemma shows that the specific bounds we obtain *will* depend on μ since that will affect ψ_j, C_k, and $(\lambda_j)_j$. The question of the optimal μ to use and how it may be chosen if one knows P (the distribution from which $\vec{x}$ are chosen) requires a full treatment on in its own right, but it is common for μ to be a Lebesgue measure.

Mapping Φ Into l_2

1. Motivation: It will be useful to consider maps that map $\Phi(\mathcal{X})$ into balls of some radius R centered at the origin. The following proposition shows that the class of all these maps is determined by the elements of l_2 and the sequence of eigenvalues $(\lambda_j)_j$.
2. l_2 Bounding of $\Phi(\mathcal{X})$ - Statement: Let $\mathcal{S}$ be a diagonal map

$$\mathcal{S}: \mathbb{R}^n \to \mathbb{R}^n \Longrightarrow \mathcal{S}: (\lambda_j)_j \mapsto \mathcal{S}(\lambda_j)_j = (s_j \lambda_j)_j$$

with

$$s_j \in \mathbb{R}$$

Then $\mathcal{S}$ maps $\Phi(\mathcal{X})$ into a ball of finite radius $R_{\mathcal{S}}$ centered at the origin if and only if

$$\left(s_j \sqrt{\lambda_j}\right)_j \in l_2$$

3. Sufficiency of $\mathcal{S}[\Phi(\mathcal{X})] \subseteq l_2$: Suppose

$$\left(s_j \sqrt{\lambda_j}\right)_j \in l_2$$

and let

$$R_{\mathcal{S}}{}^2 \doteq C_k{}^2 \left\| \left(s_j \sqrt{\lambda_j}\right)_j \right\|_{l_2}^2 < \infty$$

For any

$$\vec{x} \in \mathcal{X}$$

$$\|\mathcal{S}[\Phi(\vec{x})]\|_{l_2}^2 = \sum_{j\in\mathbb{N}} s_j{}^2\lambda_j\left|\psi_j(\vec{x})\right|^2 \leq \sum_{j\in\mathbb{N}} s_j{}^2\lambda_j C_k{}^2 = R_S{}^2$$

Hence

$$\mathcal{S}[\Phi(\mathcal{X})] \subseteq l_2$$

4. Necessity of $\mathcal{S}[\Phi(\mathcal{X})] \subseteq l_2$: Suppose $\left(s_j\sqrt{\lambda_j}\right)_j$ is not in l_2. Hence the sequence

$$(A_n)_n : A_n \doteq \sum_{j=1}^{n} s_j{}^2\lambda_j$$

is unbounded. Now define

$$a_n(\vec{x}) \doteq \sum_{j=1}^{n} s_j{}^2\lambda_j\left|\psi_j(\vec{x})\right|^2$$

Then

$$\|a_n(\vec{x})\|_{\mathrm{L}_1(\mathcal{X})} < \infty$$

due to the normalization condition on $\psi_j(\vec{x})$. However, as

$$\mu(\mathcal{X}) < \infty$$

there exists a set $\widetilde{\mathcal{X}}$ of non-zero measure such that

$$a_n(\vec{x}) \geq \frac{A_n}{\mu(\mathcal{X})}$$

for all

$$\vec{x} \in \widetilde{\mathcal{X}}$$

Combining the expression for $\|\mathcal{S}[\Phi(\vec{x})]\|_{l_2}^2$ from the sufficiency condition with the one for $a_n(\vec{x})$, we obtain

$$\|\mathcal{S}[\Phi(\vec{x})]\|_{l_2}^2 \geq a_n(\vec{x})$$

for all

$$n \in \mathbb{N}$$

and all $\vec{x}$. Since $a_n(\vec{x})$ is unbounded for a set $\widetilde{\mathcal{X}}$ with non-zero measure in $\mathcal{X}$, we can see that

$$\mathcal{S}[\Phi(\mathcal{X})] \nsubseteq l_2$$

Corrigenda to the Mercer Conditions

1. Redefining C_k: It has been pointed out that the condition #2 of Mercer's theorem, i.e.

$$\psi_j \in L_\infty(\mathcal{X})$$

and

$$\sup_j \|\psi_j\|_{L_\infty} < \infty$$

needs adjustment, and therefore C_k redefining as

$$C_k := \sup_j \sup_{\vec{x} \in \mathcal{X}} |\psi_j(\vec{x})|$$

Note that most practically used kernels still have

$$C_k < \infty$$

2. Impact of the C_k Re-definition: This means that only the sufficiency part of the proposition for mapping $\Phi(\mathcal{X})$ onto l_2 remains valid, which is all we need for bounds on the entropy numbers. All of the upper bounds on the entropy numbers still hold as long as C_k redefined as above stays finite.

l_2 Unit Ball → $\mathcal{E}$ Mapping Scaling Operator $\hat{A}$

1. Motivation for the Construction of $\hat{A}$: Once we know that $\Phi(\mathcal{X})$ is contained in the parallelepiped described above, we can use the result of the above proposition to construct a mapping $\hat{A}$ from the unit ball in l_2 to an ellipsoid $\mathcal{E}$ such that

$$\Phi(\mathcal{X}) \subset \mathcal{E}$$

as listed in the schematic below.

2. Metric Space Mapping Schematic:

   a.

$$\mathcal{X} \xrightarrow{\Phi} \Phi(\mathcal{X}) \subset l_2$$

   b.

$$\Phi(\mathcal{X}) \subset l_2 \xrightarrow{\hat{A}^{-1}} U_{l_2} \subset l_2$$

c.

$$U_{l_2} \subset l_2 \xrightarrow{\hat{A}} \mathcal{E} \subset l_2$$

d. Remember that

$$\Phi(\mathcal{X}) \subset l_2 \subset [\mathcal{E} \subset l_2]$$

3. Use of $\hat{A}$: The operator $\hat{A}$ will be useful for computing the entropy numbers of concatenations of operators. Knowing the inverse will allow us to compute the forward operator, and that - can be used to bound the covering numbers of the class of functions.
4. Approach towards constructing $\hat{A}$: We thus seek an operator

$$\hat{A}: l_2 \to l_2$$

such that

$$\hat{A}^{-1}[\Phi(\mathcal{X})] \subset U_{l_2}$$

This means that

$$\mathcal{E} \doteqdot \hat{A} U_{l_2}$$

will be such that

$$\Phi(\mathcal{X}) \subset \mathcal{E}$$

The latter can be ensured by constructing $\hat{A}$ such that

$$\hat{A}: (x_j)_j \mapsto (R_{\hat{A}} \cdot a_j \cdot x_j)_j$$

with

$$R_{\hat{A}}, a_j \in \mathbb{R}^+$$

where a_j and C_k are chosen with respect to a specific kernel, and where

$$R_{\hat{A}} \doteqdot C_k \left\| \left(\frac{\sqrt{\lambda_j}}{a_j} \right)_j \right\|_{l_2}$$

From the necessary/sufficient propositions above, it follows that all those operators $\hat{A}$ for which

$$R_{\hat{A}} < \infty$$

will satisfy the criterion

$$\hat{\mathrm{A}}^{-1}[\Phi(\mathcal{X})] \subset U_{l_2}$$

We call such scaling/inverse operators *admissible*.

Unit Bounding Operator Entropy Numbers

1. Computing the Entropy Numbers of $\hat{A}$: Here we bound the entropy number for the operator $\hat{A}$, and use it to obtain bounds on the entropy numbers for kernel machines like SV machines. We make use of the following theorem due to Gordon, Konig, and Schutt (1987) in the form stated by Carl and Stephani (1990).
2. Diagonal Operator Bounds: Let

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_j \geq \cdots \geq 0$$

be a non-increasing sequence of non-negative numbers, and let

$$\mathcal{D}\vec{x} = (\sigma_1 x_1, \sigma_2 x_2, \ldots, \sigma_j x_j, \ldots)$$

for

$$\vec{x} = (x_1, x_2, \ldots, x_j, \ldots) \in l_p$$

be the diagonal operator from l_p into itself, generated by the sequence $(\sigma_j)_j$, where

$$1 \leq p \leq \infty$$

Then for all

$$n \in \mathbb{N}$$

$$\sup_{j \in \mathbb{N}} \left(\frac{\sigma_1 \sigma_2 \ldots \sigma_j}{n}\right)^{\frac{1}{j}} \leq \epsilon_n(\mathcal{D}) \leq 6 \sup_{j \in \mathbb{N}} \left(\frac{\sigma_1 \sigma_2 \ldots \sigma_j}{n}\right)^{\frac{1}{j}}$$

3. Minimizing the Entropy Numbers by Choosing $\hat{A}$: We can exploit the freedom in choosing $\hat{A}$ to minimize the entropy number, as shown below. This will be a key ingredient in the calculation of the covering numbers of kernel machines and SV classes.
4. Scaling Operators Lemma: Let

$$k: \mathcal{X} \times \mathcal{X}$$

be a Mercer kernel with eigenvalues $(\lambda_s)_s$. Choose

$$a_j > 0$$

for

$$j \in \mathbb{N}$$

such that

$$\left(\frac{\sqrt{\lambda_s}}{a_s}\right)_s \in l_2$$

and define

$$A: (x_j)_j \mapsto (R_A \cdot a_j \cdot x_j)_j$$

with

$$R_A \doteqdot C_k \left\| \left(\frac{\sqrt{\lambda_j}}{a_j}\right)_j \right\|_{l_2}$$

Then

$$\epsilon_n(A: l_2 \to l_2) \leq \sup_{j \in \mathbb{N}} 6C_k \left\| \left(\frac{\sqrt{\lambda_s}}{a_s}\right)_s \right\|_{l_2} \left(\frac{\sigma_1 \sigma_2 \dots \sigma_j}{n}\right)^{\frac{1}{j}}$$

5. Use of the Scaling Operator Lemma: The scaling operator lemma follows immediately by identifying $\mathcal{D}$ in the diagonal bounds statement with $\hat{A}$ in the scaling operator lemma. We can optimize $\left\| \left(\frac{\sqrt{\lambda_s}}{a_s}\right)_s \right\|_{l_2}$ over all possible choices of A to obtain the optimal entropy number for $\hat{A}$. It turns out that the optimum for $\left\| \left(\frac{\sqrt{\lambda_s}}{a_s}\right)_s \right\|_{l_2}$ (i.e., the infimum over $(a_s)_s$) is attainable

when k is a Mercer kernel (Guo, Bartlett, Shawe-Taylor, and Williamson (1999)). Thus, we can minimize the RHS of the scaling operator lemma, as shown below.

6. Optimal Entropy Number for $\hat{A}$: There exists an $\hat{A}$ defined by

$$\hat{A}: (x_j)_j \mapsto (R_{\hat{A}} \cdot a_j \cdot x_j)_j$$

with

$$R_{\hat{A}}, a_j \in \mathbb{R}^+$$

that satisfies

$$\epsilon_n(\hat{A}) \leq \inf_{(a_s)_s: \left(\frac{\sqrt{\lambda_s}}{a_s}\right)_s \in l_2} \sup_{j \in \mathbb{N}} 6C_k \left\| \left(\frac{\sqrt{\lambda_s}}{a_s}\right)_s \right\|_{l_2} \left(\frac{\sigma_1 \sigma_2 \dots \sigma_j}{n}\right)^{\frac{1}{j}}$$

The SVM Operator

1. The SVM Feature Space: To recap, the hypothesis that an SVM generates may be expressed as

$$\langle \vec{w}, \overrightarrow{x^{\sim}} \rangle + b$$

where both $\vec{w}$ and $\overrightarrow{x^{\sim}}$ are defined in the feature space

$$S = Span[\Phi(\mathcal{X})]$$

and

$$b \in \mathbb{R}$$

2. Applying the Kernel Trick: The kernel trick, as introduced in Aizerman, Braverman, and Rozonoer (1964), was successfully employed by Boser, Guyon, and Vapnik (1992) and Cortes and Vapnik (1995) to extend the optimal margin hyper-plane classifier to what is now known as the SV machine. Ignoring b for now, we consider the class

$$\mathcal{F}_\Lambda \doteqdot \{f_{\vec{w}} : \vec{x} \mapsto \langle \vec{w}, \vec{x} \rangle : \vec{x} \in \mathcal{S}, \|\vec{w}\| \leq \Lambda\} \subseteq \mathbb{R}^{\mathcal{S}}$$

   Note that $\mathcal{F}_\Lambda$ depends implicitly on k since $\mathcal{S}$ does.
3. l_∞^m Covering Numbers for $\mathcal{F}_\Lambda$: We seek the l_∞^m covering numbers for $\mathcal{F}_\Lambda$ induced by the kernel in terms of the parameter Λ which is the inverse of the size of the margin in the feature space as defined by the dot-product in $\mathcal{S}$ (see Vapnik and Chervonenkis (1974) and Vapnik (1995) for details).
4. SVM Operator for SV Classes – Definition: We refer to those hypotheses classes that have length constraint on the feature space to be *SV Classes*. We define the operator $\mathcal{T}$ as

$$\mathcal{T} = \mathcal{S}_{\overrightarrow{X_m}^{\sim}} \Lambda$$

   where

$$\Lambda \in \mathbb{R}^+$$

   and the operator $\mathcal{S}_{\overrightarrow{X_m}^{\sim}}$ is defined by

$$\mathcal{S}_{\overrightarrow{X_m}^{\sim}} : l_2 \to l_\infty^m \Rightarrow \mathcal{S}_{\overrightarrow{X_m}^{\sim}} : \vec{w} \mapsto (\langle \overrightarrow{x_1}, \vec{w} \rangle, \ldots, \langle \overrightarrow{x_m}, \vec{w} \rangle)$$

   with

$$\overrightarrow{x_j}^{\sim} \in \Phi(\mathcal{X})$$

for all j. We then seek to compute the entropy numbers of the SV operators.

Maurey's Theorem

1. Introduction: Maurey's theorem is useful for computing the entropy numbers in terms of $\mathcal{T}$ and $\hat{A}$. The original theorem was extended by Carl (1985), and formulated in the form used below by Carl and Stephani (1990).
2. Statement of the Theorem: Let

$$\mathcal{S} \in \mathfrak{L}(\mathcal{H}, l_\infty^m)$$

where $\mathcal{H}$ is a Hilbert Space. Then there exists a constant

$$c > 0$$

such that for all

$$n, m \in \mathbb{N}$$

$$e_n(\mathcal{S}) \leq c\|\mathcal{S}\|\sqrt{\frac{\log\left(1+\frac{m}{n}\right)}{n}}$$

3. n vs. Input Dimensionality: Carl and Stephani (1990) state an additional condition, namely, that

$$n \leq m$$

It turns out that for

$$n > m$$

and even tighter bound holds, and so it is not incorrect to state the result as above (Williamson, Smola, and Scholkopf (2000)). This tighter bound is of little value in learning theory applications, as it corresponds to determining the ϵ-covering number for an extremely small ϵ for which

$$\log \mathcal{N}_m(\epsilon, \mathcal{F}) > m$$

4. Estimate for the Constant c: Williamson, Smola, and Scholkopf (2000) provide proof of a fairly tight bound for the constant of

$$c \leq 103$$

however, there is reason to believe that c should be 1.86, the constant obtained for identity maps from l_2^m to l_∞^m.

5. Maurey's Theorem for Entropy Numbers (instead of Dyadic Entropy): The re-statement of Maurey's theorem in terms of

$$\epsilon_{2^{n-1}}(\mathcal{S}) = e_n(\mathcal{S})$$

under the assumptions above, becomes

$$\epsilon_n(\mathcal{S}) \leq c\|\mathcal{S}\| \sqrt{\frac{\log\left(1 + \frac{m}{1 + \log n}\right)}{1 + \log n}}$$

6. Carl and Stephani's Lemma (Carl and Stephani (1990)): Let E, F, and G be Banach spaces, with

$$\mathcal{R} \in \mathfrak{L}(F, G)$$

and

$$\mathcal{S} \in \mathfrak{L}(E, F)$$

Then for all

$$n, t \in \mathbb{N}$$

a.

$$\epsilon_{nt}(\mathcal{RS}) \leq \epsilon_n(\mathcal{R}) \cdot \epsilon_t(\mathcal{S})$$

b.

$$\epsilon_n(\mathcal{RS}) \leq \epsilon_n(\mathcal{R}) \cdot \|\mathcal{S}\|$$

c.

$$\epsilon_n(\mathcal{RS}) \leq \epsilon_n(\mathcal{S}) \cdot \|\mathcal{R}\|$$

d. Note that b) and c) follow directly from a), and the fact that

$$\epsilon_1(\mathcal{R}) = \|\mathcal{R}\|$$

for all

$$\mathcal{R} \in \mathfrak{L}(F, G)$$

and

$$\epsilon_1(\mathcal{S}) = \|\mathcal{S}\|$$

for all

$$\mathcal{S} \in \mathfrak{L}(E, F)$$

Bounds for SV Classes

1. Setup: Let k be a Mercer kernel, let Φ be induced from

$$\Phi: \mathcal{X} \to l_2 \Longrightarrow \Phi: \vec{x} \mapsto \left(\phi_j(\vec{x})\right)_j \doteq \left(\sqrt{\lambda_j}\psi_j(\vec{x})\right)_j$$

and let

$$\mathcal{T} = \mathcal{S}_{\overrightarrow{X_m}}\Lambda$$

where $\mathcal{S}_{\overrightarrow{X_m}}$ is given from

$$\mathcal{S}_{\overrightarrow{X_m}}: l_2 \to l_\infty^m \Longrightarrow \mathcal{S}_{\overrightarrow{X_m}}: \vec{w} \mapsto (\langle \overrightarrow{x_1}, \vec{w} \rangle, \dots, \langle \overrightarrow{x_m}, \vec{w} \rangle)$$

and

$$\Lambda \in \mathbb{R}^+$$

Let A be defined from

$$\epsilon_n(A) \leq \inf_{(a_s)_s: \left(\frac{\sqrt{\lambda_s}}{a_s}\right)_s \in l_2} \sup_{j \in \mathbb{N}} 6C_k \left\| \left(\frac{\sqrt{\lambda_s}}{a_s}\right)_s \right\|_{l_2} \left(\frac{\sigma_1 \sigma_2 \dots \sigma_j}{n}\right)^{\frac{1}{j}}$$

and suppose

$$\overrightarrow{x_j}^{\sim} \mapsto \Phi(\overrightarrow{x_j})$$

for

$$j = 1, \ldots, m$$

2. The SV Bounds: Then, the entropy numbers of $\mathcal{T}$ satisfy the following inequalities (c here is the same as specified earlier):
    a.
    $$\epsilon_n(\mathcal{T}) \leq c\|A\|\Lambda\sqrt{\frac{\log\left(1+\frac{m}{\log n}\right)}{\log n}}$$
    b.
    $$\epsilon_n(\mathcal{T}) \leq 6\Lambda\epsilon_n(A)$$
    c.
    $$\epsilon_{nt}(\mathcal{T}) \leq 6c\Lambda\sqrt{\frac{\log\left(1+\frac{m}{\log n}\right)}{\log n}}\epsilon_t(A)$$

3. Optimal Bound for $\epsilon_n(\mathcal{T})$: The above set of bound provide several options for bounding $\epsilon_n(\mathcal{T})$. The optimal inequality bound to use depends on the rate of decay of the eigenvalues of k. The result gives effective bounds on $\mathcal{N}_m(\epsilon, \mathcal{F}_\Lambda)$ since

$$\epsilon_n(\mathcal{T}: l_2 \to l_\infty^m) \leq \epsilon_0 \Longrightarrow \mathcal{N}_m(\epsilon, \mathcal{F}_\Lambda) \leq n$$

4. Factorization of $\mathcal{T}$ to the Upper Bound $\epsilon_n(\mathcal{T})$: We will us the following factorization of $\mathcal{T}$ to upper-bound $\epsilon_n(\mathcal{T})$.

a.

$$U_{l_2} \subset l_2 \overset{\mathcal{T}}{\to} l_\infty^m$$

b.

$$U_{l_2} \subset l_2 \overset{\Lambda}{\to} \Lambda U_{l_2} \subset l_2$$

c.

$$\Lambda U_{l_2} \subset l_2 \xrightarrow{\mathcal{S}_{\Phi(\vec{X}_m)}} l_\infty^m$$

d.

$$U_{l_2} \subset l_2 \overset{\mathrm{A}}{\to} \Lambda\mathcal{E} \subset l_2$$

e.

$$\Lambda\mathcal{E} \subset l_2 \xrightarrow{\mathcal{S}_{\mathrm{A}^{-1}\Phi(\vec{X}_m)}} l_\infty^m$$

5. Description of the Factorization Schematic: The statement a) follows from the definition of $\mathcal{T}$. The fact that b) – d) commutes, as in e), stems from the fact that since A is diagonal, it is self-adjoint, and so for any

$$\overrightarrow{x^{\sim}} \in \mathcal{S}$$

$$\langle \vec{w}, \overrightarrow{x^{\sim}} \rangle = \langle \vec{w}, A\mathrm{A}^{-1}\overrightarrow{x^{\sim}} \rangle = \langle A\vec{w}, \mathrm{A}^{-1}\overrightarrow{x^{\sim}} \rangle$$

6. Alternate Representation of $\mathcal{T}$: Instead of computing the entropy number of

$$\mathcal{T} = \mathcal{S}_{\overrightarrow{X_m^{\sim}}}\Lambda$$

directly, which is difficult or wasteful, as the bound on $\mathcal{S}_{\overrightarrow{X_m^{\sim}}}$ does not take into account that

$$\overrightarrow{\tilde{x}} \in \mathcal{E}$$

but simply assumes that

$$\overrightarrow{\tilde{x}} \in \rho U_{l_2}$$

for some

$$\rho > 0$$

we will represent $\mathcal{T}$ as

$$\mathcal{S}_{\mathrm{A}^{-1}\overrightarrow{\tilde{X_m}}}\mathrm{A}\Lambda$$

This is more efficient as we constructed A such that

$$A^{-1}\Phi(\mathcal{X}) \subseteq U_{l_2}$$

filling a larger proportion of it than $\frac{1}{\rho}\Phi(\mathcal{X})$ does.

7. Proof of the SV Bounds: By construction of A, and due to the Cauchy-Schwartz inequality, we have

$$\left\|\mathcal{S}_{\mathrm{A}^{-1}\overrightarrow{\tilde{X_m}}}\right\| \leq 1$$

Thus, applying the Carl and Stephani's lemma to the factorization of this version of $\mathcal{T}$, and using the Maurey's theorem, we get the SV bounds.

Asymptotic Rates of Decay for $\epsilon_n(\mathcal{T})$

1. Introduction: Eventually we seek asymptotic rates of decay for $\epsilon_n(A)$ – in fact, we provide non-asymptotic results with explicitly evaluable constants. It is thus of some interest to give overall asymptotic rates of decay of $\epsilon_n(\mathcal{T})$ in terms of $\epsilon_n(A)$. By asymptotic here we mean asymptotic in n; this corresponds to asking how $\mathcal{N}_m(\epsilon, \mathcal{F})$ scales as

$$\epsilon \to 0$$

for fixed m.

2. Rates Bounds on $\epsilon_n(\mathcal{T})$: Let k be a Mercer kernel and suppose that A is the scaling operator defined for

$$j \in \mathbb{N}$$

such that

$$\left(\frac{\sqrt{\lambda_s}}{a_s}\right)_s \in l_2$$

and define

$$A: (x_j)_j \mapsto (R_A \cdot a_j \cdot x_j)_j$$

with

$$R_A \doteq C_k \left\| \left(\frac{\sqrt{\lambda_j}}{a_j}\right)_j \right\|_{l_2}$$

such that for

$$j \in \mathbb{N}$$

$$a_j > 0$$

and

$$\left(\frac{\sqrt{\lambda_s}}{a_s}\right)_s \in l_2$$

a. If

$$\epsilon_n(A) = \mathcal{O}\left(\left[\frac{1}{\log n}\right]^{\alpha}\right)$$

for some

$$\alpha > 0$$

then for fixed m

$$\epsilon_n(\mathcal{T}) = \mathcal{O}\left(\left[\frac{1}{\log n}\right]^{\alpha+\frac{1}{2}}\right)$$

b. If

$$\log \epsilon_n(A) = \mathcal{O}\left(\left[\frac{1}{\log n}\right]^{\beta}\right)$$

for some

$$\beta > 0$$

then for fixed m

$$\epsilon_n(\mathcal{T}) = \mathcal{O}\left(\left[\frac{1}{\log n}\right]^{\beta}\right)$$

3. Proof Step #1 Splitting the Index n: From Maurey's theorem we know that

$$\epsilon_n(\mathcal{S}) = \mathcal{O}\left(\sqrt{\frac{1}{\log n}}\right)$$

Using the 3rd SV Class bound, ignoring the constants and ignoring the constants and assuming m is fixed, we split the index n in the following way:

$$n = n^{\tau}n^{1-\tau}$$

with

$$\tau \in (0,1)$$

4. Proof Step #2 - Bounds for the First Case: For the first case, this yields

$$\epsilon_n(\mathcal{T}) \leq \epsilon_{n^{\tau}}(\mathcal{S})\epsilon_{n^{1-\tau}}(A) = \mathcal{O}\left(\sqrt{\frac{1}{\log n^{\tau}}}\right) \cdot \mathcal{O}\left(\left[\frac{1}{\log n^{1-\tau}}\right]^{\alpha}\right) = \mathcal{O}\left(\frac{1}{\sqrt{\tau}(1-\tau)^{\alpha}}\left[\frac{1}{\log n}\right]^{\alpha+\frac{1}{2}}\right)$$
$$= \mathcal{O}\left(\left[\frac{1}{\log n}\right]^{\alpha+\frac{1}{2}}\right)$$

5. Proof Step #3 - Bounds for the Second Case: For the first case, this yields

$$\epsilon_n(\mathcal{T}) \le \epsilon_{n^\tau}(\mathcal{S})\epsilon_{n^{1-\tau}}(A) = \mathcal{O}\left(\sqrt{\frac{1}{\tau \log n}}\right) \cdot \mathcal{O}\left(\left[\frac{1}{(1-\tau)\log n}\right]^\beta\right) = \mathcal{O}\left(\left[\frac{1}{\log n}\right]^\beta\right)$$

6. Impact of the Rate Bounds on $\epsilon_n(\mathcal{T})$: These bounds show that, in the first case, Maurey's theorem allows an exponent in the entropy number of $\mathcal{T}$, whereas in the 2nd it affords none, since the entropy numbers decay so fat anyway. Maurey's theorem may still help in that case for non-asymptotic n.
7. Improvements over Maurey's Theorem: In a nutshell, we can always obtain rates of convergence better than those due to Maurey's theorem because we are not dealing with *arbitrary* mappings into infinite-dimensional spaces. In fact, for logarithmic dependence of $\epsilon_n(\mathcal{T})$ on n, the effect of kernel is so strong that it completely dominates the $\frac{1}{\sqrt{n}}$ for arbitrary Hilbert spaces. An example of such a kernel is $k(x,y) = e^{-(x-y)^2}$.

References

- Aizerman, M. A., F. M. Braverman, and L. I. Rozonoer (1964): Theoretical Foundations of the Potential Function Method in Pattern Recognition Learning *Automation Remote Control* **25** 821-837.
- Ash, R. (1965): *Information Theory* **Interscience** New York.
- Boser, B. E., I. M. Guyon, and V. N. Vapnik (1992): A Training Algorithm for Optimal Margin Classifiers, in: *5th Annual ACM Workshop on COLT (D. Haussler, editor)* **ACM Press** Pittsburgh PA.
- Carl, B. (1985): Inequalities of the Bernstein-Jackson Type and the Degree of Compactness of Operators in Banach Spaces *Ann. Inst. Fourier* **35 (3)** 79-118.
- Carl, B. and I. Stephani (1990): *Entropy, Compactness, and the Approximation of Operators* **Cambridge University Press** Cambridge UK.
- Cortes, C., and V. N. Vapnik (1995): Support Vector Networks *Machine Learning* **20** 273-297.

- Gordon, Y., H. Konig, and C. Schutt (1987): Geometric and Probabilistic Estimates for Entropy and Approximation Numbers of Operators *Journal of Approximation Theory* **49** 219-239.
- Guo, Y., P. L. Bartlett, J. Shawe-Taylor, and R. C. Williamson (1999): Covering Numbers for Support Vector Machines, in: *Proceedings of the 12th Annual Conference on Computational Learning Theory* 267-277 **ACM** New York.
- Konig, H. (1986): *Eigenvalue Distribution of Compact Operators* **Birkhauser** Basel Switzerland.
- Smola, A. J., A. Elisseff, B. Scholkopf, and R. C. Williamson (2000): Entropy Numbers for Convex Combinations and mlps, in: *Advances in Large Margin Classifiers (A. Smola, P. Bartlett, B. Scholkopf, and D. Schuurmans, editors)* **MIT Press** Cambridge MA.
- Vapnik, V. N., and A. Chervonenkis (1974): *Theory of Pattern Recognition (in Russian)* **Nauka** Moscow.
- Vapnik, V. N. (1995): *The Nature of Statistical Learning Theory* **Springer-Verlag** New York.
- Williamson, R. C., A. J. Smola, and B. Scholkopf (2000): Entropy Numbers of Linear Function Classes, in: *Proceedings of the 13th Annual Conference on Computational Learning Theory (N. Cesa-Bianchi and S. Goldman, editors)* **ACM** New York.

Discrete Spectra of Convolution Operators

Kernels with Compact/Non-compact Support

1. Introduction: The results presented above show that if one knows the eigenvalue sequence $(\lambda_i)_i$ of a compact operator, one can bound its entropy numbers. While it is always possible to assume that the data fed into the SV machine have bounded support, the same cannot be said of any kernel; in fact, a commonly used kernel is

$$k(x, y) = e^{-(x-y)^2}$$

   which has non-compact support.
2. Integral Operator Induced Continuous Spectrum: As in the kernel above, the induced integral operator

$$[\mathcal{T}_k f](x) = \int_{-\infty}^{+\infty} k(x, y) f(x) dx$$

   has a continuous spectrum (i.e., a non-denumerable infinity of eigenvalues), and, thus, $\mathcal{T}_k$ is not compact (Ash (1965)). The question then arises is whether we can make use of such kernels in SV machines and still obtain generalization error bounds of the form above.
3. Convolution Kernels with Compact Support: It is well-known that the eigenvalue decay of any convolution operator defined on a compact set via a kernel having compact support can decay no faster than

$$\lambda_j = \Omega\left(e^{-j^2}\right)$$

(Widom (1963)), and thus if one seeks rapid decay of eigenvalues (and, therefore, small entropy numbers), one must seek convolution kernels with non-compact support.

4. Kernels with Compact Support: We first consider the case where the support of

$$k \subseteq [-a, +a]$$

for some

$$a < \infty$$

We also further suppose that the data points $\vec{x}_j$ satisfy

$$\vec{x}_j \in [-b, +b]$$

5. Kernels with Compact Support applied to SV Hypothesis: If $k(\cdot,\cdot)$ is a convolution kernel, i.e.

$$k(x, y) = k(x - y, 0)$$

then the SV hypothesis $h_k(\cdot)$ can be written as

$$h_k(x) = \sum_{j=1}^{m} \alpha_j k(x, \vec{x}_j) = \sum_{j=1}^{m} \alpha_j k_v(x, \vec{x}_j) \doteqdot h_{k_v}(x)$$

for

$$v \geq 2(a + b)$$

where $k_v(\cdot,\cdot)$ is the v-periodic extension of $k(\cdot,\cdot)$ - analogously

$$k_v(x,y) = k_v(x-y,0)$$

$$k_v(x,0) \doteqdot \sum_{j=-\infty}^{j=+\infty} k(x-jv,0)$$

Eigenvalues of $\mathcal{T}_{k_v}$

1. The d-dimensional Fourier Transform Operator: The d-dimensional Fourier transform is defined by

$$F: L_2(\mathbb{R}^d) \to L_2(\mathbb{R}^d) \Longrightarrow F_f(\vec{w}) \doteqdot \frac{1}{(2\pi)^{\frac{d}{2}}} \int_{\mathbb{R}^d}^{\mathbb{R}^d} e^{-i\langle \vec{w},\vec{x}\rangle} f(\vec{x}) d\vec{x}$$

   Then its inverse is defined by

$$F^{-1}: L_2(\mathbb{R}^d) \to L_2(\mathbb{R}^d) \Longrightarrow {F^{-1}}_f(\vec{x}) \doteqdot \frac{1}{(2\pi)^{\frac{d}{2}}} \int_{\mathbb{R}^d}^{\mathbb{R}^d} e^{i\langle \vec{w},\vec{x}\rangle} f(\vec{w}) d\vec{w}$$

   From the above, F is an isometry on $L_2(\mathbb{R}^d)$.
2. The Approach: We now relate the eigenvalues of $\mathcal{T}_{k_v}$ to the Fourier transform of $k(\cdot,\cdot)$. We do so for the case of

$$d = 1$$

   and state the general cases later.
3. Fourier Transform of the Symmetric Convolution Kernel - Statement: Let

$$k: \mathbb{R}^1 \to \mathbb{R}^1$$

be a symmetric convolution kernel, let

$$K(\vec{w}) = F_f(\vec{w})$$

denote the Fourier transform of $k(\cdot,\cdot)$, and k_v denote the v-periodical kernel derived from k (also assume that k_v exists). Then k_v has a representation as a Fourier series with

$$w_0 \doteqdot \frac{2\pi}{v}$$

and

$$k_v(x-y) = \sum_{j=-\infty}^{j=+\infty} \frac{\sqrt{2\pi}}{v} K(jw_0) e^{ijw_0(x-y)}$$

$$= \frac{\sqrt{2\pi}}{v} K(0) + \sum_{j=-\infty}^{j=+\infty} \frac{\sqrt{8\pi}}{v} K(jw_0) cos\big(jw_0(x-y)\big)$$

Moreover, the eigenvalues λ_j of $\mathcal{T}_{k_v}$ satisfy

$$\lambda_j = \sqrt{2\pi} K(jw_0)$$

for

$$j \in \mathbb{Z}$$

and

$$C_k = \sqrt{\frac{2}{v}}$$

4. Proof Step #1 - Fourier Series Coefficients of k_v: Clearly, the Fourier series coefficients K_j of k_v exist (as k_v exists), with

$$K_j \doteqdot \frac{1}{\sqrt{v}} \int_{-\frac{v}{2}}^{\frac{v}{2}} e^{-ijw_0 x} k_v(x) dx$$

and therefore, by the definition of k_v and the existence of $K(w)$, we conclude

$$K_j \doteqdot \frac{1}{\sqrt{v}} \int_{-\frac{v}{2}}^{\frac{v}{2}} \sum_{l=-\infty}^{l=+\infty} \left[e^{-ijw_0 x} k(x - lv) \right] dx = \frac{1}{\sqrt{v}} \sum_{l=-\infty}^{+\infty} \left[\int_{-\frac{v}{2}}^{\frac{v}{2}} e^{-ijw_0 x} k(x - lv) dx \right] = \sqrt{\frac{2\pi}{v}} K(jw_0)$$

5. Proof Step #2 - The Orthogonal L_2 Basis: This and the fact that

$$\left\{ x \mapsto \frac{1}{\sqrt{v}} e^{ijw_0 x} : j \in \mathbb{Z} \right\}$$

form an orthogonal basis in $L_2\left(\left[-\frac{v}{2}, +\frac{v}{2}\right], \mathbb{C}\right)$ proves that

$$k_v(x - y) = \frac{\sqrt{2\pi}}{v} K(0) + \sum_{j=-\infty}^{j=+\infty} \frac{\sqrt{8\pi}}{v} K(jw_0) cos\left(jw_0(x - y)\right)$$

6. Proof Step #3 - Choice of the Trigonometric Basis Function: Furthermore, we are interested in real-valued basis functions for $k(x - y)$. The functions

$$\psi_0(x) \doteqdot \frac{1}{\sqrt{v}}$$

$$\psi_j(x) \doteqdot \sqrt{\frac{2}{v}}\, cos(jw_0 x)$$

and

$$\psi_{-j}(x) \doteqdot \sqrt{\frac{2}{v}}\, sin(jw_0 x)$$

for all

$$j \in \mathbb{N}$$

satisfy

$$\|\psi_j\|_{L_2} = 1$$

$$j \in \mathbb{Z}$$

and form an eigen-system of the integral operator defined by k_v with the corresponding eigenvalues $\sqrt{2\pi}K(jw_0)$. Finally, one can see that

$$C_k = \sqrt{\frac{2}{v}}$$

by computing the maximum over

$$j \in \mathbb{N}$$

and

$$x \in \left[-\frac{v}{2}, +\frac{v}{2}\right]$$

7. Extension to non-compact $\mathcal{T}_k$ Support: Thus, even though $\mathcal{T}_k$ may not be compact, $\mathcal{T}_{k_v}$ can be - if

$$\left(K(jw_0)\right)_{j\in\mathbb{N}} \subset l_2$$

for example. The above Fourier transform can be applied whenever we can form $k_v(\cdot)$ from $k(\cdot)$. Clearly

$$k(x) = \mathcal{O}(x^{-1-\epsilon})$$

for some

$$\epsilon > 0$$

suffices to ensure that the sum

$$k_v(x, 0) \doteqdot \sum_{j=-\infty}^{j=+\infty} k(x - jv, 0)$$

converges.

8. Discrete, Periodic Spectrum: In conclusion, in order to obtain a discrete spectrum, one needs to use a periodic kernel. For a given problem, one can always periodize a non-periodic kernel in a way that changes the eventual hypothesis in an arbitrarily small way. The results that we produce below can then be applied.

Choosing v

1. Asymptotics of $K(w)$: From the Riemann-Lebesgue lemma, it is clear that for integrable $k(\cdot)$ of bounded variations (surely any kernel one would choose would satisfy that criterion), one has

$$K(w) = \mathcal{O}\left(\frac{1}{w}\right)$$

2. Tradeoff in the Choice of v: There is a tradeoff in choosing v in that, for large enough w, $K(w)$ is a decreasing function of w (at least as fast as $\frac{1}{w}$), and thus, from

$$\lambda_j = \sqrt{2\pi}K(jw_0)$$

we get that

$$\lambda_j = \sqrt{2\pi}K\left(\frac{2\pi j}{v}\right)$$

is an increasing function of v. This suggests that one should choose a small value of v. But a small v will lead to high empirical error (as the kernel "wraps around" and its localization properties are lost, i.e., its $VCDim$ becomes high) and large C_k.

3. An Approach for picking v: There are several approaches for picking a value for v. One obvious way is to *a priori* pick some

$$\overline{\epsilon} > 0$$

and choose the smallest v such that

$$|k(x) - k_v(x)| \leq \overline{\epsilon}$$

for all

$$x \in \left[-\frac{v}{2}, +\frac{v}{2}\right]$$

Thus, one would obtain a hypothesis $h_{k_v}(x)$ uniformly within $C\overline{\epsilon}$ of $h_k(x)$ where

$$\sum_{j=1}^{m} |\alpha_j| \leq C$$

Extension to d-dimension

1. <u>The Statement</u>: Assume that $k(\vec{x})$ is v-periodic in each direction

   $$[\vec{x} = (x_1, \ldots, x_d)]$$

   we get

   $$\lambda_j = (2\pi)^{\frac{d}{2}} K(\vec{j} w_0) = (2\pi)^{\frac{d}{2}} K(w_0 \|\vec{j}\|)$$

   for radially symmetric k, and finally for the eigen-functions we get

   $$C_k = \left(\frac{2}{v}\right)^{\frac{d}{2}}$$

2. <u>Kernel Bandwidth Choice</u>: Here we examine how a difference choice for the bandwidth of the kernel, i.e., after letting

   $$k_\sigma(\vec{x}) \doteqdot \sigma^d k(\sigma \vec{x})$$

affects the eigen-spectrum of the corresponding operator. We have

$$K_\sigma(\vec{w}) = K\left(\frac{\vec{w}}{\sigma}\right)$$

hence scaling a kernel by σ means more densely spaced eigenvalues in the spectrum of the integral operator $\mathcal{T}_{k_\sigma}$.

References

- Ash, R. (1965): *Information Theory* **Interscience** New York.
- Widom, H: (1963): Asymptotic Behavior of Eigenvalues of certain Integral Operators *Transactions of the American Mathematical Society* **109** 278-295.

Covering Numbers for Given Decay Rates

Asymptotic/Non-asymptotic Decay of Covering Numbers

1. Introduction: Here we show how the asymptotic behavior of

$$\epsilon_n(A: l_2 \to l_2)$$

where A is the scaling operator introduced earlier, depends on the eigenvalues of $\mathcal{T}_k$.

2. Comparison with Prosser's Analysis: A similar analysis has been carried out by Prosser (1966) in order to compute the entropy numbers of integral operators. However, all of his operators mapped into $L_2(\mathcal{X}, \mathbb{C})$. Furthermore, while the analysis here states the propositions as asymptotic results as Prosser (1966) does, the proofs provide non-asymptotic details with explicit constants.
3. Need to work with Sorted Eigenvalues: Note that we need to sort the eigenvalues in a non-increasing manner due to the requirements of the proposition for $\hat{A}$, i.e.

$$\epsilon_n(\hat{A}) \leq \inf_{(a_s)_s: \left(\frac{\sqrt{\lambda_s}}{a_s}\right)_s \in l_2} \sup_{j \in \mathbb{N}} 6C_k \left\| \left(\frac{\sqrt{\lambda_s}}{a_s}\right)_s \right\|_{l_2} \left(\frac{a_1 a_2 \dots a_j}{n}\right)^{\frac{1}{j}}$$

If the eigenvalues were unsorted, one could obtain far too small numbers for the geometrical mean of $\lambda_1, \lambda_2, \dots, \lambda_j$.

4. Non-degeneracy of Eigenvalues: Many 1D kernels have non-degenerate systems of eigenvalues in which case it is straight-forward to explicitly calculate the geometric means of the eigenvalues. While all of the instances we treat here are convolution kernels, i.e.

$$k(x, y) = k(x - y)$$

there is nothing in the formulations of the propositions themselves that requires this. When we consider the d-dimensional case, we shall see that with rotationally invariant kernels, degenerate systems of eigenvalues are generic.

5. Explicit Decay of $(\lambda_j)_j$: We shall consider the specific cases where $(\lambda_j)_j$ decays asymptotically in some polynomial or exponential degree. In this case, we choose the sequence A for which we can evaluate

$$\epsilon_n(A) \leq \inf_{(a_s)_s:\left(\frac{\sqrt{\lambda_s}}{a_s}\right)_s \in l_2} \sup_{j \in \mathbb{N}} 6C_k \left\|\left(\frac{\sqrt{\lambda_s}}{a_s}\right)_s\right\|_{l_2} \left(\frac{a_1 a_2 \ldots a_j}{n}\right)^{\frac{1}{j}}$$

explicitly. In what follows, by the eigenvalues of kernel k we refer to the sorted eigenvalues of the induced integral operator $\mathcal{T}_k$.

Polynomial Eigenvalue Decay

1. Eigenvalue Decay Statement: Let k be a Mercer kernel with eigenvalues

$$\lambda_j = \mathcal{O}(j^{-1-\alpha})$$

for some

$$\alpha > 0$$

Then, for any

$$\delta \in \left(0, \frac{\alpha}{2}\right)$$

we have

$$\epsilon_n(A: l_2 \to l_2) = \mathcal{O}\left([\log n]^{-\frac{\alpha}{2}+\delta}\right)$$

$$\epsilon_n(A: l_2 \to l_2) = \Omega\left([\log n]^{-\frac{\alpha}{2}}\right)$$

An example of such a kernel is

$$k(x) = e^{-x}$$

2. Proof Step #1 - Bounding $\left\| \left(\frac{\sqrt{\lambda_j}}{a_j} \right)_j \right\|_{l_2}$: Since

$$\lambda_j = \mathcal{O}(j^{-1-\alpha})$$

there exists some

$$\beta \in \mathbb{R}^+$$

with

$$\lambda_j = \beta^2 j^{-1-\alpha}$$

In this case, all sequences

$$(a_j)_j = \left(j^{-\frac{\tau}{2}}\right)_j$$

with

$$0 < \tau < \alpha$$

lead to an admissible scaling property. Thus, one has

$$\left\| \left(\frac{\sqrt{\lambda_j}}{a_j} \right)_j \right\|_{l_2} = \beta \left\| \left(j^{\frac{\tau-\alpha-2}{2}} \right)_j \right\|_{l_2} = \beta \sqrt{\varsigma(\alpha - \tau + 1)}$$

where $\varsigma(\cdot)$ is the Riemann's zeta function. We recall that one can bound $\varsigma(\cdot)$ by

$$x + \gamma \leq \varsigma\left(1 + \frac{1}{x}\right) \leq x + 1$$

where γ is Euler's constant.

3. Proof Step #2 - Bounding $\left(a_1 a_2 \dots a_j\right)^{\frac{1}{j}}$: We now evaluate

$$\left(a_1 a_2 \dots a_j\right)^{\frac{1}{j}} = \left(\prod_{s=1}^{j} s^{-\frac{\tau}{2}} \right)^{\frac{1}{j}} = (j!)^{-\frac{\tau}{2j}} = \Gamma(j+1)^{-\frac{\tau}{2j}}$$

The Gamma function $\Gamma(j+1)$ can be bound as follows: for

$$j > 1$$

$$\log j - 1 \leq \frac{1}{j}\Gamma(j+1) \leq \log j$$

4. Proof Step #3 - Bounds on $\epsilon_n(A)$: Thus, one may bound $\epsilon_n(A)$ as

$$\epsilon_n(A) \geq C_k \beta \inf_{\tau \in (0,\alpha)} \sup_{j \in \mathbb{N}} n^{-\frac{1}{j}} j^{-\frac{\tau}{2}} \sqrt{\frac{1}{\alpha - \tau} + \gamma}$$

and

$$\epsilon_n(A) \leq 6C_k \beta \inf_{\tau \in (0,\alpha)} \sup_{j \in \mathbb{N}} n^{-\frac{1}{j}} e^{\frac{\tau}{2}} j^{-\frac{\tau}{2}} \sqrt{\frac{1}{\alpha - \tau} + 1}$$

5. Proof Step #4 - Re-casting $\sup_{j \in \mathbb{N}}$: In order to avoid unneeded technicalities, we replace

$$\sup_{j \in \mathbb{N}}$$

by

$$\sup_{j \in [1,\infty)}$$

This is no problem when computing the upper bound, but it is an issue for the lower bound. In fact, as is shown in Williamson, Shawe-Taylor, Scholkopf, and Smola (1999)

$$a_{j^*+1} \leq \sup_{j \in \mathbb{N}} \left(\frac{a_1 a_2 \ldots a_j}{n}\right)^{\frac{1}{j}} \leq a_{j^*}$$

for that particular j^* where

$$\sup_{j \in \mathbb{N}}$$

is actually obtained. Hence, the maximum quotient

$$\frac{a_{j+1}}{a_j}$$

which in the present case is $2^{-\frac{\tau}{2}}$, determines the value by which the bound has to be lowered in order to obtain a true lower bound. Therefore, since $j^{-\frac{\tau}{2}}$ on $[1, \infty)$ is within a constant factor of $2^{-\frac{\tau}{2}}$ of its corresponding values on the integer domain $\mathbb{N}$ (the biggest discrepancy being at $[1, 2]$), we may safely ignore this concern.

6. Proof Step #5 - Estimating $\sup_{j \in [1, \infty)}$: Now we can compute

$$\sup_{j \in [1, \infty)} n^{-\frac{1}{j}} j^{-\frac{\tau}{2}} = \sup_{j \in [1, \infty)} e^{-\frac{1}{j}\log n - \frac{\tau}{2}\log j} = \left(\frac{2e \log n}{\tau}\right)^{-\frac{\tau}{2}}$$

The maximum of this argument is obtained for

$$j = \frac{2 \log n}{\tau}$$

hence the above supremum holds for all

$$\log n \geq \frac{\tau}{2}$$

which is fine, since we want to compute the bounds on $\epsilon_n(A)$ as

$$n \to \infty$$

7. Proof Step #6 - Lower Bounds on $\epsilon_n(A)$: For lower bounds on $\epsilon_n(A)$ we obtain

$$\epsilon_n(A) \geq C_k\beta\,(2e)^{-\frac{\tau}{2}} \inf_{\tau\,\in\,(0,\alpha)} \sqrt{\frac{1}{\alpha-\tau}+\gamma}\,\left(\frac{2\log n}{\tau}\right)^{-\frac{\tau}{2}}$$

$$\geq C_k\beta\,(2e)2^{-\frac{\tau}{2}} \inf_{\tau\,\in\,(0,\alpha)} \sqrt{\frac{1}{\alpha-\tau}+\gamma} \inf_{\tau\,\in\,(0,\alpha)} \left(\frac{2\log n}{\tau}\right)^{-\frac{\tau}{2}}$$

$$\geq C_k\beta\,(2e)2^{-\frac{\tau}{2}} \sqrt{\frac{1}{\alpha}+\gamma}\,\left(\frac{2\log n}{\alpha}\right)^{-\frac{\alpha}{2}}$$

This shows that $\epsilon_n(A)$ is always bounded from below by

$$\Omega\left([\log n]^{-\frac{\alpha}{2}}\right)$$

8. Proof Step #7 - Upper Bounds on $\epsilon_n(A)$: Computation of the upper bound is slightly more involved, since one has to evaluate

$$\epsilon_n(A) \leq 6C_k\beta \inf_{\tau\,\in\,(0,\alpha)} \sqrt{\frac{1}{\alpha-\tau}+1}\,\left(\frac{2\log n}{\tau}\right)^{-\frac{\tau}{2}}$$

Clearly, for any fixed

$$\tau \in (0,\alpha)$$

we are able to get a rate of

$$\epsilon_n(A) = \mathcal{O}\left([\log n]^{-\frac{\tau}{2}}\right)$$

Thus, the statement follows. For practical purposes, a good approximation to inf can be found as

$$\frac{1}{\alpha - \tau} = \log(2 \log n)$$

by computing the derivative of the argument of the inf with respect to τ, and dropping all terms independent of τ and n. However, numerical minimization of the arguments to inf is more advisable when small values of $\epsilon_n(A)$ are crucial.

Summation and Integration of Non-decreasing Functions

1. Summation and Integration in $\mathbb{R}$ - Lemma: Suppose

$$\mathbb{R} \to \mathbb{R}$$

is an integrable, non-increasing function. Then the following inequality holds for any

$$a \in \mathbb{Z}$$

$$\int_a^{\infty} f(x)dx \leq \sum_{n=a}^{\infty} f(n) \leq \int_{a-1}^{\infty} f(x)dx$$

2. Proof of the Lemma: The proof relies on the fact that $f(n) \geq \int_n^{n+1} f(n)dn \geq f(n+1)$ due to the monotonicity of f and the decomposition of the integral $\int_0^{\infty}(\cdots) \Longrightarrow \sum_{n=0}^{\infty} \int_n^{n+1}(\cdots)$. The lemma is a direct consequence thereof.

Exponential Polynomial Decay

1. Introduction: The exponential polynomial decay covers a wide range of practically used kernels, namely, those with polynomial decay in their eigenvalues. For instance, the Gaussian kernel

$$k(x) = e^{-x^2}$$

has exponential quadratic decay in λ_i. The damped harmonic oscillator kernel

$$k(x) = \frac{1}{1 + x^2}$$

is another example, this time with just exponential linear decay in its eigenvalues.

2. Exponential Polynomial Decay Bound - Proposition: Suppose k is a Mercer kernel with

$$\lambda_j = \mathcal{O}\left(e^{-\alpha j^p}\right)$$

for some

$$\alpha, p > 0$$

Then

$$|\log \epsilon_n(A: l_2 \to l_2)| = \mathcal{O}\left([\log n]^{\frac{p}{p+1}}\right)$$

3. Proof of the Proposition - Step #1: Bounding $\left\| \left(\frac{\sqrt{\lambda_j}}{a_j} \right)_j \right\|_{l_2}$ => Since

$$\lambda_j = \mathcal{O}\left(e^{-\alpha j^p}\right)$$

there exists some

$$\beta \in \mathbb{R}^{+}$$

with

$$\lambda_j \leq \beta^2 e^{-\alpha j^p}$$

As in the polynomial decay scenario, we use the series

$$\left(a_j\right)_j = \left(e^{-\frac{\tau}{2}j^p}\right)_j$$

By applying the lemma on summation and integration on $\mathbb{R}$, we have that, for any

$$\tau \in (0, \alpha)$$

$$\left\|\left(\frac{\sqrt{\lambda_j}}{a_j}\right)_j\right\|_{l_2} = \beta\sqrt{\sum_{j=0}^{\infty} e^{(\tau-\alpha)j^p}} \begin{cases} \leq \beta\sqrt{1 + \frac{\Gamma\left(\frac{1}{p}\right)}{p(\alpha-\tau)^{\frac{1}{p}}}} \\ \geq \beta\sqrt{\frac{\Gamma\left(\frac{1}{p}\right)}{p(\alpha-\tau)^{\frac{1}{p}}}} \end{cases}$$

4. Proof Step #2: Bounding the Diagonal Entries of A => We apply a similar bound to the product of the first j entries of the scaling operator A:

$$\left[a_1 a_2 \ldots a_j\right]^{\frac{1}{j}} = e^{-\frac{\tau}{2j}\sum_{s=1}^{j} s^p} \begin{cases} \geq e^{-\frac{\tau}{2(p+1)}j^p} \\ \leq e^{-\frac{\tau}{2(p+1)}j^p} + \frac{\tau}{2j(p+1)} \\ \leq e^{-\frac{\tau}{2(p+1)}j^p} + \frac{\tau}{2(p+1)} \quad \forall\, j \in \mathbb{N} \end{cases}$$

5. Proof Step #3: Bounding the Suprema => Next we compute

$$\sup_{j \in \mathbb{N}} n^{-\frac{1}{j}} e^{-\frac{\tau}{2(p+1)} j^p} = \sup_{j \in \mathbb{N}} e^{-\frac{1}{j}\log n - \frac{\tau}{2(p+1)} j^p}$$

Differentiating the exponent with respect to j leads to

$$\frac{1}{j^2}\log n - \frac{\tau p}{2(p+1)} j^{p-1} = 0 \Rightarrow \frac{\log n}{j} = \frac{\tau}{2(p+1)} j^p$$

and thus

$$\sup_{j \in [1,\infty)} n^{-\frac{1}{j}} e^{-\frac{\tau}{2(p+1)} j^p} = e^{-\frac{\tau}{2(p+1)}} \left[\frac{p+1}{p}\log n\right]^{\frac{p}{p+1}}$$

6. Proof Step #4: Replacing the Suprema Domain => Replacing the domain from

$$\sup_{j \in \mathbb{N}}$$

to

$$\sup_{j \in [1,\infty)}$$

is not a problem when it comes to computing the upper bounds on $\epsilon_n(A)$. As for the lower bounds, again, a similar reasoning to that in the polynomial decay would have to be applied. Thus, the problem reduces to bounding the quotient

$$\frac{a_{j^*+1}}{a_{j^*}}$$

where j^* is the variable for which

$$\sup_{j \in \mathbb{N}}$$

is attained. However, the quotient can only be bounded by

$$e^{-\frac{\tau p}{2} j^{p-1}}$$

Fortunately, this is of lower order than the remaining terms, hence it will not change the *rate* of the lower bounds.

7. Proof Step #5: Lower Bound of $\epsilon_n(A)$ =>

$$\epsilon_n(A) \geq C_k \beta \inf_{\tau \in (0,\alpha)} \sqrt{\frac{\Gamma\left(\frac{1}{p}\right)}{p(\alpha-\tau)^{\frac{1}{p}}}} e^{-\frac{\tau}{2(p+1)}} \left[\frac{p+1}{p} \log n\right]^{\frac{p}{p+1}}$$

$$\geq C_k \beta \inf_{\tau \in (0,\alpha)} \sqrt{\frac{\Gamma\left(\frac{1}{p}\right)}{p(\alpha-\tau)^{\frac{1}{p}}}} \inf_{\tau \in (0,\alpha)} e^{-\frac{\tau}{2(p+1)}} \left[\frac{p+1}{p} \log n\right]^{\frac{p}{p+1}}$$

$$= C_k \beta \sqrt{\frac{\Gamma\left(\frac{1}{p}\right)}{p\alpha^{\frac{1}{p}}}} e^{-\frac{\alpha}{2(p+1)}} \left[\frac{p+1}{p} \log n\right]^{\frac{p}{p+1}}$$

Hence a lower bound on the rate of $\log \epsilon_n(A)$ is

$$\Omega\left([\log n]^{\frac{p}{p+1}}\right)$$

8. Proof Step #6: Upper Bound on $\epsilon_n(A)$ =>

$$\epsilon_n(A) \leq 6C_k\,\beta \inf_{\tau\,\in\,(0,\alpha)} \sqrt{1+\frac{\Gamma\left(\frac{1}{p}\right)}{p(\alpha-\tau)^{\frac{1}{p}}}}\; e^{-\frac{\tau}{2(p+1)}}\left[\frac{p+1}{p}\log n\right]^{\frac{p}{p+1}}+\frac{\tau}{2j(p+1)}$$

This may be evaluated numerically. However, it can be seen that for any fixed

$$\tau \in (0,\alpha)$$

the rate of $\log \epsilon_n(A)$ can be bounded by

$$\mathcal{O}\left([\log n]^{\frac{p}{p+1}}\right)$$

which shows that the obtained rates are tight.

References

- Prosser, R. T. (1966): The ϵ-entropy and ϵ-capacity of certain Time-Varying Channels *Journal of Mathematical Analysis and Applications* **16** 553-573.
- Williamson, R. C., J. Shawe-Taylor, B. Scholkopf, and A. J. Smola (1999): Sample-based Generalization Bounds *NeuroCOLT2 Technical Report Series* **NC-TR-1999-055**.

Kernels for High Dimensional Data

Introduction

1. Translationally and Rotationally Invariant Kernels: Things get more complicated in higher dimension. For simplicity, we restrict ourselves to translation-invariant kernels.
2. Constructing High-Dimension Kernels: There are two simple ways to construct kernels in

$$\mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$$

with

$$d > 1$$

First, one could construct kernels by

$$k(\vec{x} - \vec{y}) = k(x_1 - y_1) \cdots k(x_d - y_d)$$

This choice in general will lead to preferred directions in the input space, as these kernels are not rotationally invariant in general. The second approach consist in setting

$$k(\vec{x} - \vec{y}) := k\left(\|\vec{x} - \vec{y}\|_{l_2}\right)$$

This approach leads to translationally invariant kernels that are also rotationally invariant.

3. Computing the Regularization Operators: In what follows, we exploit the 2nd approach to compute the regularization operators, and their corresponding Green's functions. It is quite straightforward to generalize our exposition to the rotationally asymmetric case.

Kernel Fourier Transforms

1. The Regularization Operator: We define the regularization operators P by

$$\langle Pf, Pg \rangle = \int_{supp\ P(\vec{w})}^{supp\ P(\vec{w})} \frac{\overline{F_f(\vec{w})} F_g(\vec{w})}{P(\vec{w})} d\vec{w}$$

for some non-negative function $P(\vec{w})$ converging to 0 for

$$\|\vec{w}\| \to \infty$$

It can be shown that for the kernel to be Green's function of P^*P, i.e.

$$\langle Pk(\vec{x}), Pk(\vec{x}, \vec{x}_0) \rangle = k(\vec{x}_0)$$

we need

$$F_k(\vec{w}) = P(\vec{w})$$

(Smola, Scholkopf, and Muller (1998)).

2. Radially Symmetric Kernels: For radially symmetric kernels, i.e.

$$f(\vec{x}) = f\left(\|\vec{x}\|_{l_2}\right)$$

we can carry out the integration on a sphere to obtain a Fourier transform which is also radially symmetric, i.e.

$$F_f(\|\omega\|) = w^{-\nu} H_\nu[r^\nu f(r)](\|\omega\|)$$

where

$$\nu := \frac{1}{2}d - 1$$

and H_ν is the Hankel transform over the positive real line (Sneddon (1972)).

3. The Hankel Transform: The Hankel transform H_ν is defined as

$$H_{\nu,f}(\omega) := \int_0^\infty rf(r)J_\nu(\omega r)dr$$

Here J_ν is the Bessel function of the first kind defined by

$$J_\nu(r) := \left(\frac{r}{2}\right)^\nu \sum_{j=0}^{\infty} \left(\frac{r}{2}\right)^{2j} \frac{(-1)^j}{j!\,\Gamma(j+\nu+1)}$$

Note that

$$H_\nu = {H_\nu}^{-1}$$

i.e.

$$f = H_\nu\left(H_{\nu,f}\right)$$

(in l_2) due to the Hankel inversion theorem (Sneddon (1972)).

Degenerate Kernel Bounds

1. Fourier Transforms For arbitrary Kernels: Computing the Fourier transform for an arbitrary kernel k will give us its continuous spectrum. As discussed earlier, we are interested in the discrete spectra of integral kernels defined on $\mathcal{X}$. This means that the eigenvalues are defined on the grid $\omega_0\mathbb{Z}^d$ with

$$\omega_0 = \frac{2\pi}{v}$$

2. Degeneracy of Rotationally Invariant Kernels: If $k(\vec{x})$ is rotationally invariant, so is $K(\vec{\omega})$, and therefore there are repeated eigenvalues at

$$\lambda_{\vec{j}} = (2\pi)^{\frac{d}{2}} k(\vec{j}\omega_0)$$

Consequently, we have degeneracies on the point spectrum of the integral operator given by k (or k_v respectively) as all $\vec{j}\omega_0$ with equal length will have the same eigenvalues.

3. Modified Diagonal Operator Bounds: In order to deal with the degeneracies in the eigenvalues efficiently, we modify slightly the Diagonal Operator Bounds. The degenerate Diagonal Operator Bounds allows proper account to be taken of the multiplicity of the eigenvalues, and thus allow for a more refined calculation of the desired entropy numbers.
4. Degenerate Diagonal Operator Bounds: Let

$$(s_t)_t \in \mathbb{N}_0{}^{\mathbb{N}}$$

be an increasing sequence with

$$s_0 = 1$$

and

$$\left(\sigma_j\right)_j \in \mathbb{R}^{\mathbb{N}}$$

be a non-increasing sequence of non-negative numbers with

$$\sigma_{s_j} < \sigma_{s_{\bar{j}}}$$

for

$$j < \bar{j}$$

and

$$\sigma_j < \sigma_{s_t}$$

for

$$s_{t-1} < j \leq s_t$$

and let

$$\mathcal{D}\vec{x} = \left(\sigma_1 x_1, \sigma_2 x_2, \dots, \sigma_j x_j, \dots\right)$$

for

$$\vec{x} = \left(x_1, x_2, \dots, x_j, \dots\right) \in l_p$$

be the diagonal operator from l_p to itself generated by the sequence $\left(\sigma_j\right)_j$, where

$$1 \leq p \leq \infty$$

Then for all

$$n \in \mathbb{N}$$

$$\sup_{t \in \mathbb{N}} \left[\frac{\sigma_1 \sigma_2 \dots \sigma_t}{n}\right]^{\frac{1}{s_t}} \leq \epsilon_n(\mathcal{D}) \leq 6 \sup_{t \in \mathbb{N}} \left[\frac{\sigma_1 \sigma_2 \dots \sigma_t}{n}\right]^{\frac{1}{s_t}}$$

5. Proof:

   a. Proof Approach Outline => The first part (i.e., the lower bound) follows directly from the Diagonal Operator Bounds lower bound, as it is a weaker statement than the original one. The upper bound is established by mimicking the proof in Carl and Stephani (1990). We define

$$\delta(n) = 8 \sup_{t \in \mathbb{N}} \left[\frac{\sigma_1 \sigma_2 \dots \sigma_t}{n}\right]^{\frac{1}{s_t}}$$

   and show that for all n there is an index s_t such that

$$\sigma_{s_t+1} \leq \frac{\delta(n)}{4}$$

   b. Induction Initial Phase => We choose an index n such that

$$n \leq 2^{s_j+1}$$

   and thus

$$1 \leq \frac{2}{n^{\frac{1}{s_j+1}}}$$

   Moreover, we have

$$\sigma_{s_t+1} \leq \left[\sigma_1 \sigma_2 \dots \sigma_{s_t+1}\right]^{\frac{1}{s_t+1}}$$

because of the monotonicity of

$$(\sigma_j)_j$$

and combining the both above, we get

$$\sigma_{s_t+1} \leq 2\left[\frac{\sigma_1\sigma_2 \ldots \sigma_{s_t+1}}{n}\right]^{\frac{1}{s_t+1}}$$

Using the definition of $\delta(n)$, we thus conclude that

$$\sigma_{s_t+1} \leq \frac{\delta(n)}{4}$$

If this happens to be the case for σ_1, we have

$$\epsilon_n(\mathcal{D}) \leq \sigma_1$$

which proves the theorem.

c. Induction Extension to the Sectional Operator $\mathcal{D}_{s_j}$ => If the above is not the case, there exists an index s_j such that

$$\sigma_{s_t+1} \leq \frac{\delta(n)}{4} \leq \sigma_{s_t}$$

Hence the corresponding sectional operator

$$\mathcal{D}_{s_j}: l_p \to l_p$$

with

$$\mathcal{D}_{s_j}\left(x_1, x_2, \dots, x_{s_t}, x_{s_t+1}, \cdots\right) = \left(\sigma_1 x_1, \sigma_2 x_2, \dots, \sigma_{s_t} x_{s_t}, \sigma_{s_t+1} x_{s_t+1}, 0, 0, \dots\right)$$

is of rank s_j, and the image $\mathcal{D}_{s_j}(U_P)$ of the closed unit ball U_P of l_P is isometric to the subset

$$\mathcal{D}^{(s_j)}\left(U_P{}^{(s_j)}\right)$$

of $l_P{}^{s_j}$. In any case, $\mathcal{D}_{s_j}(U_P)$ is a pre-compact set of l_p.

d. Entropy Number of the Sectional Operator $\mathcal{D}_{s_j}$ => So let $y_1, y_2, \dots, y_N$ be a maximal set of elements in $\mathcal{D}_{s_j}(U_P)$, with

$$\left\| y_j - y_{\bar{j}} \right\| > \frac{\delta(n)}{2}$$

for

$$j \neq \bar{j}$$

The maximality of this system guarantees that

$$\mathcal{D}_{s_j}(U_P) \subseteq \bigcup_{i=1}^{N} \left\{ y_i + \frac{\delta(n)}{2} U_P \right\}$$

and thus

$$\epsilon_N\left(\mathcal{D}_{s_j}\right) \leq \frac{\delta(n)}{2}$$

e. Estimation of $\epsilon_N(\mathcal{D})$ => In order to get an estimate, for $\epsilon_N(\mathcal{D})$, we split the operator $\mathcal{D}$ into two parts

$$\mathcal{D} = \left(\mathcal{D} - \mathcal{D}_{s_j}\right) + \mathcal{D}_{s_j}$$

which allows us to bound

$$\epsilon_N(\mathcal{D}) \leq \left\|\mathcal{D} - \mathcal{D}_{s_j}\right\| + \epsilon_N\left(\mathcal{D}_{s_j}\right)$$

Using

$$\left\|\mathcal{D} - \mathcal{D}_{s_j}\right\| = \sigma_{s_t+1} \leq \frac{\delta(n)}{4}$$

and the bound

$$\epsilon_N\left(\mathcal{D}_{s_j}\right) \leq \frac{\delta(n)}{2}$$

above we arrive at

$$\epsilon_N(\mathcal{D}) \leq \frac{3}{4}\delta(n)$$

f. Establishing $N \leq n$ => The final step is to show that

$$N \leq n$$

as then by substituting into the definition of $\delta(n)$ yields the result. This is achieved by a comparison of volumes. Consider the sets

$$\left\{y_j + \frac{\delta(n)}{4} {U_P}^{s_j}\right\}$$

as subsets of the space ${l_P}^{s_j}$, which is possible since

$$y_j \in \mathcal{D}_{s_j}(U_P)$$

and

$$\mathcal{D}_{s_j}(U_P) \in \mathcal{D}^{(s_j)}\left({U_P}^{(s_j)}\right)$$

These sets are obviously pairwise disjoint.

g. d-dimensional Euclidean Volumes Comparison => On the other hand, we have

$$\bigcup_{j=1}^{N}\left\{y_j + \frac{\delta(n)}{4} {U_P}^{s_j}\right\} \subseteq \mathcal{D}^{(s_j)}\left({U_P}^{(s_j)}\right) + \frac{\delta(n)}{4} {U_P}^{s_j} \subseteq 2\mathcal{D}^{(s_j)}\left({U_P}^{(s_j)}\right)$$

as

$$\frac{\delta(n)}{4} \leq \sigma_1$$

Now a comparison of the d-dimensional Euclidean volumes Vol_d provides

$$N\left(\frac{\delta(n)}{4}\right)^{s_j} Vol_{s_j}({U_P}^{s_j}) \leq 2^{s_j}\sigma_1\sigma_2 \dots \sigma_{s_j} Vol_{s_j}({U_P}^{s_j})$$

and therefore

$$N \leq \left[\frac{8}{\delta(n)}\right]^{s_j} \sigma_1\sigma_2 \dots \sigma_{s_j}$$

Using the definition of $\delta(n)$, this yields

$$N \leq n$$

Covering Numbers for Degenerate Systems

1. Optimal Entropy Numbers for Degenerate Systems: Let

$$k: \mathcal{X} \times \mathcal{X} \to \mathbb{R}$$

be a Mercer kernel, and let A be defined by the Scaling Operator Lemma with the additional restriction that the coefficients a_j have to match the degeneracy of λ_j, i.e.

$$a_{s_j} \geq a_{s_{\bar{j}}}$$

for

$$j < \bar{j}$$

and

$$a_j = a_{s_t}$$

for

$$s_{t-1} < j < s_t$$

Then one can choose A such that

$$\epsilon_n(A: l_2 \to l_2) \leq \inf_{(a_j)_j : \left(\frac{\sqrt{\lambda_j}}{a_j}\right)_j \in l_2} \sup_{j \in \mathbb{N}} 6C_k \left\| \left(\frac{\sqrt{\lambda_j}}{a_j}\right)_j \right\|_{l_2} \left(\frac{a_1 a_2 \dots a_{s_t}}{n}\right)^{\frac{1}{s_t}}$$

2. Computational Impact of Degenerate A:

$$\epsilon_n(A: l_2 \to l_2)$$

for the degenerate case is slightly more tight than that for the non-degenerate case, as the supremum effectively has to be carried out over only a subset of $\mathbb{N}$.

3. Estimation of the Multiplicity of Degeneracy: Here we compute the degree of multiplicity that occurs for different indexes $\vec{j}$. For this purpose, consider shells of radius r in $\mathbb{R}^d$ centered at the origin, i.e., rS^{d-1}, which contains a non-zero number of elements in $\mathbb{Z}^d$. Denote the corresponding radii by r_j and let $n(r_j, d)$ be the number of elements on these shells. Observe that

$$n(r, d) \neq 0$$

only when

$$r^2 \in \mathbb{N}$$

Thus

$$n(r, d) := |\mathbb{Z}^d \cap rS^{d-1}| \Longrightarrow N(r, d) := \sum_{\{0 \leq \rho \leq r : \rho^2 \in \mathbb{N}\}} n(\rho, d)$$

The determination of $n(r, d)$ is a classical problem that is completely solved by θ-series (Grosswald (1985)).

4. Occupation Numbers of Shells - Theorem: Let the formal power series $\theta(x)$ be defined by

$$\theta(x) := \sum_{j=-\infty}^{j=+\infty} x^{j^2} = 1 + 2\sum_{j=1}^{\infty} x^{j^2}$$

Then

$$[\theta(x)]^d = \sum_{j=1}^{\infty} n\left(\sqrt{j}, d\right) x^j$$

5. Use of the θ-series: While there do exist closed form asymptotic formulas for $n(r, d)$, they are inordinately complicated, and are of little use for our purposes (Grosswald (1985)). The θ-series theorem does allow one to calculate $n(r, d)$ exactly; this helps us construct an index of eigenvalues that satisfies the required ordering (at least for non-increasing functions $K(\omega)$) for us to get the main result.
6. Optimal Entropy Numbers Using Degeneracy Multiplicity: Let

$$k: \mathcal{X} \times \mathcal{X} \to \mathbb{R}$$

be a Mercer kernel with eigenvalues given by a radially symmetric non-increasing function on a lattice, i.e.

$$\lambda_{\vec{j}} = \lambda(\|\vec{j}\|)$$

with

$$\vec{j} \in \mathbb{Z}^d$$

and let A be defined by the Scaling Operator Lemma with the additional restriction that the coefficients $a_{\vec{j}}$ have to match the degeneracy of $\lambda_{\vec{j}}$, i.e.

$$a_{\vec{j}} = a(\|\vec{j}\|)$$

Then

$$\epsilon_n(A: l_2 \to l_2) \leq \inf_{(a_j)_j : \left(\frac{\sqrt{\lambda_j}}{a_j}\right)_j \in l_2^d} \sup_{t \in \mathbb{N}} 6C_k \left\| \left(\frac{\sqrt{\lambda_j}}{a_j}\right)_j \right\|_{l_2^d} \left(\frac{\prod_{q=1}^{t} a(r_q)^{n(r_q,d)}}{n} \right)^{\frac{1}{N(r_t,d)}}$$

7. Impact of Using the Degeneracy Multiplicity in Entropy Numbers: Note that while computing the optimal entropy using the degeneracy multiplicity as shown above may appear straightforward, it cannot be obtained from the Proposition for Computing Optimal Entropy Numbers for non-degenerate systems directly, as there the supremum would have to be evaluated over $\mathbb{N}$ instead of

$$[N(r_t, d)]_t$$

Bounds for Kernels in $\mathbb{R}^d$

1. Overview: In this section we examine some examples of the eigenvalue sequences for kernels typically used in SV machines. These can be used to evaluate the optimal entropy numbers for those degenerate systems using their degeneracy multiplicity. We begin by computing the Fourier/Hankel transform for these kernels - recalling that

$$\nu = \left(\frac{d}{2}\right) - 1$$

   in all the cases here.
2. Gaussian Radial Basis Functions: For Gaussian radial basis functions in d-dimensions, we have

$$k(r) = \sigma^{-d} e^{-\frac{r^2}{2\sigma^2}}$$

and correspondingly

$$F_k(\omega) = \omega^{-\nu}\sigma^{-d} H_\nu\left[r^\nu e^{-\frac{r^2}{2\sigma^2}}\right](\omega) = \omega^{-\nu}\sigma^{2(\nu+1)-d}\omega^\nu e^{-\frac{\omega^2\sigma^2}{2}} = e^{-\frac{\omega^2\sigma^2}{2}}$$

3. Exponential Radial Basis Functions: In this case

$$k(r) = e^{-ar}$$

so

$$F_k(\omega) = \omega^{-\nu} H_\nu[r^\nu e^{-ar}](\omega) = \omega^{-\nu} 2^{\nu+1}\omega^\nu \frac{a}{\sqrt{\pi}}\Gamma\left(\nu+\frac{3}{2}\right)\frac{1}{(a^2+\omega^2)^{\nu+\frac{3}{2}}}$$

$$= a\sqrt{\frac{2^d}{\pi}}\Gamma\left(1+\frac{d}{2}\right)\frac{1}{(a^2+\omega^2)^{\frac{d+1}{2}}}$$

Thus, in the case of

$$d = 1$$

we recover the case of damped harmonic oscillator (below) in the frequency domain.

   a. Eigenvalue Decay => In general, we get a decay in terms of eigenvalues like ω^{-d-1}. Moreover, one can conclude from this that the Fourier transform of k, viewed itself as a kernel, i.e.

$$k(r) = (1+r^2)^{-\frac{d+1}{2}}$$

yields the initial kernel as its corresponding power spectrum in the Fourier domain.

4. Damped Harmonic Oscillator: Another way to generalize the Harmonic oscillator, this time in a way that k does not depend on the dimensionality d, is to set

$$k(r) = \frac{1}{a^2 + r^2}$$

   a. Fourier Transform and Asymptotics => Following Watson (1958), we get

$$F_k(\omega) = \omega^{-\nu} H_\nu \left[\frac{r^\nu}{a^2 + r^2}\right](\omega) = \left(\frac{a}{\omega}\right)^\nu K_\nu(\omega a)$$

   where K_ν is the Bessel function of the 2nd kind, defined by

$$K_\nu(x) = \int_0^\infty e^{-x \cosh t} \cosh(\nu t)\, dt$$

   (Sneddon (1972)). It is possible to upper bound by utilizing the asymptotic representation

$$K_\nu(x) \sim \sqrt{\frac{\pi}{2x}} e^{-x}$$

   (e.g., Gradshteyn and Ryzhik (1981), equation (8.451.6)), and we get exponential decay of the eigenvalues.

5. Bounds Computation Steps - Recap: Using a) the theorem for the Occupation Numbers of Shells, b) the Optimal Entropy Numbers using Degeneracy Multiplicity, and c) the d-dimensional Eigenvalue Calculation Lemma, one may compute the entropy numbers numerically for a particular kernel, and for a particular set of parameters.
6. Conservative Nature of the Obtained Bounds: The obtained bounds may seem too loose and conservative from a practical point of view. However, as the ultimate goal is to use the obtained bounds for model selection, it is desirable to obtain as tight bounds as possible, esp. in the constants.

7. Computation of more Precise Bounds: If much more precise bounds may be obtained by some not too expensive numerical calculations, it is definitely worthwhile to use that instead of the theoretically nice but insufficiently tight upper bounds. The computational effort required to calculate these quantities is typically negligible in comparison to training the actual machine.

Impact of the Fourier Transform Decay on the Entropy Numbers

1. Introduction: Notwithstanding the caveats of the section above, in order to get a feeling for the decay of the Fourier transform of the kernel on the entropy numbers of the A operator, we conclude with the following general result, and its eventual proof.
2. Eigenvalue Polynomial Exponential Decay in $\mathbb{R}^d$: For kernels

$$k: \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$$

with

$$\lambda(\omega) = \mathcal{O}\left(e^{-\alpha\|\omega\|^p}\right)$$

where

$$\alpha, p > 0$$

the entropy number of the corresponding scaling operators satisfies

$$|\log \epsilon_n(A; l_2 \to l_2)| = \mathcal{O}\left([\log n]^{\frac{p}{p+d}}\right)$$

3. Proof:

a. Assumption of Continuous Eigenvalues => We will completely ignore the fact that we are dealing with a countable set of eigenvalues in a lattice, and replace all summations by integrals. Of course, this is not accurate, but will still give the correct rates for the entropy numbers.
b. Estimate Corresponding Infinitesimal Volumes and Number Density => We denote

$$\frac{1}{v} := \left(\frac{2\pi}{v}\right)^{\frac{d}{2}}$$

as the size of the unit cell, i.e.

$$v := \left(\frac{v}{2\pi}\right)^{\frac{d}{2}}$$

as the density of the lattice points in the frequency space as seen earlier. Then we get for infinitesimal volumes ΔV and the number of eigen-points ΔN in frequency space

$$\Delta V = S_{d-1} r^{d-1} \Delta r$$

and therefore

$$\Delta N = v S_{d-1} r^{d-1} \Delta r$$

(here S_{d-1} denotes the volume of the $d-1$ dimensional unit sphere) leading to

$$N(r, d) = \frac{1}{d} v S_{d-1} r^{d}$$

c. Eigenvalue Decay of the Scaling Operator => We introduce a scaling operator whose eigenvalue decays as

$$a(\omega) = e^{-\frac{\tau}{2}\|\omega\|^p}$$

for

$$\tau \in [0, \alpha)$$

It is straightforward to check that all these values lead to both useful and admissible scaling operators. We now estimate the individual terms in the expression for the Optimal Entropy Numbers using the Degeneracy Multiplicity:

$$\epsilon_n(A: l_2 \to l_2)$$

$$\leq \inf_{(a_j)_j : \left(\frac{\sqrt{\lambda_j}}{a_j}\right)_j \in l_2^d} \sup_{t \in \mathbb{N}} 6C_k \left\| \left(\frac{\sqrt{\lambda_j}}{a_j}\right)_j \right\|_{l_2^d} \left(\frac{\prod_{q=1}^{t} a(r_q)^{n(r_q,d)}}{n} \right)^{\frac{1}{N(r_t,d)}}$$

d. Estimation of $\left\| \left(\frac{\sqrt{\lambda_j}}{a_j}\right)_j \right\|_{l_2}^2$ =>

$$\left\| \left(\frac{\sqrt{\lambda_j}}{a_j}\right)_j \right\|_{l_2}^2 \approx \int dN(\omega) \frac{\lambda(\omega)}{a^2(\omega)} = vS_{d-1} \int \omega^{d-1} \beta^2 e^{-(\alpha-\tau)\|\omega\|^p} d\omega$$

$$= \frac{vS_{d-1}\beta^2}{p(\alpha-\tau)^{\frac{d}{p}}} \left(\frac{d}{p}\right)$$

e. Estimating $\left(\frac{1}{n}\right)^{\frac{1}{N(r,d)}}$ => Next we have

$$\log\left[n^{-\frac{1}{N(r,d)}}\right] = -\frac{d}{vS_{d-1}{r_0}^d} \log n$$

f. Estimating $\left(\prod_{q=1}^{t} a(r_q)^{n(r_q,d)}\right)^{\frac{1}{N(r_t,d)}}$ =>

$$\log\left[\left(a_1 a_2 \dots a_{N(r,d)}\right)^{\frac{1}{N(r,d)}}\right] = -\frac{d}{vS_{d-1}{r_0}^d}\sum_{j=1}^{N(r,d)} \log a_j \approx \frac{d}{r^d}\int_0^r \omega^{d-1}\log a(\omega)\, d\omega$$

$$= -\frac{d}{r^d}\int_0^r \omega^{d-1}\frac{\tau}{2}\omega^p d\omega = -\frac{\tau}{2}\frac{d}{d+p}r^p$$

g. Computing $\epsilon_n(A)$ => Combining all of the above leads to

$$\epsilon_n \leq 6C_k\beta\sqrt{\frac{vS_{d-1}\Gamma\left(\frac{d}{p}\right)}{p}}\ \inf_{\tau\in[0,\alpha)}\frac{1}{(\alpha-\tau)^{\frac{d}{2p}}}\ \sup_{r\in\mathbb{R}^+} e^{-\frac{d}{vS_{d-1}r^d}\log n-\frac{\tau}{2}\frac{d}{d+p}r^p}$$

Computing

$$\sup_{r\in\mathbb{R}^+}$$

yields

$$r = \left[\frac{2}{\tau vS_{d-1}}\frac{(p+d)d}{p}\log n\right]^{\frac{1}{p+d}}$$

and therefore

$$\epsilon_n \leq 6C_k\beta\sqrt{\frac{vS_{d-1}\Gamma\left(\frac{d}{p}\right)}{p}}\ \inf_{\tau\in[0,\alpha)}\frac{1}{(\alpha-\tau)^{\frac{d}{2p}}}\ \sup_{r\in\mathbb{R}^+} e^{-\left(\frac{\tau}{2}\right)^{\frac{d}{d+p}}\left(\frac{(d+p)d}{p}\frac{\log n}{vS_{d-1}}\right)^{\frac{p}{d+p}}}$$

h. Optimizing over τ => Already from the above one can observe the rate bounds on ϵ_n. What remains to be done is the computation of the infimum over τ. This can be done by differentiating the final expression with respect to τ. Defining

$$T_n := \left[\frac{\log n}{\nu S_{d-1}} \frac{(p+d)d}{p}\right]^{\frac{p}{p+d}}$$

which leads to the optimality condition on τ

$$\frac{\alpha - \tau}{\tau^{\frac{p}{d+p}}} = \frac{d+p}{2pT_n} 2^{\frac{d}{d+p}}$$

with

$$\tau \in [0, \alpha)$$

can be solved numerically.

References

- Gradshteyn, I. H., and I. M. Ryzhik (1981): *Tables of Integrals, Series, and Products* **Academic Press** New York.
- Grosswald, F. (1985): *Representation of Integers as Sums of Squares* **Springer-Verlag** New York.
- Smola, A. J., B. Scholkopf, and A. J. Muller (1998): The Connection between Regularization Operators and Support Vector Kernels *Neural Networks* **11** 637-649.
- Sneddon, I. H. (1972): *The Use of Integral Transforms* **McGraw Hill** New York.
- Watson, G. N. (1958): *A Treatise on the Theory of Bessel Functions, 2nd edition* **Cambridge University Press** Cambridge UK.

Regularization Networks Entropy Numbers Determination - Practice

Introduction

1. Motivation: Williamson, Smola, and Scholkopf (2001) have shown how to connect properties known about mappings into feature spaces with bounds on covering numbers. Exploiting the geometric structure of the feature space map, they relate properties of the kernel inducing the feature space to the covering numbers of the class of functions implemented by the SV machines based on such kernels.
2. Usage in Model Selection: The application of the results of Williamson, Smola, and Scholkopf (2001) (e.g., for model selection using structural risk minimization) is somewhat limited, but certainly doable. For instance, Guo, Bartlett, Shawe-Taylor, and Williamson (1999) apply these results to study the performance of SV machines for pattern classification.

Custom Application of the Kernel Machines Entropy Numbers

1. Choosing the Kernel and the Bandwidth: While the results do not aid in the selection of the kernel machines per se, they offer suggestions on the choice of bandwidth. Since the Fourier transform of the kernel goes as

$$K_\sigma(\omega) = K\left(\frac{\omega}{\sigma}\right)$$

scaling the kernel by σ implies more densely spaced eigenvalues in the spectrum of the integral operator $\mathcal{T}_{k_\sigma}$.

2. Choosing the Kernel Period v: Since

$$K(\omega) = \mathcal{O}\left(\frac{1}{\omega}\right)$$

its eigenvalue increases with v, thereby suggesting a small choice of v. However, small v results in large empirical error, as the kernel loses its localization properties. Williamson, Smola, and Scholkopf (2001) suggest an approach to pick an optimally satisfactory v based on the above trade-off.

3. Bound on $\epsilon_n(A)$: These bounds are computed using either the Optimal Scaling Operator Lemma for the unidimensional case, or the adjustment to the Optimal Scaling Operator Lemma for Degeneracy Multiplicity (along with the theorem for the Occupation Numbers of Shells) for the multidimensional case.
4. Bound on $\epsilon_n(\mathcal{T})$: These are estimated using the combination of the Maurey's theorem, the Carl and Stephani's theorem, and the customization applied for kernel machines in Williamson, Smola, and Scholkopf (2001).
5. Taking Account of $+b$: The key observation is that given a class $\mathcal{F}$ with known $\mathcal{N}_m(\epsilon, \mathcal{F})$, one can bound $\mathcal{N}_m(\epsilon, \mathcal{F}^+)$ where

$$\mathcal{F}^+ := \{f + b : f \in \mathcal{F}, b \in \mathbb{R}\}$$

as follows. Suppose that V_ϵ is an ϵ-cover for $\mathcal{F}$, and that the elements of $\mathcal{F}^+$ are uniformly bounded by B (this implies a limit on $|b|$ as well as a uniform bound on the elements of $\mathcal{F}$). Then

$$V_\epsilon{}^+ := \bigcup_{j=-\frac{B}{\epsilon}}^{j=+\frac{B}{\epsilon}} V_\epsilon + j\epsilon$$

is an ϵ-cover for $\mathcal{F}^+$, and thus

$$\mathcal{N}_m(\epsilon, \mathcal{F}^+) \leq \frac{2B}{\epsilon}\mathcal{N}_m(\epsilon, \mathcal{F})$$

Obviously, this will only be "noticeable" for those classes $\mathcal{F}$ whose covering numbers grow very slowly (i.e., polynomial in $\frac{1}{\epsilon}$).

6. Taking Account of the Loss Functions: While much of the treatment of Williamson, Smola, and Scholkopf (2001) focusses on the regularization component of the regularized error, the corresponding treatment of the empirical error considers loss functions in terms of their Lipschitz-continuous and Lipschitz-continuous-light equivalents. Applying one of the standard empirical bounds in terms of their covering numbers, one may extract the corresponding uniform convergence results.

Extensions to the Operator-Theoretic Viewpoint for Covering Numbers

1. Overall Bounds of Learning Machines: The overall bounds for SV classes, via a more involved argument, can be somewhat simplified (Guo, Bartlett, Shawe-Taylor, and Williamson (1999)). The general approach can be applied to various other learning machines such as convex combinations of basis functions and multi-layer networks (Smola, Elisseff, Scholkopf, and Williamson (2000)).
2. Tighter Statistical Bounds and Unsupervised Learning Extensions: When combined with an appropriate statistical argument (Shawe-Taylor and Williamson (1999)), the approach yields bounds on the generalization that depend strongly on the particular sample observed (Williamson, Shawe-Taylor, Scholkopf, and Smola (1999)). These methods can also be applied to some problems of unsupervised learning (Smola, Williamson, Mika, and Scholkopf (1999)).
3. Extensions to non-l_2 Norms: The results of Williamson, Smola, and Scholkopf (2001) hinge on the measurement of the size of the weight vector $\vec{\omega}$ using the l_2 norm. Williamson, Smola, and Scholkopf (2000) show the impact of different norms for measuring the size of $\vec{\omega}$, as well as a number of related results.

References

- Guo, Y., P. L. Bartlett, J. Shawe-Taylor, and R. C. Williamson (1999): Covering Numbers for Support Vector Machines, in: *Proceedings of the 12th Annual Conference on Computational Learning Theory* 267-277 **ACM** New York.
- Shawe-Taylor, J., and R. C. Williamson (1999): Generalized Performance of Classifiers in Terms of Observed Covering Numbers, in: *Proceedings of the 4th European Conference on Computational Learning Theory (EUROCOLT '99)* 274-284.
- Smola, A. J., R. C. Williamson, S. Mika, and B. Scholkopf (1999): Regularized Principal Manifolds, in: *Proceedings of the 4th European Workshop in Computational Learning Theory (EUROCOLT '99)* 214-229.
- Smola, A. J., A. Elisseff, B. Scholkopf, and R. C. Williamson (2000): Entropy Numbers for Convex Combinations and mlps, in: *Advances in Large Margin Classifiers (A. Smola, P. Bartlett, B. Scholkopf, and D. Schuurmans, editors)* **MIT Press** Cambridge MA.
- Williamson, R. C., J. Shawe-Taylor, B. Scholkopf, and A. J. Smola (1999): Sample-based Generalization Bounds *NeuroCOLT2 Technical Report Series* **NC-TR-1999-055**.
- Williamson, R. C., A. J. Smola, and B. Scholkopf (2000): Entropy Numbers of Linear Function Classes, in: *Proceedings of the 13th Annual Conference on Computational Learning Theory (N. Cesa-Bianchi and S. Goldman, editors)* 309-319.
- Williamson, R. C., A. J. Smola, and B. Scholkopf (2001): Generalization Performance of Regularization Networks and Support Vector Machines via Entropy Numbers of Compact Operators *IEEE Transactions on Information Theory* **47 (6)** 2516-2532.

Minimum Description Length Approach

Motivation

1. MDL Function Coding Philosophy: The MDL approach is based on the following notion of simplicity: A function object is simple if it only needs a small number of bits to code it.
2. MDL Function Coding Approach: A naïve function coding approach maintains a simple input-to-output table scheme (either direct or through ranges). More sophisticated coding schemes us the theory of data coding and compression to uncover data regularity.

Coding Approaches

1. MDL Coding Practice: Given a function space $\Im$ and a set of training points, we try to pick out the function

$$f \in \Im$$

that minimizes the following expression:

$$\mathcal{L}(f) + \mathcal{L}(Training\ Data \mid f)$$

where $\mathcal{L}(f)$ is the length of the code used to encode the function

$$f \in \Im$$

2. $\mathcal{L}(Training\ Data \mid f)$: The term $\mathcal{L}(Training\ Data \mid f)$ denotes the length of the code needed to express the given training data with the aid of the function f. The idea is simple; if f fits the training data well, this part of the code will be very small.
3. MDL vs. Classical SLT: $\mathcal{L}(Training\ Data \mid f)$ corresponds to the training error the function f makes on the data. The term $\mathcal{L}(f)$ measures the complexity of f. Thus, MDL and SLT are both very similar; the both sum up the training error and the complexity term. However, in SLT, the complexity term is only a function of $\mathfrak{I}$, whereas in MDL it is a function of f.
4. Building Universal Function Codes: One way to approach the above is to decompose $\mathfrak{I}$ into subsets

$$\mathfrak{I}_1 \subset \mathfrak{I}_2 \subset \cdots$$

Then we encode the elements of each subset with a fixed length code that assigns each member of $\mathfrak{I}_i$ the same length. If $\mathfrak{I}_i$ is finite with N elements, it is possible to encode each member

$$f \in \mathfrak{I}_i$$

using $\log N$ bits. If $\mathfrak{I}_i$ is infinite, one may use the VC dimension (or some complexity measure) to encode the function.
5. MDL Coding Complexity: The *coding complexity* of $\mathfrak{I}_i$ can be defined as the smallest code length one can achieve for encoding the functions

$$f \in \mathfrak{I}_i$$

It is uniquely defined, and is *universal* in the sense that it does not rely on a specific scheme.
6. Function Coding Technique: Given a function

$$f \in \mathfrak{I}_i$$

we encode our data in several steps; we first encode the index i of the function class, then we use a uniform code to encode which element of $\mathfrak{I}_i$ the function f is, and finally code the data with help of f. The code length then becomes

$$\mathcal{L}(i) + \mathcal{L}(f \mid f \in \mathfrak{I}_i) + \mathcal{L}(Training\ Data \mid f)$$

The goal of this segmented approach is, of course, to choose the hypothesis f that minimizes this code.

a. Function Class Code $\mathcal{L}(i)$ => Usually for $\mathcal{L}(i)$ one chooses a uniform code of integers, as long as we are dealing with finitely many classes $\mathfrak{I}_i$. Thus, this entity is typically constant across all classes.
b. $\mathcal{L}(f \mid f \in \mathfrak{I}_i)$ => This term is identical for all

$$f \in \mathfrak{I}_i$$

It does not distinguish between different functions within $\mathfrak{I}_i$, but only from with functions that come from different $\mathfrak{I}_i$'s.

c. $\mathcal{L}(Training\ Data \mid f)$ => This term explains how well the given f can explain the training data.

MDL Analyses

1. MDL for a Fixed Function Class $\mathfrak{I}$: If we are given just one fixed function class $\mathfrak{I}$ (without splitting it down into sub-classes $\mathfrak{I}_i$), the code length essentially depends on some measure of complexity of $\mathfrak{I}$ plus an additional term that accounts for how well f fits the data. In this setting, it is easy to prove that the learning bounds of MDL compare as well with SLT.
2. Compression Coefficients SLT: As shown in Vapnik (1995), the MDL approach outlined above is closely related to the classical SLT approaches based on compression bounds.

3. MDL and the PAC-Bayesian Approach: In more advanced MDL approaches one assigns different code lengths to different elements of $\mathfrak{I}_i$ – in this case the approach is more closely related to the PAC-Bayesian approach.
4. Infinite Data Limit: Just as in classical SLT (but under certain other assumptions), MDL can be performed in a consistent way, i.e., in the infinite data limit, the approach can find the correct model for the data (Grunwald (2007), Rissanen (2007)).

References

- Grunwald, P. (2007): *The Minimum Description Length Principle* **MIT Press** Cambridge, MA.
- Rissanen, J. (2007): *information and Complexity in Statistical Modeling* **Springer** New York.
- Vapnik, V. (1995): *The Nature of Statistical Learning Theory* **Springer Verlag** New York.

Bayesian Methods

Bayesian and Frequentist Approaches

1. Bayesian Approach - Principle: As opposed to the agnostic approach taken in classical SLT, we assume that the underlying data distribution comes from some class of probability distribution $\mathcal{P}$. This probability class is described by one/more parameters, i.e., it is of the form

$$\mathcal{P} = \{\mathcal{P}_\alpha | \alpha \in \mathcal{A}\}$$

   where $\mathcal{A}$ is the set of parameters values, and α the corresponding distribution parameters (e.g., if it is normal, this may be the mean/variance).
2. Goal of the Frequentist Approach: The main goal of the frequentist approach is to infer the correct parameter set of the underlying distribution. Classification etc. is then performed on the data set using the inferred parameters.
   a. Maximum Likelihood Estimate => The frequentist approach de facto estimates the likelihood of the data given the parameter α realization, i.e., $\mathcal{P}(Data \mid \alpha)$. The MLE computes the parameter that maximizes this likelihood, i.e.

$$\hat{\alpha}(Data) = \underset{\alpha \in \mathcal{A}}{\arg max}\, \mathcal{P}(Data \mid \alpha)$$

      Obviously MLE only estimates the confidence of the parameters given the data, not the probability that parameters are correct in any absolute sense.

Bayesian Approaches

1. Motivation: Here we define a distribution $\mathcal{P}_\alpha$ which, for each parameter α, encodes how likely we find that this is a good parameter to describe our problem. The important point is that this prior distribution is defined *before* we get to see the data points from which we like to learn.
2. Literature:
    a. Cox (1961) introduces the fundamental axioms that allow expressing beliefs using probability calculus.
    b. Jaynes (2003) addresses philosophical/practical issues with these axioms.
    c. O'Hagan (1994) provides a very general treatment.
    d. Bayesian methods in Machine Learning approaches are available in Tipping (2003) and Bishop (2006).
3. Computation of the Posterior: As in the frequentist approach we do compute $\mathcal{P}(Data \mid \alpha)$, the likelihood term. In addition, combining $\mathcal{P}(\alpha)$ with the prior, we can also compute the posterior distribution $\mathcal{P}(\alpha \mid Data)$ as

$$\mathcal{P}(\alpha|Data) \propto \mathcal{P}(\alpha)\mathcal{P}(Data|\alpha)$$

4. MAP Estimator: The MAP approach involves choosing the $\widehat{\alpha_{MAP}}$ that maximizes the posterior probability

$$\widehat{\alpha_{MAP}} = \underset{\alpha \in \mathcal{A}}{\arg max}\, \mathcal{P}(\alpha \mid Data\,)$$

5. Bayesian Approaches for Finite Samples: On a finite sample, usage of the prior helps bias towards solutions more likely (thereby, no washing out of the prior, see Berger (1985)). Here it is more appropriate to indicate that the Bayesian method is more convenient means of updating our beliefs about the solutions before looking at the data.
6. MAP re-cast in SLT terms:

$$-\log \mathcal{P}(\alpha \mid Data) = \underset{\alpha \in \mathcal{A}}{\arg max}\, [-\log \mathcal{P}(\alpha) - \log \mathcal{P}(Data \mid \alpha)]$$

Like SLT, $\log \mathcal{P}(Data \mid \alpha)$ is the term that describes the model fit to the data.

7. $-\log \mathcal{P}(\alpha)$ as a measure of Model Complexity: If there are more models, then the probability gets stretched across the entire parameter space, thereby trimming/limiting the contribution due to each $\mathcal{P}(\alpha)$, thus increasing $-\log \mathcal{P}(\alpha)$. From coding theory, this indicates that, as the model complexity increases, greater is $-\log \mathcal{P}(\alpha)$. This is consistent with the concept of model complexity of SLT.

References

- Berger, J. O. (1985): *Statistical Decision Theory and Bayesian Analysis* **Springer Verlag** New York.
- Bishop, C. (2006): *Pattern Recognition and Machine Learning* **Springer**.
- Cox, R. T. (1961): *The Algebra of Probable Inference* **Johns Hopkins University Press** Baltimore.
- Jaynes, E. T. (2003): *Probability Theory – The Logic of Science* **Cambridge University Press** Cambridge.
- O'Hagan, A. (1994): Bayesian Inference, in: Volume 2B, *Kendall's Advanced Theory of Statistics* **Arnold** London.
- Tipping, M (2003): Bayesian Inference: An Introduction to Principles and Practice in Machine Learning, in: *Advanced Lectures in Machine Learning* (editors O. Bosquet, U. von Luxburg, and G. Ratsch) 41-62 **Springer**.

Knowledge Based Bounds

Places to incorporate Bounds

1. The Purpose: The purpose is to get tighter bounds into the pessimistic bounds above, as well as for accounting for the issues raised in the *NFL Theorems*, which states that learning is impossible unless we make assumptions on the underlying distribution.
2. Incorporating Prior Knowledge into Data Space: This may be done using a topology-based distance metric, for e.g., and the closeness of this metric signals the likelihood of the output. The classifier inputs will then be coded using these metric transforms (e.g., as in kNN).
3. Prior Knowledge via the Underlying Probability Distribution: This simply an alternate way of specifying

$$f \in \mathfrak{I}$$

   However, this *hits* our assumption of distribution agnosticity (we are OK with that in this section, given that we try to cater to NFT).
4. Encoding Prior Knowledge via the Loss Function l: The loss function encodes the learning goals, so can be customized in different manner using different weights (e.g., on daily points or "error types").
5. Example: In many problems "false positives" and "false negatives" have differential costs associated. For instance, in spam filtering, the cost of accidentally labeling the spam as "not spam" is not that high, whereas the cost of labeling a non-spam as a spam is high.

Prior Knowledge into the Function Space

1. Prior Knowledge into the Choice of $f \in \mathfrak{I}$: Here the assumptions of usefulness of a classifier are encoded by the appropriate choice of

$$f \in \mathfrak{I}$$

   Generally, this is not very useful, as it still leaves significant wiggle room in the choice – the exception where the encoding via f becomes important is in Bayesian approaches.
2. Shortcomings of the Capacity Measures: While both

$$f \in \mathfrak{I}$$

   and the data points go together into the construction of the $\mathfrak{I}$ capacity/complexity measure, they do not typically weight the contribution of a specific f (except in a fairly cumbersome manner). The focus is on how to incorporate the knowledge into the bounds rather than simply counting the function in $\mathfrak{I}$, either via knowledge a *priori* or *a posteriori*.
3. Classifier Function Specific Prior Knowledge: The Bayesian approach is one way to incorporate the classifier function specific prior knowledge. A prior distribution $\pi(f)$ expresses our belief on how to place emphasis on the classifier function space. However, $\pi(f)$ should be chosen well before the statistical inference process is started.
4. PAC Bayesian Bounds: With the incorporation of the Bayesian priors into the classical SLT framework, it can be shown that, with a probability of at least $1 - \delta$

$$R(f) \leq R_{emp}(f) + \sqrt{\frac{\log\frac{1}{\pi(f)} + \log\frac{1}{\delta}}{2n}}$$

   This is the simplest among the PAC bounds (Boucheron, Bousquet, and Lugosi (2005)), and does not involve the capacity term for $\mathfrak{I}$; instead, it penalizes the individual functions according to $\pi(f)$.
   a. Data Agnosticism of PAC Bounds => While the PAC Bayesian approach outlined above accounts for fixing the function agnosticism challenge, it is data sample

agnostic, and therefore produces conservative bounds without differentiating among the different types of data.

5. Luckiness Framework: Here, both the bounds as well as the capacity are allowed to depend on the actual data at hand, thus varying with different "groups" of data. This approach was first introduced in statistics under the name "conditional confidence sets" (Keifer (1977)), and then expanded under the classical SLT framework by Shawe-Taylor, Bartlett, Williamson, and Anthony (1998) and Herbrich and Williamson (2002).

References

- Boucheron, S., O. Bousquet, and G. Lugosi (2005): Theory of Classification *ESAIM: Probability and Statistics* **9** 323-375.
- Herbrich, R., and R.C. Williamson (2002): Learning and Generalization: Theoretical Bounds, in *Handbook of Brain Theory and Neural Networks* (editor: M Arbib).
- Keifer, J. (1977): Conditional Confidence Statements and Confidence Estimators *Journal of the American Statistical Association* **72 (360)** 789-808.
- Shawe-Taylor, J., P. Bartlett, R. Williamson, and M. Anthony (1998): Structural Risk Minimization over Data-dependent Hierarchies *IEEE Transactions on Information Theory* **44 (5)** 1926-1940.

Approximation Error and Bayes' Consistency

Motivation

1. Estimation vs. Approximation Error Optimization - Contradictory Goals: In general, the estimation error may be made small by reducing the complexity of the hypothesis space. However, the approximation error requires a widening of the function class!
2. Approaches for Tackling Bayes' Consistency: If

$$f_{Bayes} \in \mathfrak{I}$$

for some known, small function space $\mathfrak{I}$, the approximation error becomes zero, and the Bayes' consistency criterion then reduces to consistency with respect to $\mathfrak{I}$. If the above is not the case, then the only other known method is to ensure that the nested function spaces contain f_{Bayes} "*in the limit*".

Nested Function Spaces

1. The Concept: As the sample size n increases, so does the complexity of the function space $\mathfrak{I}_n$. The standard construction is to choose the spaces $\mathfrak{I}_n$ such that they constitute an expanding sequence of nested function spaces, i.e.

$$\mathfrak{I}_1 \subset \mathfrak{I}_2 \subset \mathfrak{I}_3 \ldots$$

Two conditions are needed to ensure this classifier is Bayes consistent.

2. Zero Estimation Error:

$$Estimation\ Error \to 0$$

as

$$n \to \infty$$

The estimation error bound decreases as the sample size increases, but increases as the complexity term increases. To ensure that the overall estimation error is still decreasing, the complexity term contribution should not dominate the sample size contribution, i.e., $\mathfrak{I}_n$ does not grow too fast as n increases.

3. Zero Approximation Error:

$$Approximation\ Error \to 0$$

as

$$n \to \infty$$

This is possible only if, for some large n, either f_{Bayes} is contained in $\mathfrak{I}_n$, or it can be approximated by it closely enough.

4. Necessary and Sufficient Condition for Bayes' Consistency: This comes from Devroye, Gyorfi, and Lugosi (1996). Let

$$\mathfrak{I}_1 \subset \mathfrak{I}_2 \subset \mathfrak{I}_3 \ldots$$

be a sequence of nested function spaces, and consider the classifier set described by

$$f_n = \underset{f \in \mathfrak{I}}{\arg min}\, R_{emp}(f)$$

Assume that for a distribution $\mathbb{P}$ the following conditions are satisfied:

a. The VC Dimension of the spaces in $\mathfrak{I}_n$ satisfies

$$\mathcal{VC}(\mathfrak{I}_n).\frac{\log n}{n} \to 0$$

as

$$n \to \infty$$

and

b.

$$R\left(f_{\mathfrak{I}_n}\right) \to R\left(f_{Bayes}\right)$$

as

$$n \to \infty$$

Then the sequence of classifiers $\mathfrak{I}_n$ is Bayes' consistent.

5. Bayes' Consistency – Sample: Say we choose the function space such that

$$\mathcal{VC}(\mathfrak{I}_n) \approx n^{\alpha}$$

for some

$$\alpha \in]0,1[$$

Then the first condition is satisfied, as

$$\mathcal{VC}(\mathfrak{I}_n).\frac{\log n}{n} \approx \frac{\log n}{n^{1-\alpha}} \to 0$$

as

$$n \to \infty$$

However, if

$$\alpha = 1$$

then

$$\mathcal{VC}(\mathfrak{I}_n).\frac{\log n}{n} \to \infty$$

as

$$n \to \infty$$

Regularization

1. Penalizing Regularizer: An implicit way of working with nested function spaces is the principle of regularization. Instead of minimizing the empirical risk $R_{emp}(f)$ and the expressing the generalization ability of the resulting classifier f_n using some capacity measure of the underlying function class $\mathfrak{I}$, one can directly minimize the regularized risk expressed as

$$R_{Reg}(f) = R_{emp}(f) + \lambda\Omega(f)$$

2. Regularizer Behavior: $\Omega(f)$ is called the regularizer, and it is designed to punish complexity. For instance, to punish functions with large fluctuations, one can choose a $\Omega(f)$ that is small

for functions that are smooth/vary smoothly, and large for functions that fluctuate a lot. Thus, for linear classifiers one chooses $\Omega(f)$ as the inverse of the margin of a function.

3. Regularizer Trade-off Constant: λ weights $R_{Emp}(f)$ vs. $\Omega(f)$. High λ implies greater penalty, and one ends up choosing functions with low empirical risk. Low λ implies more empirical risk.
4. Regularization Principle: This principle chooses the f_n that minimizes the regularization risk $R_{Reg}(f)$. Many popular classifiers can be cast in this regularization frame-work (Scholkopf and Smola (2002)).
5. Trade-off Factor Impact for Bayes' Consistency: Ensuring Bayes' consistency in the regularization framework requires that:
    a. The nested function space f_n from $\Im$ contain the Bayes' classifier
    b. The impact of λ should be such that, as

$$n \to \infty$$

the ERM hypothesis space contribution dominates the penalty contribution

    c. The regularizing penalty should still have an impact at

$$n < \infty$$

i.e., as

$$n \to \infty$$

λ should not go to 0 too fast (Steinwart (2005)).

6. Function Class Complexity vs Function Complexity: While ERM worries about function class complexity, regularization, de facto, is concerned with the *complexity* as measured by $\Omega(f)$ on an individual function.

Achieving Zero Approximation Error

1. Approximation Theory to achieve Zero Approximation Error: Essentially, we need to ensure that

$$f_{Bayes} \in \Im$$

or that f_{Bayes} is approximatable from $\Im$ for large enough n. Often simple results from approximation theory are sufficient (e.g., Cucker and Zhang (2007) work pout more sophisticated one). However, from an approximation viewpoint, we need to be able establish that if two functions are *close*, then their corresponding risk values are also close.

2. Approximation Error for Binary Classifiers: If f approximates g (where f and g are binary classifiers), and their L_1 risk is less than δ, so is their $0-1$ risk, i.e.

$$\mathbb{P}[f(x) \neq sgn\{g(x)\}] < \delta$$

Thus, to prove that the approximation error of a function space $\Im$ is smaller than δ, we have to show that every

$$f \in \Im_{all}$$

can be approximated up to δ in the norm by functions from $\Im$.

3. Approximation Theory Application - Real Numbers: Results of the above type are abundant in the literature. For example, if $\mathcal{X}$ is a bounded set of real numbers, it is well-known that on can approximate any measurable function on this set well by a polynomial. Hence on would choose the space $\Im_n$ as the spaces of polynomials with at most a degree of d_n where d_n grows slowly with n. This is enough to guarantee the convergence of the approximation error.

Rate of Convergence

1. Definition: *Rate of Convergence* gives information about how fast a quantity converges – in particular, how large n should be so as to ensure that the error is smaller than a certain quantity, i.e., for estimation error ≤ 0.1, we need

$$n = 100{,}000$$

   samples.
2. Factors determining Rate of Convergence: Rate of Convergence depends on many parameters of the underlying problem, but while the estimation error is independent of the underlying distribution $\mathbb{P}(x, y)$, the convergence of the approximation error is dependent on it.
3. Nested Function Space: Unless one makes additional assumptions on the true distribution, for any fixed sequence $\{\Im_n\}$ of nested function spaces, the rate of convergence of the approximation error can be arbitrarily low.
4. Weakening of the i.i.d. Assumptions: Rates of Convergence depend strongly on the underlying assumptions. For example, even with independent (but not identical) sampling, no uniform rate of convergence for the approximation error exists. A similar situation is true even for the estimation error if we weaken the sampling assumptions, where the X_i's are not i.i.d. anymore.
5. Near i.i.d. Sampling: While for nearly independent sampling such as stationary α-mixing process it may still be possible to recover the results similar to above, as soon as we leave this regime it can become impossible to achieve such results (Steinwart, Hush, and Scovel (2006)). Even in the case of stationary ergodicity, we can no longer achieve universal consistency (Nobel (1999)).
6. Fast Rates: If we assume that the sample points are i.i.d., and make a few assumptions about the underlying distribution (in particular, about the label noise), the rates of convergence can be improved dramatically. This branch of learning is called *Fast Learning* (Boucheron, Bousquet, and Lugosi (2005)).
7. Consistency vs. Rate of Convergence - Reprise: Consistency is a *worst-case* statement; it indicates that ultimately the algorithm will provide correct solution without systematic errors. Rates of insights provides insights on how well-behaved the algorithm is, how fast it

converges (sample sizes, etc.), and which approach is most fruitful and under which situation.

References

- Boucheron, S., O. Bousquet, and G. Lugosi (2005): Theory of Classification *ESAIM: Probability and Statistics* **9** 323-375.
- Cucker, F., and D. X. Zhou (2007): *Learning Theory – An Approximation Viewpoint* **Cambridge University Press**.
- Devroye, L., L. Gyorfi, and G. Lugosi (1996): *A Probabilistic Theory of Pattern Recognition* **Springer** New York.
- Nobel, A. (1999): Limits to Classification and Regression Estimation from Ergodic Processes *Annals of Statistics* **27 (1)** 262-273.
- Scholkopf, B. and A. Smola (2002): *Learning with Kernels* **MIT Press** Cambridge, MA.
- Steinwart, I. (2005): Consistency of Support Vector Machines and other Regularized Kernel Classifiers *IEEE Transactions on Information Theory* **51 (1)** 128-142.
- Steinwart, I., D. Hush, and C. Scovel (2006): Learning from Dependent Observations *Technical Report LA-UR-06-3507* **Los Alamos National Laboratory**.

No Free Lunch Theorem

Introduction

1. Best Classifier across all Distributions: Under the assumption of i.i.d. sampling from an underlying distribution, is there a classifier that *on average over all probability distributions* achieves better results than any other? NFT proves that the answer is ***NO***. For the finite case proof see Ho and Pepyne (2002), for convergence rates, see Devroye, Gyorfi, and Lugosi (1996), and for the most general proof see Wolpert and Macready (1997) and Wolpert (2001).
2. Intuition Behind NFT: For any set of $\mathbb{P}(x, y)$ for which a given classifier performs better than any other, we can always construct another $\mathbb{P}'(x, y)$ where the classifier performs worse than any other (Ho and Pepyne (2002)).
3. Implication: The most important ramification of NFT is: *Learning is possible ONLY in the presence of definite, non-uniform* $\mathbb{P}(x, y)$ (uniform $\mathbb{P}(x, y)$ reduces this to a uniform Bayes' prior scenario, which is de-facto information free). Thus, prior knowledge needs to be *injected* using one of the means above.
4. NFT vs. Consistency: NFT does not contradict the (universal/functional) consistency criterion, since, in some ways, consistency also indicates convergence of risk at

$$n \to \infty$$

   sample sizes independent of f (remember f translates into $\mathbb{P}(x, y)$). However, at finite sizes, some f_n may be more effective than the other.
5. NFT vs. Hypothesis Space Reduction: NFT simply states that, in order to be able to learn successfully with guarantees on the behavior of the classifier, we need to make assumptions on the underlying distribution. This fits with the complexity of $\mathfrak{I}$ - more complexity, less limiting assumption on $\mathbb{P}(x, y)$.

6. Combining NFT and the Hypothesis Reduction Paradigms: Use NFT principles to first restrict the space of probability distributions, and then use a small function class that is able to model the distributions in the class.

Algorithmic Consistency

1. Countably Finite Data Space: When the data space χ is finite, even where is no relationship between the inputs and the outputs, since repeated sampling of inputs "converge" to their *TRUE* values, simple learning-by-heart (i.e., a majority vote when the instance has been seen before, and an arbitrary prediction otherwise) would be consistent.
2. Uncountably Finite or Infinite Data Space: For the uncountable case, there is a subtle hidden assumption. To be able to define a probability measure $\mathbb{P}$ on χ, one needs a σ-algebra on the space, which is typically the Borel σ -algebra. So the hidden assumption is that $\mathbb{P}$ is a Borel measure.
3. Topological Consistency: The fact that $\mathbb{P}$ is a Borel measure means that the topology of $\mathbb{R}$ plays a role, and therefore the target function (the Bayes' classifier) is a Borel measurable. This guarantees that it is possible to approximate the target function from its value (or approximate value) at a finite number of points.
4. Continuous Data Space: The algorithms that will achieve consistency are thus those that will use the topology in the sense of "generalizing" the observed values to their neighborhoods (e.g., local classifiers). In a way, measurability of the target function is one of the crudest notions of smoothness of functions.

NFT Formal Statements

1. Basic Statement: Also called as theorem on *Arbitrarily Close to Random Guessing* (Devroye, Gyorfi, and Lugosi (1996)); Fix some

$$\varepsilon > 0$$

For every

$$n \in \mathbb{N}$$

and for every classifier f_n, there exists a distribution $\mathbb{P}$ with Bayes' risk 0 such that the expected risk of f_n is greater than $\frac{1}{2} - \varepsilon$.

2. Statement on Algorithmic Consistency: An algorithm is consistent if for any probability measure $\mathbb{P}$

$$\lim_{n\to\infty} R(f_n) = R^*$$

almost surely.

3. No Free Lunch At All: For any algorithm, and any sequence (a_n) that converges to 0, there exists a probability distribution $\mathbb{P}$ such that

$$R^* = 0$$

and

$$R(f_n) \leq a_n$$

4. VC Consistency of ERM: The ERM algorithm is consistent if for any probability measure $\mathbb{P}$

$$R(f_n) = R(f^*)$$

in probability, and

$$R_n(f_n) = R(f^*)$$

in probability.

5. VC non-trivial Consistency of ERM: The ERM algorithm is non-trivially consistent for the set $\mathfrak{I}$ and the probability distribution $\mathbb{P}$ if for any

$$c \in \mathbb{R}$$

$$\inf_{f \in \mathfrak{I}: Pf > c} P_n(f) \to \inf_{f \in \mathfrak{I}: Pf > c} P(f)$$

in probability.

References

- Devroye, L., L. Gyorfi, and G. Lugosi (1996): *A Probabilistic Theory of Pattern Recognition* **Springer** New York.
- Ho, Y. C., and D. L. Pepyne (2002): Simple Explanation of the No Free Lunch Theorem and its Implications *Journal of Optimization Theory and Applications* **115 (3)** 549-570.
- Wolpert, D., and W. G. Macready (1997): No Free Lunch Theorems for Optimization *IEEE Transactions on Evolutionary Computing* **1 (1)** 67-82.
- Wolpert, D. (2001): The Supervised Learning No Free Lunch Theorems *Proceedings of the 6th Online World Conference on Soft Computing and Industrial Applications*.

Generative and Discriminative Models

Generative Models

1. Definition: Here, the joint distribution between the observations and the hidden states are modeled. Another way of saying this is that the a priori distribution of the hidden states is first generated from the transition probabilities; the conditional distribution of the observations given the states (i.e., the emission probabilities) is then generated.
2. Observation Generation Process: This can also be a two-stage process that is generated using Gaussian mixtures. First one Gaussian distribution among the candidate list is picked; then the probability of a given set of observations is generated.
3. Generative Model Kick-off: The first few steps of a generative model may essentially be seeded off using a discriminant type parameter space initialization to kick start the run.
4. Parameter-Sparse and Model-Rich: Given that there could be times where the generative models work off of limited data, it needs to be more model intensive (and correspondingly parameter space sparse) – thereby lending itself well to be able to model joint state/observation distributions.
5. Measurements in Generative Models: Measurements steer state estimates closer to "true" estimates, but do not entirely determine it, partly because of measurement uncertainty.

Discriminant Models

1. Definition: Here the conditional distribution of the hidden states is generated directly given the observations, rather than modeling the joint state/observation distributions.
2. Advantages of Discriminant Models - #1: Arbitrary features (i.e., functions) of the observations can be modeled by injecting domain specific knowledge targeted at the problem

at hand (essentially via the predictor-response transform). Further, straightforward features/correlations/combinations of any observation set may be modeled directly.

3. Advantages of Discriminant Models - #2: The observation features need not be statistically independent of either the states, or among themselves.
4. Disadvantages of Discriminant Models:
    a. Given the limited nature of the model parameters, the types of priors applied on the hidden state models are limited.
    b. Further, these models cannot be used to predict the probability of an arbitrary state/observation sequence.

Examples of Discriminant Approaches

1. Maximum Entropy Markov Models (MEMM): MEMM models the continuous distribution of states from observations using logistic regressions.
2. Linear Chain Conditional Random Field (CRF): This discriminant model uses undirected graphical model of the Markov variables (also referred to as Markov Random Field) as opposed to the directional graphical model used in MEMM-type approaches. Therefore CRF/MRF do not suffer from label bias, increasing the chances for accuracy. However, they are computationally slower.
3. Factorial Hidden Markov Model: Here, a single observation can be conditioned on the corresponding hidden variables of a set of K independent Markov chains (Ghahramani and Jordan (1997)). Thus, if there are N states per each chain, the Viterbi learning algorithm attains a complexity of $\mathcal{O}(N^{2K}) \times T$, where T is the number of observations.
    - Exact solution for the factorial HMM may be achieved by using the junction-tree algorithm (with a complexity of $\mathcal{O}(N^{K+1}) \times K \times T$), but approximation techniques such as variational approaches may also be used.
4. K-Level Markov Tree: Here, the state at i depends on $i-1, \cdots, i-K$. Thus, hidden state is not strictly Markovian anymore, but a Markov tree; again, the complexity is $\mathcal{O}(N^K) \times T$ for K adjacent states and T total observations.

5. Triplet Markov Models: Here, the process model is augmented to model data specificities. The Theory of Evidence and Triplet Markov Models (Pieczynski (2002), Pieczynski (2007)) are now unified to be able to fuse data in a Markovian context (Bouderan, Monfrini, Pieczynski, and Aissani (2012), Lanchantin and Pieczynski (2005)) and to model non-stationary data (Bouderan, Monfrini, and Pieczynski (2012)).
    - It is also possible that Triplet Markov Models are the ones that deal explicitly with disequilibrium/non-stationary, and/or time evolving Markov models.

Differences Between Generative and Discriminant Models

1. Differences in the Approach Philosophy: Discriminant philosophy appears to be "collect data first, characterize relationships later". Generative is collect – characterize – collect – characterize - …
2. Accommodating State Evolution: The generative models may have to accommodate state evolution as well, in addition to modeling the steady state (although modeling of the evolutionary dynamics may be limited). Discriminant models appear to be primarily for steady-state/constitutive modeling.
3. Unconditional vs. Conditional Worlds: The Regression/Discriminant Models operate in a world where the observations are the only given, therefore they restrict their exploration to that conditional world. Generative Hidden Markov models do not. Therefore, these models cover a wider state space, i.e., they explore vaster axiomatic informational frameworks.
    - The generative models treat the conditional world merely as a constraint they have to accommodate, and the constraint may be hard/soft (depending upon the confidence of the observations). However, from a formulation point of view, it is important to reiterate that both discriminant and generative models are about stationary state characterization, or dynamic equilibrium (as reflected in the state transition probabilities etc.). Within this framework, generative models do accommodate richer modeling paradigms.
4. Unifiability: Given that generational models can build steady state constructs off of other latent drivers, notions such as unified approaches are more naturally handled than they are in discriminant models.

5. State Structure Estimation: Estimation of both the emission probabilities and the state transition probabilities may be possible in both frameworks. However, in discriminant/regression frameworks, these are characterized "as is", whereas in generative frameworks the parameter inference is part of a bigger story.
    - Discriminant models typically simply regress the latent state against the observation measures. Generative models infer the hidden state from the observation measures as well as past history. Further, in the generative models, the regression is to primarily establish the constitutive state relationship between an inelastic state predictor and an elastic state quantification metric.
6. Comparison in the Context of Supervised vs. Unsupervised: Usage of discriminant algorithms in the learning phase is highly useful, as they uncover the relationships among the predictors/responses across the feature contexts. Once these relationships are uncovered, however, the stage is set for the synthesis of a broader set of rules to construct the generative models that generate the joint distributions.

References

- Boudaren, M. Y., E. Monfrini, and W. Pieczynski (2012): Unsupervised Segmentation of Random Discrete Data Hidden with Switching Noise Distributions, *IEEE Signal Processing Letters* **19 (10)** 619-622.
- Ghahramani, Z. and M. I. Jordan (1997): Factorial Hidden Markov Models, *Machine Learning* **29 (2/3)**: 245-273.
- Lanchantin, P. and W. Pieczynski (2005): Unsupervised Restoration of Hidden Nonstationary Markov Chain using Evidential Priors, *IEE Transactions on Signal Processing* **53 (8)**: 3091-3098.
- Pieczynski, W. (2002): Triplet Markov Chains (Chaines de Markov Triplet) *Comptes Rendus de l'Academie des Sciences – Mathematique, Series I* **335 (3)**: 275-278.
- Pieczynski, W. (2007): Multi-sensor Triplet Markov Chains and Theory of Evidence, *International Journal of Approximate Reasoning* **45 (1)**: 1-16.

Supervised Learning

Introduction

1. Supervised Learning as a Calibration Exercise: One very valid view is that any form of calibration is supervised learning in that it deciphers the "optimal learning algorithm" (Mohri, Rostamizadeh, and Talwalkar (2012)). Therefore, as such considerations such as bias/variance trade-offs will apply (they do, for example, for all regressions – logistic or real-valued).
2. Human-Animal Concept Learning: Psychological learning (which may or may not be concept learning) is the best example of supervised learning (Supervised Learning (Wiki)). This may still be a giant hash-map, in which case the "indexer" algorithm is the calibration routine!
3. Challenges with Hand-Labeling: How do you do that for a humongous input data set? Something like Penn Tree-bank is painful handcrafted, but how do we do this for finance data? Are there effective, general-purpose feature extractors (thus, this in itself will become the supervised set in part!)?
4. Effectiveness: Vast sets of inputs data may effectively, through classification/calibration/other supervised learning paradigms, be "digested" onto learned fragments - specific to practice domains at hand – used for future predictions.

Supervised Learning Practice Steps

1. Step #1 => Training Example Types: Physical/Cognition level identification of the training data types could be heterogeneous (i.e., combination of elaborate data and/or single characterizer) using a weight-based state.

2. Step #2 => Gathering Training Set: The training set should be a representation of the real-world, as well contain the cross-validatable groups. Many require some automated algorithm for hand-labeling (esp. if the data is humongous). In all, this stage pairs the cognition/physical input groups with their label names.
3. Stage #3 => Feature Representation: Convert the set of input data into a machine-represented feature vector (again heterogeneous). Balance must be achieved between too grainy and too broad, i.e., balance between curse of dimensionality and inaccuracy.
4. Step #4 => Learning Algorithm Choice: Determination of the learning algorithm/physical model also requires making decisions on the algorithm design choice, parameters, complexity, flexibility, and performance.
5. Step #5 => Run setup: Group the training inputs into cross validation/GCV subsets. This helps:
    - Identify the base parameter set
    - Bayesian parameter distribution.
    - If the input is rich enough, it also helps determine the Bayesian hyper-parameters.

    Attention needs to be paid to over-fitting and inaccuracy at this stage as well.
6. Step #6 => Execution/Post-Execution: This stage is also called active learning. After the run, you may need to re-adjust the calibrated state to improve after additional observations (say using Kalman Filter). Further, this improvement may take the form of enhancements to parameters, Bayesian distributions, and/or hyper-parameters.

Challenges with the Supervised Learning Practice

1. Bias-Variance Trade-off: Typically lowering the bias with regard to one training group renders higher variance across the other variance groups. Optimizing for the variance across all groups inevitably ends up increasing the bias. Therefore, calibration algorithms must optimize for a combination of both (Geman, Bienenstock, and Doursat (1992), James (2003)).
2. Function Complexity vs. Sufficiency of Training Data:
    - Complicated function + lots of data => Need optimized bias/variance algorithms.

- More complex function + less training data => Need lower bias functions.
- Less Complex function => Use higher bias, and lower variance functions, no matter what amount of training data you have.

3. Dimensionality of the Input: The challenge is to avoid over-fitting in the case of high dimension inputs. Choices are:
   - Reduce the input features to a lower feature dimensionality space.
   - Hand eliminate instances of lower relevance
   - Since this is in essence similar to measurement noise, but effectively introduced due to ignoring input dimensions, this is also called deterministic noise. Techniques described in Brodely and Friedl (1999) and Smith and Martinez (2011) help identify noisy training examples and remove them.
4. Stochastic Noise in Training Output: Need to distinguish between the supervisory signal (the real response class), the supervisory target (the biased target output value), and the supervisory noise (the measurement error between the supervisory signal and the measured output). If you can characterize the nature of measurement error, use that to reduce variance (and therefore increase the bias!).
5. Training Data Heterogeneity: Many algorithms work on only one type of data, so the heterogeneous training data need to be transformed into the appropriate type (e.g., onto real-valued type for the distance classifier, logistic regression, etc.). Some techniques such as decision trees automatically handle heterogeneous training data.
6. Data Redundancy: Redundant (e.g., highly correlated) inputs result in poor performance on some algorithms (e.g., linear/logistic regressions, distance methods). Regularization helps transform the input.
7. Modeling Interactions and non-linearities: Non-interacting linear methods work well under the following situations:
   - Contributions from the individual factors/features are independent.
   - The features may be dependent, but may be transformed using a straight non-linear transformation, in which case the transformation must be specified to avoid deterministic noise.

   If the above situations do not apply, algorithms such as neural nets and decision trees work better, as they uncover the relationships more effectively.

Formulation

1. Nomenclature: The training set is

$$\{(x_1, y_1), \cdots, (x_N, y_N)\} \Longrightarrow \{X, Y\}$$

The learning algorithm seeks a function

$$g: X \longrightarrow Y$$

where g is part of the space of hypothesis functions G. Alternately, define

$$f: X, Y \longrightarrow R$$

where f is part of the space of scoring functions F. Then g is the function returning the value of y that gives the highest score, i.e.

$$g(x) = \underset{y}{\arg max}\, f(x, y)$$

2. Choice of g/f: Typically, g is chosen from the discriminant family that uses a conditional probability model

$$g(x) = P(y|x)$$

(say linear/logistic regression), whereas f is chosen from the generative family that is modeled using the joint probability

$$f(x) = P(x, y)$$

(e.g., naïve Bayes', LDA).

3. Choice of the Loss Function: Assume that $\{X, Y\}$ is an i.i.d. sample. Fitness of data is estimated by minimizing a specified loss function L defined from

$$L: Y \times Y \rightarrow R^{\geq 0}$$

4. Approaches for Loss Function Minimization from Data:
   - Empirical Loss Risk Minimization seeks the function that best fits the data, i.e., the low bias solution.
   - Structural Loss Risk Minimization seeks the function that controls the bias/variance trade-off (Vapnik (2000)).
5. Empirical Loss Minimization: Define

$$R_{Emp} = \frac{1}{N} \sum_{i} L\big(y_i, g(x_i)\big)$$

   The minimization of R_{Emp} chooses the corresponding g. If g is the conditional probability distribution, and L is the negative log likelihood, i.e.,

$$L\big(y, g(x)\big) = -\log P(y|x)$$

   then this corresponds to an MLE approach.
   - Drawbacks => When G contains many candidate functions, or the training set is not sufficiently large, empirical loss minimization leads to high variance and poor generalization, i.e., pure training data memorization/over-fitting.
6. Structural Loss Minimization: This aims to prevent over-fitting by incorporating a regularization penalty $C(g)$. For instance, when g may be expressed as a combination of linear basis functions, i.e.

$$g(x) = \sum_{i=0}^{n-1} \beta_i h(x)$$

the penalty may assume the following forms: a) The L_2 or Euclidean Norm => $\sum_{i=0}^{n-1} {\beta_i}^2$; b) The L_1Norm => $\sum_{i=0}^{n-1} |\beta_i|$; c) The L_0 Norm => The number of non-zero β_i's.

- Formulation => The challenge is to get the g that minimizes

$$J(g) = R_{Emp}(g) + \lambda C(g)$$

where λ is the bias-variance trade-off tuner;

$$\lambda = 0$$

corresponds to pure empirical risk with low bias and high variance;

$$\lambda \to \infty$$

corresponds to pure structural risk minimizer with high bias and low variance.

- Bayesian Interpretation of Structural Risk Minimization => $C(g)$ may be interpreted as $-\log P(g)$ where $P(g)$ is the joint probability, and the corresponding $J(g)$ may be interpreted as the posterior probability of g. Thus, empirical risk minimizer is frequentist, whereas structural risk minimizer is Bayesian.

7. Generative Training: This corresponds to the special case

$$f(x, y) = P(x, y)$$

and

$$L = \sum_i \log P(x_i, y_i)$$

Often these are simpler and computationally easier.

References

- Brodely, C. E., and M. A. Friedl (1999): Identifying and Eliminating Mislabeled Training Instances *Journal of Artificial Intelligence Research* **11** 131-167.
- Geman, S., E. Bienenstock, and R. Doursat (1992): Neural Networks and the Bias/Variance Dilemma *Neural Computation* **4** 1-58.
- James, G (2003): Variance and Bias for General Loss Functions *Machine Learning* **51** 115-135.
- Mohri, M., A. Rostamizadeh, and A. Talwalkar (2012): *Foundations of Machine Learning* **MIT Press**.
- Smith, M. R., and T. Martinez (2011): Improving Classification Accuracy by Identifying and Removing Instances that should be Misclassified *Proceedings of International Joint Conference on Neural Networks* 2690-2697.
- Vapnik, V. N. (2000): *The Nature of Statistical Learning Theory, 2nd Edition* **Springer Verlag**.

Unsupervised Learning

1. Definition: Unsupervised learning is the task of finding hidden structures in unlabeled data – the samples given to the learner are unlabeled, so there is no error or reward signal to evaluate a solution (Unsupervised Learning (Wiki)).
2. Techniques: Unsupervised learning is closely related to density estimation in statistics (Jordan and Bishop (2004)) and it encompasses many other techniques that seek to summarize and explain the key features of the data – e.g., data mining.
3. Approaches to Unsupervised Learning:
    a. Clustering (e.g., k-means, mixture models, hierarchical clustering)
    b. Hidden Markov Models
    c. Blind Signal Separation using Feature Extraction Techniques for Dimensionality Reduction (PCA, ICA, non-negative matrix factorization, SVD) (Acharyya (2008)).
4. Neural Network Models - SOM: The self-organizing map (SOM) is a topographic organization in which nearby locations in the map represent inputs in similar properties.
5. Neural Network Models - SRT: The Adaptive Resonance Theory (ART) Model allows the number of clusters to vary with the problem size and lets the user control the degree of similarity between members of the same clusters by means of a vigilance parameter. ART networks are used for many pattern recognition tasks such as automatic target recognition and seismic signal processing (Carpenter and Grossberg (1988)).

References

- Acharyya, R. (2008): *A New Approach for Blind Source Separation for Convolutive Sources* **VDM Verlag Dr. Mueller e K**.

- Carpenter, G. A., and S. Grossberg (1988): The ART of adaptive pattern recognition by a self-organizing neural network *Computer* **21** 77-88.
- Jordan, M. I., and C. M. Bishop (2004): *Neural Networks* **Chapman and Hall, CRC Press** Boca Raton FL.
- Unsupervised Learning (Wiki): *Wikipedia Entry for Unsupervised Learning*.

Machine Learning

Calibration vs. Training

1. Nomenclature Fix: Training is a machine learning term with no explicit attempt at establishing/inferring states. However, calibration is in one sense a true state-inference technique, constructed with a view of incorporating the specific additional constraints:
    a. A unified latent state representation across the predictor ordinate range.
    b. Explicit mathematical associative link to one/more underlying drivers.
    c. Basis hypotheses set chosen more rigorously to adhere to the underlying "physics/neumenology".
    d. Volatility and State Dynamics are afforded significant consideration in the build-out of the inferred states.
2. The Conceptual Challenge: Currently state of the art for machine learning is that it ends up delegating the challenge of choosing the basis function set to the "physics" of the problem. Thus, by this very nature, it ends up getting more widely deployed for problems that already possess a sharp cognitive clarity (e.g., image recognition).
3. Calibrated Oriented Computational Analyses: Sensitivity computation, hedge estimation (where hedge is defined as the real-world scaling/marking-to-market of latent state sensitivity), relative value computation, Custom Latent State scenario variation estimation, etc. are all primarily meaningful only in the calibration context. "Prediction" is the one operation that is meaningful to both training and calibration (although the inherent value on a single predicted value is of less epistemic significance in calibration, and opposed to training).

Pattern Recognition

Introduction

1. Purpose: Pattern Recognition is the process associated with assignment of the label to a given input value/value set, e.g.;
   - Classification => Each input value is assigned to one among the given set of classes (e. g., spam/non-spam classes).
   - Regression => Assign a real-value to an output from an input stream.
   - Sequence Labeling => Assign a class to each member of the input sequence of values (e.g., part-of-speech tagging).
   - Parsing => Assign a parse-tree to an input sentence, describing the syntactic structure of the sentence.
2. Pattern Recognition vs. Pattern Matching: Pattern Recognition algorithms aim to provide a reasonable answer for all possible inputs and to perform the "most likely" matches on the inputs, taking into account their statistical variation. Pattern matching looks for exact matches in the input against pre-existing patterns (e.g., in regular expression parsing/matching).

Supervised vs. Unsupervised Pattern Recognition

1. Learning Procedure for Supervised Learning: The learning procedure generates a model that meets possibly conflicting objectives:
   - Perform as well as possible on the hand-labeled training data.
   - Generalize as well as possible to new data.

2. Supervised vs. Unsupervised Terminology Clarification: Unsupervised equivalent of classification is called clustering, based on the notion that clustering into groups results from say, using a distance-based metric.
3. Instance of Input Data: This is the same as one part of input data that is described by a vector of features, and is used to fully characterize that instance. Features could be categorical, ordinal (e.g., fist, second etc.), integer/real valued.

Probabilistic Pattern Recognition

1. Advantages of Probabilistic Pattern Matching:
   - These output a probabilistic confidence value with each answer choice.
   - These may choose N best outcomes and their confidence values, ranked according to their inference probabilities (esp. if N is small, as in classification).
   - Chained probabilities may be readily incorporated into larger learning tasks in such as way as to either discard a choice, so as to avoid error propagation.
   - Class Probability as a Performance Metric: In addition, a) they are not affected by the relative class sizes (Mills (2011)), and b) it imposes no penalties for simply re-arranging classes.
2. Feature Selection: Feature Selection attempts to prune out redundant and/or irrelevant features (Clopinet and Elisseeff (2003)). Complexity here arises from the need to process the entire $m^n - 1$ feature power-set, given m realizations per feature. While branch-and-bound algorithms (Foroutan and Sklansky (1987)) may reduce the complexity, they become rapidly untenable for large n (Kudo and Sklansky (2000)).
3. Feature Extraction: Feature Extraction is an alternate to Feature Selection. Here the raw feature vectors are first transformed to reduce the dimensionality and the redundancy using PCA/ICA variants. The features after feature selection may appear very different from the ones before. Feature selection may still follow feature extraction.

Formulation of Pattern Recognition

1. Supervised Pattern Recognition Problem Statement: Given an unknown function

$$g: X \to Y$$

- the ground truth - that maps input instances

$$x \in X$$

to output labels

$$y \in Y$$

along with training data

$$D = \{(x_1, y_1), \cdots, (x_n, y_n)\}$$

assumed to represent accurate examples of the mapping, produce a function

$$f: X \to Y$$

that approximates g as closely as possible.

2. "As Closely as Possible": In decision theory, this is defined by specifying a loss function that assigns a specific value to the "loss" resulting from the production of an "incorrect" label. The goal is to minimize the risk/loss function.
3. Generative vs. Discriminant Probabilistic Pattern Recognition: Discriminant pattern recognition estimates the probability

$$p\left(label \mid \vec{X}, \theta\right) = f\left(\vec{X}, \theta\right)$$

of the label given the observations, where $\vec{X}$ is the feature vector over the input observations, and θ is the parameterization. With generative pattern recognition, the inverse probability of the observations given the labels $p(\vec{X} \mid label)$ is first estimated, and then combined with the prior probability $p(label \mid \theta)$ using Bayes' rule as

$$p(label \mid \vec{X}, \theta) = \frac{p(\vec{X} \mid label)p(label \mid \theta)}{\sum_{L \in \{All\ Labels\}} p(\vec{X} \mid label)p(label \mid \theta)}$$

for discrete labels and

$$p(label \mid \vec{X}, \theta) = \frac{p(\vec{X} \mid label)p(label \mid \theta)}{\int_{L \in \{All\ Labels\}}^{L \in \{All\ Labels\}} p(\vec{X} \mid L)p(L \mid \theta)dL}$$

for continuous labels.

4. MLE Inference for θ: Here, θ is estimated basically as the point-value maximum of

$$\theta^* = \arg\max_{\theta} p(\theta \mid \vec{D})$$

where

$$p(\theta \mid \vec{D}) = \left[\prod_{i=1}^{n} p(y_i \mid x_i)\right] p(\theta)$$

This is still Bayesian, except that here θ is replaced by its point-value maximum θ^*.

5. Bayesian Approach: This integrates over all possible θ, weighted according to the posterior. Bayesian lets you explicitly specify $p(\theta)$ (presumably by drawing into past experience), from

$$p\left(Label \mid \vec{X}\right) = \int_{\theta}^{\theta} p\left(Label \mid \vec{X}, \theta\right) p\left(\theta \mid \vec{X}\right) d\theta$$

Pattern Recognition Practice SKU

1. Model Selection: Identify stochastic relations between the feature vectors and the categories to be predicted. This brings both model selection and feature selection into focus (Wolpert (2001)). In general, fewer the parameters, the better.
2. Model Determinants: In addition to the simplicity and the performance of the algorithms and the models, the following criteria are also important:
    - Parametric Prior Distribution
    - Whether the classifier can cope with the missing features
    - Whether changes in class prior probabilities can be incorporated
    - Speed of the trainer, and the corresponding memory required
    - Parallelizability
    - Closeness of resemblance/proxying of human perceptions
    - Any target-specific features (i.e., in Visual Pattern Recognition, variations are needed for color, rotation, scale etc.)
3. Optimal Bayes' Classifier: The theoretically optimal Bayes' classifier minimizes the loss risk-function. When all types of mislabeling are associated with equal loss intensities (i.e., outcome A becoming B is as undesirable as outcome B becoming A), the Bayes' classifier with the minimal error rate (on the given training set) is the optimal one for the classification task.
    - Given that the error term is a convolution of a) the error magnitude/loss intensity for a given vector feature configuration, and b) the probability of the feature vector configuration, in general it is unknown what the optimal classifier type and the parameter set are. However, upper bound on the error-rate may be worked out in specific classifier schemes (e.g., in the case of K-nearest neighbor).
4. Supervised Classification Pattern Recognition Practice Steps (Pattern Recognition (Wiki)):

- Separate the available data, at random, into a training set and a test set. Test set is used only for the final performance evaluation of the trained set.
- Experiment by training a number of classification algorithms, including parametric (e.g., discriminant analysis, multinomial classifier (Glick (1973))), and non-parametric algorithms (K-nearest neighbor, support vector machines, feed-forward neural net, standard decision-tree, etc.).
- Test distribution assumptions of the continuous features – including distributions per category (Gaussian?).
- Which subset of the feature vector contributes most to the discriminative performance of the classifier?
- Work out the detailed confidence intervals for the error-rates and the class-predictions (McLachlan (2004)).
- White box vs. black box considerations may render specific classifiers unsuited for the task.

Pattern Recognition Applications

1. Pattern Recognition Applications:
   - Automatic Recognition of Sub-topic Images/Hand-writing (Duda, Hart, and Stork (2001), Milewski and Govindaraju (2008), Brunelli (2009)).
   - Identification/Authentication, e.g., License Plate Recognition, Finger-printing, face detection/verification.
   - Medical diagnostics, e.g., PAPNET/Tumor Screening.
   - Defense Navigation/Guidance, Target Recognition (Egmont-Peterson, de Ridder, and Handels (2002)).
2. Specialized Psychology Applications: As related to psychology and perception, pattern recognition may be understood as being multi-staged:

- Stage #1 => This consists of template matching, where the incoming stimuli are compared with templates (i.e., patters used to produce items of the same proportion) in the long-term memory, AND
- Stage #2 => If there is a template match, the secondary feature detection models are triggered.

References

- Brunelli, R. (2009): *Template Matching Techniques in Computer Vision: Theory and Practice* **Wiley**.
- Clopinet, I. G., and A. Elisseeff (2003): An Introduction to Variable and Feature Selection *Journal of Machine Learning Research* **3** 1157-1182.
- Duda, R. O., P. E. Hart, and D. G. Stork (2001): *Pattern Classification 2nd Edition* **Wiley**, New York.
- Egmont-Petersen, M., D. de Ridder, and H. Handels (2002): Image Processing with Neural Networks – A Review *Pattern Recognition* **35 (10)** 2279-2301.
- Foroutan, I. and J. Sklansky. (1987): Feature Selection for Automatic Classification of non-Gaussian Data *IEEE Transactions on Systems, Man, and Cybernetics* **17 (2)** 187-198.
- Glick, N. (1973): Sample Based Multinomial Classification *Biometrics* **29 (2)** 241-256.
- Kudo, M. and J. Sklansky. (2000): Comparison of Algorithms that select Features for Pattern Classifiers *Pattern Recognition* **33 (1)** 25-41.
- McLachlan, G. J. (2004): *Discriminant Analysis and Statistical Pattern Recognition* **Wiley Series in Probability and Statistics**, New Jersey.
- Milewski, R. and V. Govindaraju (2008): Binarization and Cleanup of Handwritten Text from Carbon Copy Medical Images *Pattern Recognition* **41 (4)** 1308-1315.
- Mills, P. (2011): Efficient Statistical Classification of Satellite Measurements, *International Journal of Remote Sensing* **32 (21)**.
- Pattern Recognition (Wiki): *Wikipedia Entry for Pattern Recognition*.
- Wolpert, D. H. (2001): *The Supervised Learning No Free Lunch Theorems*.

Statistical Classification

1. Classification vs. Clustering: Classification is the term used in the supervised learning context, whereas clustering is classification in the unsupervised learning context. Since there is no absolute context to go by in the case of unsupervised learning context, all the unsupervised clustering algorithms only uncover "similar" instances using similarity metric (e.g., distance metric).
2. Machine Learning as a Dimension Reduction Process: In one sense machine learning is used primarily to reduce the observation space to the "lower dimension" parameter/rules space. However, it may proceed by employing additional dimension reduction approaches internally, e.g., reduce the class features from a higher to a lower dimension space.
3. Binomial Frequentist Classification: Fisher's (Fisher (1936), Fisher (1938), Gnanadesikan (1977)) work depended on the assumption that the output probabilities were multivariate on the input features, and extracted the classification/clustering rules.
4. Multinomial Frequentist Classification: Fisher's approach was extended to multinomial output states (Rao (1952), Gnanadesikan (1977), Har-Peled, Roth, and Zimak (2003)), and further extensions to non-linear rules were done in Anderson (1958), essentially using classifications based off of the adjusted Mahalanobis distance.
5. Multinomial Bayesian Classification: This incorporates distributions of feature populations (Binder (1978)). Approximations to the clustering rules were introduced in Binder (1981) to enable tractability. MCMC, ABC, and later computational developments reduced the need for these approximations.

References

- Anderson, T. W. (1958): *An Introduction to Multivariate Analysis* **Wiley**.
- Binder, D. A. (1978): Bayesian Cluster Analysis *Biometrika* **65** 31-38.

- Binder, D. A. (1981): Approximations to Bayesian Clustering Rules *Biometrika* **68** 275-285.
- Fisher, R. A. (1936): The Use of Multiple Measurements in Taxonomic Problems *Annals of Eugenics* **7** 179-188.
- Fisher, R. A. (1938): The Statistical Utilization of Multiple Measurements *Annals of Eugenics* **8** 376-386.
- Gnanadesikan, R. (1977): *Methods for Statistical Data Analysis of Multivariate Observations* **Wiley**.
- Har-Peled, S., D. Roth, and D. Zimak (2003): Constraint Classification for Multi-class Classification and Ranking, in: *Advances in Neural Information Processing Systems 15: Proceedings of the 2002 Conference: B. Becker, S. Thrun, and K. Obermayer (eds.)*, **MIT Press**.
- Rao, C. R. (1952): *Advanced Statistical Methods in Multivariate Analysis* **Wiley**.

Linear Discriminant Analysis

Introduction

1. Definition: LDA and Fisher Linear Discriminant are related statistical methods that find a linear combination of features to identify two/more classes of objects/events underlying the data (Linear Discriminant Analysis (Wiki)).

1. Comparison LDA and related Methods:
   - LDA => Real-valued Predictors and Categorical Responses (Fisher (1936), McLachlan (2004)).
   - ANOVA => Categorical Predictors and Real-valued Responses (Wetcher-Hendricks (2011)).
   - Linear Regression => Real-valued Predictors and Real-valued Responses (Fisher (1936), McLachlan (2004)).
   - Logistic Regression => Real-valued Predictors and Categorical Responses (Fisher (1936), McLachlan (2004)).
   - Discriminant Correspondence Analysis => Categorical Predictors and Categorical Responses (Abdi (2007), Perriere and Thioulouse (2003)).
2. LDA vs. Component/Factor Analysis:
   - LDA explicitly models class difference probabilities, whereas PCA does not (Martinez and Kak (2001)).
   - Factor Analysis and PCA => Here there is no real distinction between predictors and responses, since that is not the primary purpose.

Setup and Formulation

1. Core Assumption for the 2 Class Formulation: The conditional probability of realization of each class is multinomial Gaussian on the predictors (Venables and Ripley (2002)):

$$P(\vec{x} \mid y = 0) = e^{-\frac{\left(\vec{x} - \vec{\mu}_{y=0}\right)^2}{[\sigma^2]_{y=0}}}$$

   and similar expression for $P(\vec{x} \mid y = 1)$. Here $\vec{x}$ is the feature vector of the predictor ordinates, $\vec{\mu}_{y=0,1}$ is the mean of the feature vector for each class, and $[\sigma^2]_{y=0,1}$ is the covariance of the feature vector for each class.
2. LDA Likelihood Ratios:

$$-\log\frac{P(\vec{x} \mid y = 0)}{P(\vec{x} \mid y = 1)} = -\left\{\frac{\left(\vec{x} - \vec{\mu}_{y=0}\right)^2}{[\sigma^2]_{y=0}} - \frac{\left(\vec{x} - \vec{\mu}_{y=1}\right)^2}{[\sigma^2]_{y=1}}\right\}$$

   Further, LDA assumes that

$$[\sigma^2]_{y=0} = [\sigma^2]_{y=1} = \sigma^2$$

   i.e., the feature set variance is homoscedastic.
3. Threshold-based Homoscedasticity: By applying a) the homoscedasticity assumption, b) the full rank assumption, and c) the chance that the points belonging to class 0 result from exceeding a threshold, we get the probability of belonging to class 0 as

$$\frac{\overrightarrow{x_p} \cdot \left(\vec{\mu}_{y=0} - \vec{\mu}_{y=1}\right)}{\sigma^2} > C$$

   where C is the threshold (or its transformation).
4. Hyper-plane and Hyper-surface View: The above relation is a simple linear relation among the feature set co-ordinates, and as such identifies the hyper-plane dividing the classes. Removing the homoscedasticity assumption sets the divider to be a hyper-surface.

Fisher's Linear Discriminant

1. Definition: Fisher (Fisher (1936)) defined the separation between the classes 0 and 1 to be the ratio of the variance between the classes to the variance within the classes at the vector plane $\vec{w}$:

$$S = \frac{{\sigma_{Between}}^2}{{\sigma_{Within}}^2} = \frac{[\vec{w} \cdot (\vec{\mu}_1 - \vec{\mu}_0)]^2}{\vec{w}^{\mathrm{T}}(\Sigma_0 + \Sigma_1)\vec{w}}$$

   This may be viewed as the classifier's signal-to-noise ratio.

2. Rao Multi-Class LDA: This is also referred to as Canonical Discriminant Analysis for multiple classes. Rao (Rao (1948)) generalized Fisher's LDA by defining the class variability using the sample covariance of the class means as

$$\Sigma_b = \frac{1}{C}\sum_{i=1}^{C}(\mu_i - \mu)(\mu - \mu_i)^{\mathrm{T}}$$

   The class separation is then defined as

$$S = \frac{\vec{w}^{\mathrm{T}}\Sigma_b\vec{w}}{\vec{w}^{\mathrm{T}}\Sigma\vec{w}}$$

   Thus, when $\vec{w}$ is an eigenvector of ${\Sigma_b}^{-1}\Sigma$, the corresponding separation produces the eigen-value.

3. Cross Validation Type Classification: For multiple classes, they may be partitioned, and the LDA applied individually to each partition. This provides either the C classes, or the $\frac{C(C-1)}{2}$ re-combined class set for the final classification.

4. Essence of the Fisher/Rao Approach: This approach is much more geometric/loose cognitive appeal oriented rather than the full-fledged LDA rigor developed earlier. Of course, by

applying the appropriate tweaks/resets in Fisher's formulation, the LDA approach formulation may be recovered.

5. Sample Size Impact on Fisher/LDA: Specific approaches are needed for dealing with small sample sizes. These include LDA on a reduced sub-space projection (Yu and Yang (2001)), or regularized discriminant analysis (Friedman (1989)) – the latter uses a shrinkage estimator of the covariance matrix, i.e.,

$$\Sigma_{Shrunk} = (1 - \lambda)\Sigma + \lambda I$$

where λ is the shrinkage intensity/regularization parameter (Ahdesmaki and Strimmer (2010)).

6. Extension of LDA to multiple Classes: The "one against the rest" approach may be applied class-by-class. This implies that the threshold of the original LDA partition shifts with each cross run. This also provides a natural way to accommodate multinomial ordinal classification.

Quadratic Discriminant Analysis

1. QDA/Quadratic Classifier Definition: As noted earlier, heteroscedasticity of predictor variances across classes implies that the partition surface is quadratic/conic (Quadratic Discriminant Analysis (Wiki)).
2. Transformed Basis Predictors: The predictors may be combined to produce a bigger set of convolved basis function predictors of the same order, i.e, the set of $\{x_1, x_2, x_3\}$ can be convolved to $\{x_1, x_2, x_3, {x_1}^2, {x_2}^2, {x_3}^2, x_1x_2, x_1x_3, x_2x_3\}$ as the fresh basis set for order 2.
3. Circular Special Case of QDA: This corresponds to introducing only the quadratics terms $\{{x_1}^2, {x_2}^2, {x_3}^2\}$ without the cross terms $\{x_1x_2, x_1x_3, x_2x_3\}$. This has proven to be the optimal compromise between extending the classifier's representation power and controlling the risk of over-fitting (the Vapnik-Chervonenkis dimension) (Cover (1965), Ridella, Rovetta, and Zunino (1997)).

References

- Abdi, H. (2007): Discriminant Correspondence Analysis, in *Encyclopedia of Measurement and Statistic (N. J. Salkind (ed.))* **Sage**, Thousand Oaks, CA.
- Ahdesmaki, J., and K. Strimmer (2010): Feature Selection in omics Prediction Problems using CAT Scores and False non-discovery Rate Control *Annals of Applied Statistics* **4 (1)** 503-519.
- Cover, T. M. (1965): Geometrical and Statistical Properties of Systems of Linear Inequalities with Applications in Pattern Recognition *IEEE Transactions on Electronic Computers* **EC-14 (3)** 326-334.
- Fisher, R. A. (1936): The Use of Multiple Measurements in Taxonomic Problems *Annals of Eugenics* **7** 179-188.
- Friedman, J. H. (1989): Regularized Discriminant Analysis *Journal of the American Statistical Association* **84 (405)** 165-175.
- Linear Discriminant Analysis (Wiki): *Wikipedia Entry for Linear Discriminant Analysis*.
- Martinez, A. M., and A. C. Kak (2001): PCA *versus* LDA *IEEE Transactions on Pattern Intelligence and Machine Learning* **23 (2)** 228-233.
- McLachlan, G. J. (2004): *Discriminant Analysis and Statistical Pattern Recognition* **Wiley Series in Probability and Statistics**, New Jersey.
- Perriere, G. and J. Thioulouse (2003): Use of Correspondence Discriminant Analysis to Predict the Sub-cellular Location of Bacterial Proteins *Computer Methods and Programs in Bio-medicine* **70** 99-105.
- Quadratic Discriminant Analysis (Wiki): *Wikipedia Entry for Quadratic Discriminant Analysis*.
- Rao, C. R. (1948): The Utilization of Multiple Measurements in Problems of Biological Classification *Journal of the Royal Statistical Society, Series B* **10 (2)** 159-203.
- Ridella, S., S. Rovetta, and R. Zunino (1997): Circular Back-propagation Networks for Classification *IEEE Transactions on Neural Networks* **8 (1)** 84-97.

- Venables, W. N., and B. D. Ripley (2002): *Modern Applied Statistics with S 4th Edition* **Springer Verlag**.
- Wetcher-Hendricks, D. (2011): *Analyzing Quantitative Data: An Introduction for Social Researchers* **Wiley**.
- Yu, H., and J. Yang (2001): A Direct LDA Algorithm for High-Dimensional Data - with Application to Face Recognition *Pattern Recognition* **34 (10)** 2067-2069.

Logistic Regression

Introduction

1. Definition: Probabilistic classification model used for predicting the outcome of a categorical dependent variable (e.g., class label) based on one/more of the predictor variable features (Logistic Regression (Wiki), Bishop (2006), Bhandari and Joensson (2008)).
2. Popular Applications: TRISS (Boyd, Tolson, and Copes (1987)), Voting Behavior (Harrell (2001)), Survival Analysis for Process/Product/System (Strano and Colosimo (2006), Palei and Das (2009)).

Formulation

1. Setup using a Logit Link Function: Probability of the observed instance characterized by the feature vector $\vec{x}$ belonging to the TRUE (i.e., 1) class is

$$\pi(\vec{x}) = \frac{e^{\sum_{i=0}^{n-1} \beta_i f_i(\vec{x})}}{1 + e^{\sum_{i=0}^{n-1} \beta_i f_i(\vec{x})}}$$

The logistic function (also called the link function) (Hosmer and Lemeshow (2000)) is defined as

$$g(\vec{x}) = \log \frac{\pi(\vec{x})}{1 - \pi(\vec{x})} = \sum_{i=0}^{n-1} \beta_i f_i(\vec{x})$$

2. MLE based Estimation: Since the estimation here is going to be non-linear (i.e., not closed form/quasi-analytic in the coefficients), iterative non-linear methods (such as iteratively re-

weighted least squares (IRLS), quasi-Newton (e.g., L-BGFS) need to be used (Menard (2002)).

3. Challenges with MLE:
   - Large number of predictors to cases ratio => Rule of thumb is that logistic regression models require a minimum of 10 events per variable (Peduzzi, Concato, Kemper, Holford, and Feinstein (1996)).
   - Multi-collinearity among the predictors => In this case the coefficients remain unbiased, but the errors increase, making the convergence less likely. This situation may be identified by regressing the predictors among each other.
   - Sparseness in data => This refers to a large proportion of empty cells with zero count (log of zero is undefined). To remedy this, categories may be collapsed in a meaningful way, or a harmless constant may be added to all the cells.
   - Complete Separation => In this case, the predictors deterministically predict $1/0$ without stochasticity. Such a situation may indicate a data formulation error.
4. Minimum Chi-squared Estimator for Grouped Data: In grouped data, you use the fraction of $1/0$ per each group of the feature vector, and the minimum chi-squared involves using weighted least squares to estimate a linear model of the logit (Greene (2003)).

Goodness of Fit

1. Deviance – Definition: Since sum of squares penalty does not work well in the logistic regression setup, deviance (D) as defined below (Cohen, Cohen, West, and Aiken (2002)) is used:

$$D = -2\log\frac{Likelihood_{FittedModel}}{Likelihood_{SaturatedModel}}$$

Here the "saturated model" is the model with a theoretically perfect fit. This is also referred to as the likelihood-ratio test.

2. NULL Model vs. Fitted Model Deviance: The NULL Model refers to a model with no predictors (only intercept), in comparison with the fitted model that has one/more predictors.

$$D_{NULL} = -2\log\frac{Likelihood_{NULLModel}}{Likelihood_{SaturatedModel}}$$

D_{Fitted} is the same D as before. The NULL model provides a baseline on top of which to evaluate a valid predictor/response model.

$$D_{Fitted} - D_{NULL} = -2\log\frac{Likelihood_{FittedModel}}{Likelihood_{NULLModel}}$$

3. Chi Squared Testing: In linear regression, non-significant chi-square values indicate little unexplained variance, whereas significant chi-square values indicate significant unexplained variance. Likewise, in logistic regression, significantly smaller fitted deviance than NULL deviance indicates significantly improved fit (this is analogous to the F-test used in linear regression (Cohen, Cohen, West, and Aiken (2002))).
4. Pseudo-R^2 Metrics of Goodness of Fit:

$$R_L{}^2 = \frac{D_{NULL} - D_{MODEL}}{D_{NULL}}$$

This is also called the likelihood ratio R^2, and is the most commonly used goodness of fit metric, but the drawback is that it is not monotonic with the logit.

- Cox/Snell R^2 is proportional to the logit, but reaches a maximum at 0.75 (which occurs when the maximum variance is 0.25). Nagelkerke R^2 removes this limitation (Cohen, Cohen, West, and Aiken (2002)), but exhibits wider departure with $R_L{}^2$.
- Since by construction logistic regression is heteroscedastic, all these goodness-of-fit metrics are called pseudo-R^2, as pure R^2 is of questionable value in this.

5. Hosmer-Lemeshow Test: This method uses a test statistic that asymptotically follows a χ^2 distribution to assess whether or not the observed event rates match the exected event rates in sub-groups of the model population.
6. Significance of Coefficients: The $\frac{dResponse}{dCoefficient}$ sensitivity metric of linear regression is analogous to the $\frac{dLogit}{dCoefficient}$ sensitivity metric of logistic regression (Cohen, Cohen, West, and Aiken (2002)). In addition, the significance attributed to the individual predictors is assessed using the likelihood ratio test (discussed above) or the Wald statistic (discussed below).
    - The likelihood ratio deviance test for goodness of fit discussed above may also be applied for the basis functions one-at-a-time, to incrementally assess the impact in order. The validity of these so-called hierarchical/stepwise assessment is examined in Hosmer and Lemeshow (2000), Menard (2002), and Cohen, Cohen, West, and Aiken (2002).
7. Predictor Significance Assessment using Wald Statistic: The Wald Statistic for the coefficient β_j is defined as

$$W_j = \frac{{\beta_j}^2}{{\varepsilon_j}^2}$$

where ε_j is the error estimate on β_j. The Wald Statistic is an asymptotic χ^2 distribution. Unlike the deviance likelihood ratio, the Wald Statistic is exposed in statistical packages such as SPSS, SAS, R, etc. However, the limitations with this are:

- When the regression coefficients are large, the corresponding error for a given Wald Statistic also tend to be large, thereby increasing the chances of Type-II errors;
- Wald Statistic tends to be biased when the data is sparse (Menard (2002), Cohen, Cohen, West, and Aiken (2002)).

Mathematical Setup

1. Base Mathematical Setup: The probability of the given categorical outcome is the consequence of the Bernoulli process conditioned on the realizations of the predictor ordinates and the category under consideration, i.e.

$$Y_i \mid \{x_{ij}\}_{i=1}^{m} \approx Bernoulli\ (p_i)$$

or alternatively

$$\mathbb{E}\left[Y_i \mid \{x_{ij}\}_{i=1}^{m}\right] = p_i$$

2. As a Generalized Linear Model: Here

$$Logit\ \left(\mathbb{E}\left[Y_i \mid \{x_{ij}\}_{i=1}^{m}\right]\right) = Logit\ (p_i) = \log\frac{p_i}{1-p_i} = \sum_{i=0}^{n-1}\beta_i f_i(\vec{x})$$

Thus

$$\mathbb{E}\left[Y_i \mid \{x_{ij}\}_{i=1}^{m}\right] = p_i = Logit^{-1}\left(\sum_{i=0}^{n-1}\beta_i f_i(\vec{x})\right) = \frac{1}{1+e^{\sum_{i=0}^{n-1}\beta_i f_i(\vec{x})}}$$

3. As a Latent Variable Model:

$$Y_i^{*} = \sum_{l=0}^{n-1}\beta_l f_l(\vec{x}) + \varepsilon$$

where

$$\varepsilon \approx Logistic\ (0, 1)$$

and

$$Y_i = 1$$

if

$$Y_i^* > 0$$

i.e.

$$\varepsilon < -\sum_{l=0}^{n-1} \beta_l f_l(\vec{x})$$

and

$$Y_i^* < 0$$

otherwise. This formulation can be shown to be identical to both the base setup as well as the GLM setup. Further the logistic function is symmetric with a peak around the mean just like Gaussian, but accommodates fatter tails, so it is better for calibrations.

4. As a 2-way Latent Variable Model: Here we have

$$Y_{0,i}^* = \sum_{l=0}^{n-1} \beta_{0,l} f_l(\vec{x}) + \varepsilon_0$$

and

$$Y_{1,i}^* = \sum_{l=0}^{n-1} \beta_{1,l} f_l(\vec{x}) + \varepsilon_1$$

where

$$\varepsilon_0 \approx EV_0(0,1)$$

$$\varepsilon_1 \approx EV_1(0,1)$$

and EV is a standard type-1 extreme value distribution with

$$\varepsilon_1 - \varepsilon_0 \approx Logistic\ (0,1)$$

Then the postulate

$$Y_{0i} = 1$$

if

$${Y_{0,i}}^* > {Y_{1,i}}^*$$

and 0 otherwise can be shown to be equivalent to the 1-way latent variable formulation above. As may be seen, the advantage of the 2-way formulation is that it can be extended to multi-way/multi-class.

5. Log-Linear Model: For a class C model, we may set

$$\log P(Y_i = C) = \sum_{l=0}^{n-1} \beta_{C,l} f_l(\vec{x}) - \log Z$$

Summing up across all classes, we get

$$P(Y_i = C) = \frac{e^{\sum_{l=0}^{n-1} \beta_{C,l} f_l(\vec{x})}}{\sum_i e^{\sum_{l=0}^{n-1} \beta_{i,l} f_l(\vec{x})}}$$

Thus, this produces a normalized logistic multinomial class setup.

6. As a Single-Layer Perceptron: Also called a single-layer ANN, here the Bernoulli probability p_i may be specified as

$$p_i = \frac{1}{1 + e^{-\sum_{l=0}^{n-1} \beta_l f_l(\vec{x})}}$$

This computes a single layer of continuous output in place of a step function. The derivative of p_i with respect to $\vec{x}$ is computed from the general form

$$y = \frac{1}{1 + e^{-f(\vec{x})}}$$

This choice makes this ANN identical to logistic regression, and thus maybe used in back-propagation and easy derivative extraction from

$$\frac{dy}{d\vec{x}} = y(1 - y)\frac{df}{d\vec{x}}$$

7. As Bernoulli-distributed Binomial Data: Each

$$Y_i \approx Binomial\ (n_i, p_i)$$

where Y_i is the number of successes observed. Then

$$p_i = \mathbb{E}\left[\frac{Y_i}{n_i} | \vec{X}_i\right]$$

implies that

$$Logit\ (p_i) = \log\frac{p_i}{1 - p_i} = \sum_{l=0}^{n-1} \beta_l f_l(\vec{x})$$

Equivalently

$$P\left(Y_i = y_i | \vec{X}_i\right) = C_{y_i}^{n_i} {p_i}^{y_i} (1 - p_i)^{n_i - y_i}$$

where p_i is given from above.

Bayesian Logistic Regression

1. Philosophy: Here the prior distributions are placed on the regression coefficients, which can be problematic for the logistic distribution, as conjugates for them is hard to extract.
2. Solution Approaches:
   - Perform a MAP point estimate in place of a full-fledged MAP.
   - Employ the MCMC/Metropolis-Hastings variants with the heavy-tailed multi-variate candidate distributions found by matching the mode/curvature at the normal approximation to the posterior – and then use the student-t distribution with a low degree of freedom (Bolstad (2010)). Conjugates – further, this approach may be used to sample arbitrary distributions too.
   - Use a latent variable model and approximate the logistic and/or extreme value distributions using a more tractable distribution, e.g., student-t or even a mixture of normal distribution (or even a probit distribution).
   - Use a Laplace distribution in place of the posterior distribution (Bishop (2006)). This approximates the posterior with a Gaussian (which is not too good), but the posterior mean and variance may be estimated, and schemes such as variational Bayes (Bishop (2006)) may also be used.

Logistic Regression Extensions

1. Multinomial Logit/Polytomous Regression: Multi-way categorical dependence variables with unordered values (also called “classification”).
2. Ordered Logit: Handles Ordinal Dependent (i.e., ordered) values
3. Mixed Logit: Allows for Correlations among the choices of the dependent variables.

Model Suitability Tests with Cross Validation

1. Base Methodology: Use “one-against-the-rest” selected cross-validation (Myers and Forgy (1963), Mark and Goldberg (2001)). The cross-validation sample is referred to as the “holdout”.
2. Suitability Estimation for Binary Logistic Regression: Cross-examine and validate actual and predicted values. Use the following definitions (TRUE/FALSE is defined with respect to the Holdout):
   - True Negative (TN) => Prediction – FALSE and Holdout - FALSE
   - False Negative (FN) => Prediction – FALSE and Holdout - TRUE
   - False Positive (FP) => Prediction – TRUE and Holdout - FALSE
   - True Positive (TP) => Prediction – TRUE and Holdout – TRUE
   - Accuracy => Fraction of Observations with Correct Prediction is computed as $\frac{TP+TN}{TP+FP+TN+FN}$
   - Precision => Fraction of Correctly Predicted Positives is computed as $\frac{TP}{TP+FP}$
   - Negative Predictive Value => Fraction of Correctly Predicted Negatives is computed from $\frac{TN}{TN+FN}$
   - Recall/Sensitivity => Fraction of TRUE Holdout’s Correctly Predicted is computed from $\frac{TP}{TP+FN}$
   - Specificity => Fraction of FALSE Holdout’s Correctly Predicted is computed as $\frac{TN}{FP+TN}$

References

- Bhandari, M., and A. Joenssen (2008): *Clinical Research for Surgeons* **Thieme**.
- Bishop, C. M. (2006): *Pattern Recognition and Machine Learning* **Springer** New York.
- Bolstad, W. M. (2010): *Understanding Computational Bayesian Statistics* **Wiley**.
- Boyd, C. R., M. A. Tolson, and W. S. Copes (1987): Evaluating Trauma Care: The TRISS Method - Trauma Score and the Injury Severity Score *Journal of Trauma* **27 (4)** 370-378.
- Cohen, J., P. Cohen, S. G. West, and L. S. Aiken (2002): *Applied Multiple Regression/Correlation Analysis for the Behavioral Sciences 3rd Edition* **Routledge**.
- Greene, W. H. (1993): *Econometric Analysis 5th Edition* **Prentice Hall**.
- Harrell, F. (2001): *Regression Modeling Strategies* **Springer-Verlag**.
- Hosmer, D. W. and S. Lemeshow (2000): *Applied Logistic Regression 2nd Edition* **Wiley**.
- Logistic Regression (Wiki): *Wikipedia Entry for Logistic Regression*.
- Mark, J., and M. A. Goldberg (2001): Multiple Regression Analysis and Mass Assessment: A Review of the Issues *Appraisal Journal* 89-109.
- Menard, S. W. (2002): *Applied Logistic Regression 2nd Edition* **SAGE**.
- Myers, J. H. and E. W. Forgy (1963): The Development of Numerical Credit Evaluation Systems *Journal of American Statistical Association* **58 (303)** 799-806.
- Palei, S. K., and S. K. Das (2009): Logistic Regression Model for Prediction of Roof Fall Risks in Board and Pillar Workings in Coal Mines: An Approach *Safety Science* **47** 88.
- Peduzzi, P., J. Concato, E. Kemper, T. R. Holford, and A. R. Feinstein (1996): A Simulation Study of the Number of Events per Variable in Logistic Regression Analysis *Journal of Clinical Epidemiology* **49 (12)** 1373-1379.
- Strano, M., and B. M. Colosimo (2006): Logistic Regression Analysis for Experimental Determination of Forming Limit Diagrams *International Journal of Machine Tool and Manufacture* **46 (6)**.

Multinomial Logistic Regression

Introduction

1. Logistic Regression Calibration Data Sufficiency: Given that the logit itself is a continuous function of the predictor, you need a sufficient number of observations per each predictor ordinate instance. This can happen if you specify a bucket (infinitesimal or otherwise) and proxy the predictor ordinate instance to a central measure, and the response to the logit of the probability. Of course, all other observations regarding data sufficiency for this and notes regarding non-linearity of the calibrations still hold.
2. Multinomial Logistic Regression Definition: This is a logistic regression model allowing more than two categorical outcomes (Greene (1993)). This is also referred to as softmax or multinomial logit regression.
3. Normalization Constraint for Multinomial Logistic Regression: This would imply that the logit of the last class is determined ENTIRELY from the remaining predictor<->class representations; therefore, no exogenous spline-based basis functional specification is possible for this.
4. Ordered Multinomial Logistic Regression: Roving/pregressing thresholds are a natural fit for ordered multinomial logits. LDA supports this by construction; vanilla multinomial logistic regression may be extended relatively easily to do this.

Setup and Formulation

1. Assumption of IIA (Independence of irrelevant Alternatives): This assumption states that the odds of preferring one class over another does not depend on the presence of other “irrelevant” alternatives. If violation of this occurs, other models such as nested logit or multinomial probit may be used.

2. Independent Binary Regressors: Consider a conditional universe where the outcome subset is (j, k), where k is the reference/pivot class. The logit P_j is given from

$$\log \frac{P(y = j|\vec{x})}{P(y = k|\vec{x})} = \sum_{i=0}^{n-1} \beta_{ij} f_i(\vec{x})$$

Thus, this introduces separate sets of regression coefficients, one for each j. Exponentiating the above, we get

$$P(y = j|\vec{x}) = P(y = k|\vec{x}) \times e^{\sum_{i=0}^{n-1} \beta_{ij} f_i(\vec{x})}$$

Summing the probabilities to unity, i.e.,

$$\sum_{j=1}^{K} P(y = j|\vec{x}) = 1$$

we get

$$P(y = k|\vec{x}) = \frac{1}{1 + \sum_{\substack{j=1 \\ j \neq k}}^{K} e^{\sum_{i=0}^{n-1} \beta_{ij} f_i(\vec{x})}}$$

The fact that we used multiple regression runs off of a single pivot reveals the reliance on the assumption of IIA.

3. Log Linear Model: Setup the class probability as

$$\log P(y = k|\vec{x}) = \sum_{i=0}^{n-1} \beta_{ij} f_i(\vec{x}) - \log Z$$

Summing the probabilities again yields

$$Z = \sum_{j=1}^{K} e^{\sum_{i=0}^{n-1} \beta_{ij} f_i(\vec{x})}$$

where Z is referred to as the partition function. Z is not a function of the observation set – it is a function of $\vec{x}$ and β_{ij}.

4. Softmax Function:

$$SoftMax(k, \{x_i\}_{i=0}^{n-1}) = \frac{e^{x_k}}{\sum_{i=0}^{n-1} e^{x_k}}$$

The impact of exponentiating x_i is to exaggerate the difference between x_i and the others. Thus,

$$SoftMax(k, \{x_i\}_{i=0}^{n-1}) \to 0$$

if

$$x_i \ll \max(\{x_i\}_{i=0}^{n-1})$$

and

$$SoftMax(k, \{x_i\}_{i=0}^{n-1}) \to 1$$

if

$$x_i \sim \max(\{x_i\}_{i=0}^{n-1})$$

Thus, softmax can be used to construct a weighted average that behaves as a smooth function (differentiable easily, etc.) as an approximation to the non-smooth function $\max(\{x_i\}_{i=0}^{n-1})$, i.e.,

$$f(\{x_i\}_{i=0}^{n-1}) = \max(\{x_i\}_{i=0}^{n-1}) \approx \sum_{i=0}^{n-1} x_i SoftMax(k, \{x_i\}_{i=0}^{n-1})$$

5. Latent Variable Model: For each data point p and outcome j, there exists a continuous latent variable distributed as

$$Y_{p,j}{}^{*} = \sum_{i=0}^{n-1} \beta_{ij} f_i(\vec{x}) + \varepsilon_j$$

where

$$\varepsilon_j \sim EV_1(0,1)$$

a standard type-1 extreme value distribution. $Y_{p,j}{}^{*}$ may be thought of as the utility associated with the data point p choosing an outcome j, where some randomness is used to account for the unmodeled factors.

- Actual Outcome Probability => The actual outcome j occurs only when

$$Y_{p,j}{}^{*} > Y_{p,k}{}^{*}$$

for all

$$j \neq k$$

Thus

$$P(Outcome \equiv j) = P\left(Y_{p,j}{}^{*} > Y_{p,k}{}^{*} \; \forall \; k \neq j\right)$$
$$= P\left(\sum_{i=0}^{n-1} \left(\beta_{ij} - \beta_{ik}\right) f_i(\vec{x}) > \varepsilon_j - \varepsilon_k \; \forall \; k \neq j\right)$$

- Nature of $\varepsilon_j - \varepsilon_k$ => Since both ε_j and ε_k are $EV_1(0,1)$, by definition $\varepsilon_j - \varepsilon_k$ then becomes

$$\varepsilon_j - \varepsilon_k \sim Logistic(0,1)$$

In fact, the generalized $\varepsilon_j - \varepsilon_k$ may also be shown to be scale-separable, i.e.,

$$\varepsilon_j - \varepsilon_k \sim b\ Logistic(0,1) = Logistic(0,b)$$

References

- Greene, W. H. (1993): *Econometric Analysis 5th Edition* **Prentice Hall**.

Decision Trees and Decision Lists

1. Decision Tree: A decision tree is a decision support tool that uses a tree-like graph or model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility (Decision tree (Wiki), Yuan and Shaw (1995)).
2. Drawback of Decision Trees: General inability to handle multi-valued, complex, and uncertain paths; in particular, for data including categorical variables with different number of levels, information gain in decision trees are biased in favor of those attributes with more levels (Deng, Runger, and Tuv (2011)).
3. Decision Lists: Decision lists are representations for Boolean functions. They are more expressive than conjunctions/disjunctions, but are typically less expressive than the disjunctive/conjunctive normal forms (Decision List (Wiki), Rivest (1987)).
    - In particular, decision lists are useful for efficient attribute learning (Klivans and Servedio (2006)).

References

- Decision List (Wiki): Wikipedia Entry for Decision List.
- Decision Tree (Wiki): Wikipedia Entry for Decision Tree.
- Deng, H., G. Runger, and E. Tuv (2011): Bias of Importance Measures for Multi-Valued Attributes and Solutions *Proceedings of the 21st International Conference on Artificial Neural Networks (ICANN)*.
- Klivans, A. R., and R. A. Servedio (2006): Toward Attribute Efficient Learning of Decision Lists and Parities *Journal of Machine Learning Research* **7 (12)** 587-602.
- Rivest, L. R. (1987): Learning Decision Lists *Machine Learning* **2 (3)** 229-246.
- Yuan, Y., and M. J. Shaw (1995): Induction of Fuzzy Decision Trees *Fuzzy Sets and Systems* **69** 125-139.

Variable Bandwidth Kernel Density Estimation

1. Definition: Variable bandwidth kernel density estimators are a form of kernel density estimator in which the size of the kernels used in the estimate are varied depending upon on either the location of the samples or the location of the test point (Variable Kernel Density Estimation (Wiki), Terrell and Scott (1992)).
2. Setup: Given a set of samples $\{\vec{x}_i\}$ we aim to estimate the point density $P(\vec{x})$ at the test point $\vec{x}$, i.e.,

$$P(\vec{x}) \approx \frac{W}{nh^D}$$

where

$$W = \sum_{i=0}^{n-1} w_i$$

and

$$w_i = K\left[\frac{\vec{x} - \vec{x}_i}{h}\right]$$

Here n is the number of samples, h is the width, and D is the number of dimensions in $\vec{x}$. If K is chosen to be linear in $\vec{x}$, it may be imagined as a simple, linear band-pass filter.

3. Balloon Estimation: In Balloon Estimation, the kernel width is varied to make it proportional to the density at the test point $\vec{x}$, i.e.,

$$h \approx \frac{g}{[nP(\vec{x})]^{1/D}}$$

where g is a constant. This results in a constant W across samples, and produces a generalization of kNN - i.e., a uniform kernel function returns the unbiased kNN evaluator (Mills (2011)).

4. Pointwise Estimation: Here, the kernel width is altered with respect to the location (Terrell and Scott (1992)). For multivariate distributions, h can be varied on the shape - not just the size.
5. Variable Kernel Density in Statistical Classification: In this approach, the probability distribution functions of each class is computed separately with its own kernel, custom bandwidth, etc. (Taylor (1997)). In an alternate approach, the sum of each class is divided across the sample variate space, i.e.,

$$P(j,\vec{x}) \approx \frac{1}{n}\sum_{\substack{i=0 \\ C_i=j}}^{n-1} w_i$$

where

$$C_i = j$$

is the class of the i[th] sample.

- Decision Boundary => Say you are classifying for classes 1 and 2 using any smooth kernel (say Gaussian) – this makes sure that the estimates of joint or conditional probabilities both continuous and differentiable. The border is searched by zeroing the difference between the conditional probability as

$$R(\vec{x}) = \frac{P(2|\vec{x}) - P(1|\vec{x})}{P(2|\vec{x}) + P(1|\vec{x})}$$

Any $1D$ root finding algorithm for

$$R(\vec{x}) = 0$$

establishes the border straddling samples.

- The Classification Work-out =>
  - Spot the point index closest to $\vec{x}$ as

$$j = \underset{i}{\arg min} \left| \vec{b}_i - \vec{x} \right|$$

  - The gradient shift at j is

$$GradientShift = \left(\vec{x} - \vec{b}_j \right) \cdot \vec{\nabla}_{\vec{x}} R(\vec{x})_{\vec{x}=\vec{b}_j}$$

  - The estimated class then would be

$$C = \frac{1}{2}\left[3 + \frac{p}{|p|}\right]$$

References

- Mills, P. (2011): Efficient Statistical Classification of Satellite Measurements, *International Journal of Remote Sensing* **32 (21)**.
- Taylor, C. (1997): Classification and Kernel Density Estimation *Vistas in Astronomy* **41 (3)** 417-441.
- Terrell, D. G., and D. W. Scott (1992): Variable Kernel Density Estimation *Annals of Statistics* **20** 1236-1265.
- Variable Kernel Density Estimation (Wiki): Wikipedia Entry for Variable Kernel Density Estimation.

k-Nearest Neighbors Algorithm

1. Definition: kNN is a non-parametric method for classification and regression, which predicts object values and class memberships based on the k closes training examples in the feature space (K-Nearest Neighbor (Wiki), Altman (1992)).
2. kNN Motivation: kNN is a type of instance-based learning or lazy learning where the function is only approximated locally (e.g., using majority voting) and all computation is deferred until classification.
3. kNN Regression: Here the property value of the object is the average of its k nearest neighbors, perhaps with their corresponding weights applied (i.e., $\frac{1}{d}$ inverse distance metric).
4. kNN Implict Decision Boundary: Since kNN rules implicitly compute the decision boundary, it is sensitive to the local structure of the data. Further since it is possible to compute the decision boundary explicitly and efficiently (SVM techniques do it in a non-probabilistic sense), the classification computational complexity is a function of the decision boundary computation complexity (Bremner, Demaine, Erickson, Iacono, Langerman, Morin, and Touissant (2005)).
5. kNN Distance Metric: Euclidean distance metric is commonly employed for real-valued continuous feature vector classifications. For discrete variables, alternatives such as the overlap metric (e.g., the Hamming distance) may be used. Classification accuracy maybe improved significantly if the distance metrics are learned using specialized algorithms such as Large Margin Nearest Neighbor or Neighborhood Component Analysis.
6. Drawbacks of the Majority Voting Technique: Classification skew may be introduced owing to potential dominance of one of the candidate classes – which, in turn, may be addressed by distance weighting techniques (Coomans and Massart (1982)).
7. Classification Accuracy Improvement through Improved Data Representation: A good instance for such a representation would be the self-organizing map (SOM). In a SOM, each node is a representative center of a cluster of similar points (i.e., each node would be a source density field). kNN may then be applied to the SOM.

8. Choice of k in kNN: While large k reduces the impact of noise/variation in the classification, it ends up making the boundaries less distinct and more diffuse (Everitt, Landau, Leese, and Stahl (2011)). Appropriate k for the problem set may be chosen using targeted heuristic techniques (such as hyperparameter optimization).
9. Noisy/Irrelevant Feature Reduction: Since kNN accuracy maybe severely degraded by the presence of noisy/irrelevant features, much effort has been put into reducing them. Typical approaches include:
    a. Use of evolutionary algorithms to optimize feature scaling (Nigsch, Bender, van Buuren, Tissen, Nigsch, and Mitchell (2006)).
    b. Scaling the features by using mutual information across the training data and the training classes (this approach must be similar to cross validation).
10. Binary kNN Classification (2 Class Classification): Here it is useful to choose k to be an odd number to avoid ties. An empirically optimal value for k maybe found out using the bootstrap method (Hall, Park, and Samworth (2008)).
11. kNN Customization using Feature Scaling: Although the kNN approach is treated as a non-parametric method, it may be customized by working out an optimal k. This is indeed the case, for example, in metric-based classifications, SOM approaches, noisy/irrelevant feature reduction, etc.
12. Efficient kNN Approaches: Naïve kNN approaches that aim to compute distance metrics to all the data points are often computationally intractable, so approaches seek to use a "small enough" k. The consequence of these approaches is that they reduce the number of distance metric and/or feature scaling evaluations performed. This is the reason approaches such as variable bandwidth kernel density balloon estimators with a uniform kernel are among the most common kNN approaches (Terrell and Scott (1992), Mills (2001)).
13. kNN Asymptotic Error Rates: The kNN approach has strong consistency results. As sample size approaches infinity, the kNN approach produces an error rate that is less than twice the best possible error rate achievable - viz. the Bayes' error rate (Cover and Hart (1967)) for some value of k. Further improvements may be possible through the use of proximity graphs (Touissant (2005)).
14. Feature Extraction/Reduction Example using the kNN Approach: An example of a typical computer vision computation pipeline for face recognition using kNN includes the feature

extraction and dimension reduction pre-processing steps (this is implemented in OpenCV) is as follows:

a. Haar Face Detection.
b. Mean-Shift Tracking Analysis.
c. PCA or Fischer LDA Projection onto the Feature Space.
d. kNN Classification.

15. kNN Curse of Dimenstionality: For high dimensions, Euclidean and other similar distance metrics become unhelpful and/or computationally infeasible, so dimension reduction precedes kNN (Beyer, Goldstein, Ramakrishnan, and Shaft (1999)).
16. Low-Dimensional Embedding: Here feature extraction and dimension reduction are combined together in one step using PCA/LDA/CCA as a pre-processing step before applying kNN (Shaw and Jebara (2009)).
17. High Dimensional Tick Dataset: For very high dimensional real-time/tick datasets (e.g., when performing similarity search on live video streams, DNA data, or high-dimensional time series), running a fast approximate kNN search using locality sensitive hashing, random projections (Bingham and Mannila (2001)), sketches (Shasha (2004)), or other high dimension similarity search techniques from the VLDB tool-box may be the only feasible option.
18. kNN Data Reduction: Since only some points are needed for accurate classification (these are called prototypes), they may be identified as follows:
    a. Select the class outliers – the training data those are incorrectly classified by kNN for a given k.
    b. Separate the remainder into two sets:
        i. The prototypes that are to be used for classification decisions
        ii. The absorbed points that can correctly classified by kNN using prototypes, and therefore be removed from the training set
19. Reasons for Outliers: Outlier is an instance in the training data that is surrounded by instances of other classes. Outliers may occur due to:
    a. Random Error
    b. Insufficient training examples of this class (an isolated example occurs instead of a cluster)

c. Missing important inputs (classes may be separated in dimensions that we do not as yet know about)
d. Too many training instances of other classes (unbalanced classes) that create a hostile background for the given small class

20. $(k,r)NN$ Class Outlier: Given 2 numbers

$$k > r > 0$$

a training example is called a $(k,r)NN$ class outlier if its k nearest neighbors include more than r examples of other classes. Class Outliers should be detected and separated.

21. Condensed Nearest Neighbor (CNN) Algorithm: This is also called the Hart's algorithm (Hart (1968)), CNN selects a set of prototypes U from the training set such that $1NN$ with U can classify the examples almost as accurately as $1NN$ does with the full data set.
    a. Hart's Algorithm => Given a training set X CNN works with full iterative scans:
        i. Scan all elements of X looking for element $\mathcal{X}$ whose nearest prototype from U has a different label than that of $\mathcal{X}$.
        ii. Move $\mathcal{X}$ from X onto U.
        iii. Repeat until there are no more prototypes to add to U.

22. Mirkes' Border Ratio: For the scan above, it is efficient to scan the training examples in the order of decreasing border ratio (Mirkes (2011)), defined as

$$a(x) = \frac{\|x' - y\|}{\|x - y\|}$$

where $\|x - y\|$ is the distance between x and y where y has a different color than x, and $\|x' - y\|$ is the distance between x' and y where y has a different color than x'. This ratio stays inside the bracket $[0, 1)$, as $\|x' - y\|$ never exceeds $\|x - y\|$.

23. kNN Regression: This consists of a typical inverse distance weighted kNN algorithm:
    a. Compute the Euclidean/Mahalanobis distance from the query instance to the labeled instances.
    b. Order the labeled instances by their distance metrics.

c. Find a heuristically optimal k based on RMSE, and using cross validation.
d. Compute the inverse distance weighted average among the k nearest multivariate neighbors.

24. *kNN* Accuracy Validation: This is typically done using either a confusion matrix (for binary classifications) or a matching matrix (for multi-class classifications). Likelihood ratio tests are also often applied.

References

- Altman, N. S. (1992): An Introduction to Kernel and Nearest-Neighbor Non-parameteric Regression *American Statistician* **46 (3)**: 175-185.
- Beyer, K., J. Goldstein, R. Ramakrishnan, and U. Shaft (1999): When is Nearest Neighbor Meaningful? *Database Theory – ICRT '99* 217-235.
- Bingham, E., and H. Mannila (1996): Maximum Entropy Approach in Natural Language Processing *Computational Linguistics* **22 (1)**: 39-71.
- Bremner, D., E. Demaine, J. Erickson, J. Iacono, S. Langerman, P. Morin, and G. Touissant (2005): Output Sensitive Algorithms for Computing Nearest Neighbor Decision Boundaries *Discrete and Computational Geometry* **33 (4)** 593-604.
- Coomans, D., and D. L. Massart (1982): Alternative k-Nearest Neighbor Rules in Supervised Pattern Recognition: Part I: k-Nearest Neighbor Classification using Alternative Voting Rules *Analytica Chimica Acta* **136** 15-27.
- Cover, T. M., and P. E. Hart (1967): Nearest Neighbor Pattern Classification *IEEE Transactions on Information Theory* **13 (1)** 21-27.
- Everitt, B. S., S. Landau, M. Leese, and D. Stahl (2011): Miscellaneous Clustering Methods, in *Cluster Analysis, 5th Edition* **John Wiley and Sons**, Chichester, UK.
- Hall, P., B. U. Park, and R. J. Samworth (2008): Choice of Neighbor Order in Nearest-Neighbor Classification *Annals of Statistics* **36 (5)** 2135-2152.
- Hart, P. E. (1968): The Condensed Nearest Neighbor Rule *IEEE Transactions on Information Theory* **18** 515-516.

- K-Nearest Neighbors Algorithm (Wiki): Wikipedia Entry for K-Nearest Neighbors Algorithm.
- Mills, P. (2011): Efficient Statistical Classification of Satellite Measurements, *International Journal of Remote Sensing* **32 (21)**.
- Mirkes, E. M. (2011): kNN and Potential Energy **University of Leicester**.
- Nigsch, F., A. Bender, B. van Buuren, J. Tissen, E. Nigsch, and J. B. Mitchell (2006): Melting-Point Prediction employing k-Nearest Neighbor Algorithms and Genetic Parameter Optimization *Journal of Chemical Information and Modeling* **46 (6)** 2412-2422.
- Shasha, D. (2004): *High-Performance Discovery in Time Series* **Springer** Berlin.
- Shaw, B., and T. Jebara (2009): Structure Preserving Embedding *Proceedings of 26th Annual International Conference on Machine Learning, ACM 2009*.
- Terrell, D. G., and D. W. Scott (1992): Variable Kernel Density Estimation *Annals of Statistics* **20** 1236-1265.
- Touissant, G. T. (2005): Geometric Proximity Graphs for improving Nearest Neighbor Methods in Instance-based Learning and Data-Mining *International Journal of Computational Geometry and Applications* **15 (2)** 101-150.

Perceptron

1. Definition: Perceptron is an algorithm for supervised classification of the input to one of the several possible binary outputs (Perceptron (Wiki), Rosenblatt (1957), Rosenblatt (1958), Rosenblatt (1962)).
2. Perceptron as a Linear Classifier: The perceptron makes its prediction based on a linear predictor function that combines a set of weights with a feature vector describing a given input state using the delta rule.
3. Perception as an Online Algorithm: The learning algorithm is an online algorithm, as it processes elements in the training set one at a time.
4. Setup: It is a binary classifier that maps the input feature vector $\vec{x}$ to an output value $f(\vec{x})$ – a single binary value, as

$$f(\vec{x}) = \begin{cases} 1 & \vec{w} \cdot \vec{x} + b > 0 \\ 0 & otherwise \end{cases}$$

   where $\vec{w}$ is a real-valued vector of weights, and b is the bias term that is independent of the input. Spatially, the bias alters the position, but not the orientation, of the decision boundary.
5. Linearly Separable Learning Set: The perceptron learning algorithm does not terminate if the learning set is not linearly separable, and then the classification will be achieved (e.g., Boolean XOR). The solution spaces of decision boundaries for all the binary functions and their learning behaviors are studied in Liou, Liou, and Liou (2013).
6. From an ANN perspective: A perceptron may be viewed as an artificial neuron that uses the Heaviside step function as the activation function – this is also referred to as the single-layer perceptron (as opposed to a multi-layer perceptron, which is just a neural network). As a linear classifier, single layer perceptron is the simplest feed forward neural network.
    a. The multi-layer perceptron learning algorithm => Here, a hidden layer exist, so that the algorithms such as back-propagation should be used (Minski and Papert (1969)). Alternatively, methods such as delta-rule can be used if the perceptron function is

non-linear and differentiable. Finally, each neuron in the ANN operates independently of the others – so the learning outputs may be considered in isolation.

7. Perceptron Setup:

$$y = f(\vec{z})$$

is the learning perceptron function for the input vector $\vec{z}$. The bias b is assumed to be zero here, and

$$\{\vec{D}\} \Longrightarrow \{\vec{x}_j, d_j\}_{j=1}^{S}$$

is the training set of S samples. $\vec{x}_j$ is the feature vector for sample j with

$$i = 0, \cdots, n-1$$

feature components.

$$\vec{w}(t) = \{w_i(t)\}_{i=1}^{n-1}$$

corresponds to the feature vector weights at time t. α is the learning rate with the restriction

$$0 < \alpha \leq 1$$

8. The Algorithm:
   a. Initialization => Initialize the weight vector and the error threshold γ; weights may be initialized to zero, or a small random value.
   b. Update $y_j(t)$ => For each example j in our training set $\{D\}$, compute the output $y_j(t)$ from the following:

$$y_j(t) = f\left(\vec{w}(t) \cdot \vec{x}_j\right) = f\left(w_0(t)x_{j,0} + w_1(t)x_{j,1} + \cdots + w_{n-1}(t)x_{j,n-1}\right)$$

which can be generalized to automatically incorporate the plane intercept bias onto the feature vector set.

c. Update the weights =>

$$w_i(t+1) = w_i(t) + \alpha \sum_j \left[d_j - y_j(t)\right] x_{j,i}$$

for all

$$0 \le i < n$$

d. Iterate until convergence => Repeat the past two steps until the error

$$\frac{1}{S}\sum_{j=1}^{S}\left[d_j - y_j(t)\right] \le \gamma$$

(or bail if the pre-determined number of steps have been reached). The previous two steps immediately update the weights to the given pair rather than wait until all the pairs have undergone the steps.

9. Separability: The training set $\{D\}$ is said to be linearly separable if the positive examples can be separated from the negative examples using a hyperplane – that is, there exists a positive constant vector $\vec{\zeta}$ AND a weight vector $\vec{w}$ such that

$$\left(\vec{w} \cdot \vec{x}_j + b\right) d_j > \zeta_j$$

for all j.

a. Linear Separable Convergence from Below => From below, along the iteration, the weight vector always gets adjusted by a bounded amount in the direction it has

negative dot product with, thus getting bounded above by $\mathcal{O}\left(\sqrt{t}\right)$, where t is the number of steps.

b. Linear Separable Weight Vector Convergence from Below => The weight vector gets bounded from below by $\left(\frac{2\kappa}{\zeta}\right)^2$ where κ is the maximum norm of the input vector (Novikov (1962)).

c. Combining Above and Below => Combining the observations listed in a) and b), linearly separable data sets cause the perceptron algorithm to converge after a finite number of steps. In fact, these steps and their convergence characteristics are really just the single-layer ANN analogue of matrix linear reduction techniques (such as Gauss-Jordan, Newton's, or Caylet-Hamilton convergence techniques), and as such, all the criteria that cause non-convergence apply here too.

10. Invariance of the Decision Boundary to the Scaling of the Weight Vector: The decision boundary of the perceptron is invariant to the scaling of the weight vector, i.e., a perceptron trained with an initial weight vector $\vec{w}$ and a learning rate α behaves identically to a perceptron trained with with an initial weight of $\frac{\vec{w}}{\alpha}$ and learning rate unity.

11. Perceptron Variants:

a. Gallant's Pocket Algorithm => The pocket algorithm with ratchet (Gallant (1990)) improves efficiency by returning the best solution in the pocket rather than the latest solution. This may also be used for non-linearly separable data-sets, where the aim is to identify the solution with the least misclassification.

b. Perceptron with the Largest Separation Margin => Also referred to as to the perceptron of optimal stability, these techniques employ iterative training and optimization schemes such as the Min-Over algorithm (Krauth and Mezard (1987)) or the AdaTron algorithm (Anlauf and Biehl (1989)). The perceptron of optimal stability, together with the kernel trick, serve as the conceptual foundations for the support vector machines techniques.

c. α-Perceptron => This technique uses a pre-processing layer of random fixed weights along with a set of feature-thresholded units. Among other applications, these perceptron variants are used to classify analogue patterns by projecting them onto the binary space (after digitizations, of course).

d. Multi-layer Perceptrons => By adding extra non-linear layers between the inputs and the outputs, one can separate all data, and can model any well-defined function according to an arbitrary precision given enough training data. This generalization is called the multi-layer perceptron.

12. High-Dimension Separability: "High dimensions" refers to high feature vector dimensions. Sure enough, if there are as many feature vectors as samples, one WILL achieve complete separation – through straightforward I/O mapping!
13. Higher Order Networks ($\sigma - \pi$ Units): These techniques help address non-linear perceptron functions without using multiple layers. Here each element of the input vector is extended with a pairwise combination of multiplied inputs (aka the quadratic classifier). This may also be applied to the multi-layer network.
14. "Best Classifier": Remember that the best classifier need not classify all the training data perfectly (i.e., it need not always uncover a hyperplane boundary). As an example, given the constraint that the input data each come from its own Gaussian distribution, the logistic regression/LDA might turn out be optimal classifiers.
15. Multi-class Perceptron: The multi-class perceptron may be defined as follows: Given an input feature vector $\vec{x}$ and an output class vector $\vec{y}$, the feature representation function $\vec{f}(\vec{x}, \vec{y})$ maps each possible input/output pair to a finite-dimensional real-valued feature vector. The score that results from the feature vector/weight vector dot product is used to choose among the many possible outputs, i.e.,

$$\hat{y} = \underset{y}{\arg max} \left[\vec{f}(\vec{x}, \vec{y}) \cdot \vec{w}\right]$$

16. Iterative Multi-class Perceptron Learning: As in the binary classification, multi-class classifications again iterate over training instances, predicting an output for each example, leaving the weights unchanged when the predicted output matches the target, and changing them when the predictions don't match:

$$w_{t+1} = w_t + f(x, y) - f(x, \hat{y})$$

As expected, this reduces to the original binary classifier when x is a real-valued vector, y is chosen from $(0, 1]$, and

$$f(x, y) = xy$$

17. Efficient Feature-Score Calculations: Efficient algorithms are available to calculate the feature score

$$\hat{y} = \underset{y}{\arg max} \left[\vec{f}(\vec{x}, \vec{y}) \cdot \vec{w}\right]$$

under specific problem spaces:

   a. In NLP for tasks such as part-of-speech tagging and syntactic parsing (Collins (2002)).
   b. Large-scale machine learning in a distributed computing setting (McDonald, Hall, and Mann (2010)).

18. Perceptron Extensions: Margin Bound guarantee for the Kernel Perceptron algorithm was proposed initially by Aizerman, Braverman, and Rozonoer (1964), and extended to the general non-separable case by Freund and Shapire (1998), Freund and Shapire (1999), and Mohri and Rostamizadeh (2013).

References

- Aizerman, M. A., E. M. Braverman, and L. I. Rozonoer (1964): Theoretical Foundations of the Potential Function Method in Pattern Recognition Learning *Automation and Remote Control* **25** 821-837.
- Anlauf, J. K., and M. Biehl (1989): The AdaTron: An Adaptive Perceptron Algorithm *Europhysics Letters* **10** 687-692.

- Collins, M. (2002): Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with the Perceptron Algorithm *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP '02)*.
- Freund, Y., and R. E. Schapire (1998): Large Margin Classification using the Perceptron Algorithm *Proceedings of the 11th Annual Conference on Computational Learning Theory (COLT'98)* **ACM Press**.
- Freund, Y., and R. E. Schapire (1999): Large Margin Classification using the Perceptron Algorithm *Machine Learning* **37 (3)** 277-296.
- Gallant, S. I. (1990): Perceptron-Based Learning Algorithms *IEEE Transactions on Neural Networks* **1 (2)** 179-191.
- Krauth, W., and M. Mezard (1987): Learning Algorithms with Optimal Stability in Neural Networks *J. of Physics A: Math. Gen.* **10** L745-L752.
- Liou, D. R., J. W. Liou, and C. Y. Liou (2013): *Learning Behaviors of Perceptron* **Concept Press**.
- McDonald, R., K. Hall, and G. Mann (2010): Distributed Training Strategies for the Structured Perceptron *Association for Computational Linguistics* 456-464.
- Minsky, M. L., and S. A. Papert (1969): *Perceptrons* **MIT Press** Cambridge, MA.
- Mohri, M., and A. Rostamizadeh (2013): Perceptron Mistake Bounds.

Support Vector Machines (SVM)

1. Definition: SVM Model is a representation of the training data as points in space, mapped such that the examples of the distinct categories are separated by a clear gap that is as wide as possible (Support Vector Machine (Wiki), Cortes and Vapnik (1995)).
2. SVM Kernel Trick: In addition to performing linear classifications, SVM's can efficiently perform a non-linear classification using the kernel trick (i.e., mapping their inputs to high-dimensional space using custom kernel basis functions).
    a. The kernel function is essentially the same as a basis function, i.e., the feature vector set appears only as arguments to the kernel function during the entire formulation, thus the earlier spline basis function analysis we did should be usable here.
3. Hyperplane Construction: An SVM constructs a hyperplane or a set of hyperplanes in a high or infinite dimensional space, which may be used for classification, regression, or other tasks.
    a. Intuition behind the Hyperplane Construction => A good separation is achieved by the hyperplane that has the largest distance to the nearest training data point of any class (the so-called function margin), since, in general, larger the function margin, lower the generalization error of the classifier (Press, Teukolsky, Vetterling, and Flannery (2007)).
    b. Alternate names for the hyperplane => This is also called the maximum margin hyperplane (for a p-dimension feature vector, we will need a $p-1$ dimensional hyperplane). The corresponding linear classifier is then referred to as the maximum margin classifier, or the perceptron of optimal stability.
4. Binary Classification Linear SVM: Given a sample set

$$D = \{(x_i, y_i): x_i \in R^p, y_i \in [-1, +1]\}_{i=1}^{n}$$

our intent is to find out the maximum margin hyperplane that separates

$$y_i = +1$$

from

$$y_i = -1$$

i.e.,

$$\vec{w} \cdot \vec{x}_i - b = \pm 1$$

The corresponding offset that we chose to maximize is $\frac{b}{\|\vec{w}\|}$, i.e., we seek to minimize $\|\vec{w}\|$. More formally, we seek to minimize $\|\vec{w}\|$ in $(\vec{w}, b)$ subject to the constraint

$$y_i(\vec{w} \cdot \vec{x}_i - b) \geq 1$$

5. <u>The Primal Form</u>: Here we minimize $\frac{1}{2}\|\vec{w}\|^2$ subject to the constraint

$$y_i(\vec{w} \cdot \vec{x}_i - b) \geq 1$$

using the method of Lagrange multipliers. We compute

$$\underset{\vec{w}, b}{\arg min}\ \underset{\alpha \geq 0}{\arg max} \left\{ \frac{1}{2}\|\vec{w}\|^2 - \sum_{i=1}^{n} \alpha_i [y_i(\vec{w} \cdot \vec{x}_i - b) - 1] \right\}$$

which automatically implies that we ignore all contributions from

$$y_i(\vec{w} \cdot \vec{x}_i - b) > 0$$

as their corresponding

$$\alpha_i = 0$$

- Solution for the Primal Form => The stationary Karush-Kahn-Tucker condition implies that the solution can be expressed as a linear combination of the training vectors: $\sum_i \alpha_i y_i \vec{x}_i$. The corresponding $\vec{x}_i$'s are called the SUPPORT VECTORS, as they lie at the margin and satisfy

$$y_i(\vec{w} \cdot \vec{x}_i - b) \equiv 1$$

This allows one to define b as

$$b = \vec{w} \cdot \vec{x} - y$$

or more generally

$$b = \frac{1}{N_{SV}} \sum_{i=1}^{N_{SV}} (\vec{w} \cdot \vec{x}_i - y_i)$$

where N_{SV} is the number of the support vector samples.

6. The Dual Form: Writing the classification rule in its unconstrained dual form reveals that the maximum margin hyperplane (and therefore the classification task) is a function of only the support vectors, the set of training points that lie on the margin. Using

$$\vec{w} = \sum_i \alpha_i y_i \vec{x}_i$$

reduces the optimization task to

$$\arg\min_{\alpha_i} \left\{ \sum_i^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n \alpha_i y_i \alpha_j y_j \vec{x}_i \cdot \vec{x}_j \right\} = \arg\min_{\alpha_i} \left\{ \sum_i^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n \alpha_i y_i \alpha_j y_j \kappa(\vec{x}_i, \vec{x}_j) \right\}$$

subject to the constraint

$$\alpha_i \geq 0$$

The kernel, therefore, is

$$\kappa(\vec{x}_i, \vec{x}_j) = \vec{x}_i \cdot \vec{x}_j$$

a.

$$b = 0$$

is referred to as the biased SVM, and

$$b \neq 0$$

as the unbiased SVM.

7. SVM Soft Margin: If there exists no hyperplane such that the "yes" and the "no" categories cannot be split, the soft margin method will choose a hyperplane that separates the example set as cleanly as possible, while still maintaining the distances to the nearest cleanly split examples. Thus, this modified maximum margin allows for mislabeled examples (Cortes and Vapnik (1995)).
8. Soft Margin Slack Variable: The non-negative slack variable ξ_i measures the degree of miscalculation of the data $\vec{x}_i$ such that

$$y_i(\vec{w} \cdot \vec{x}_i - b) \geq 1 - \xi_i$$

for all

$$1 \leq i \leq n$$

The objective function now has an additional term that penalizes non-zero ξ_i, and the optimization is a trade-off between a large margin and a small penalty error.

9. Soft-Margin Optimization Setup: If the penalty function is linear, the optimization problem becomes

$$\underset{\vec{w},\vec{\xi},b}{\arg min}\left\{\frac{1}{2}\|\vec{w}\|^2 + C\sum_{i=1}^{n}\xi_i\right\}$$

subject to

$$y_i(\vec{w}\cdot\vec{x}_i - b) \geq 1 - \xi_i;\ \xi_i \geq 0$$

for all

$$1 \leq i \leq n$$

The corresponding Lagrangian objective function becomes

$$\underset{\vec{w},\vec{\xi},b}{\arg min}\ \underset{\vec{\alpha},\vec{\beta}}{\arg max}\left\{\frac{1}{2}\|\vec{w}\|^2 + C\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}\alpha_i[y_i(\vec{w}\cdot\vec{x}_i - b) - 1 + \xi_i]\right\};\ \alpha_i,\beta_i,\xi_i \geq 0$$

10. Soft Margin – Dual Form: This is similar to the non-soft margin formulation. Minimize

$$\underset{\alpha_i}{\arg min}\left\{\sum_{i}^{n}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{n}\alpha_i y_i \alpha_j y_j \kappa(\vec{x}_i,\vec{x}_j)\right\}$$

subject to the constraints

$$0 \leq \alpha_i \leq C$$

and

$$\sum_{i}^{n} \alpha_i y_i = 0$$

For linear separation, the kernel $\kappa(\vec{x}_i, \vec{x}_j)$ is still the dot product $\vec{x}_i \cdot \vec{x}_j$.

a. The key advantage of the linear penalty is that the slack variables vanish from the dual formulation, with constant C appearing purely as an additional constraint on the Lagrange multipliers. Nonlinear penalty functions have been used particularly to reduce the effect of outliers on the classifiers, but unless care is taken this problem can become non-convex, thus making it difficult to find a global solution.

11. Nonlinear SVM Classification: Boser, Guyon, and Vapnik (1992) extend the kernel trick originally proposed by Aizerman, Braverman, Emmanuel, and Rozonoer (1964) to create nonlinear classifiers – the formulation is more or less identical to that of the linear case, with the key difference being that the kernel basis function is not $\vec{x}_i \cdot \vec{x}_j$ anymore – it becomes a nonlinear kernel function. This allows the algorithm to fit the maximum margin hyperplane in the transformed feature space.
    a. The Transformed Feature Space => The transformation is nonlinear, and the transformed space is high-dimensional. Thus, though the classifier is a hyperplane in the high-dimensional feature space, it may be nonlinear in the original input space.
12. The Nonlinear Kernel Basis Function:
    a. Properties => In all the cases (including the linear basis), the kernel basis function is symmetric in $\vec{x}_i$ and $\vec{x}_j$, and in all cases except the Gaussian radial basis, it only depends on the $\vec{x}_i \cdot \vec{x}_j$ dot product (in the case of Gaussian radial basis function, there is additional dependence on the self-dot-product, or the Euclidean/Mahalanobis distance of the given feature vector).
    b. Usage => More generally, the kernel basis function is related to the feature vector transform $\varphi(\vec{x}_i)$ as

$$\kappa(\vec{x}_i, \vec{x}_j) = \varphi(\vec{x}_i) \cdot \varphi(\vec{x}_j)$$

$\vec{w}$ is evaluated in the transformed feature vector space as

$$\vec{w} = \sum_i \alpha_i y_i \varphi(\vec{x}_i)$$

Dot products involving $\vec{w}$ for classification can be computed again using the kernel trick, i.e.,

$$\vec{w} \cdot \varphi(\vec{x}) = \sum_i \alpha_i y_i \kappa(\vec{x}_i, \vec{x})$$

However, in general, there does not exist a $\vec{w}'$ such that

$$\vec{w}' \cdot \varphi(\vec{x}) = \kappa(\vec{w}', \vec{x})$$

(this straight scaling, however, is valid in the linear case).

c. Kernel Basis Function: Gaussian Radial Basis => If the kernel used is a Gaussian radial basis function, the corresponding feature space is a Hilbert-space of infinite dimensions. Maximum margin classifiers are well-regularized, so infinite feature vector dimensions do not spoil the results. The kernel Gaussian radial basis function is

$$\kappa(\vec{x}_i, \vec{x}_j) = e^{-\gamma \|\vec{x}_i - \vec{x}_j\|^2}$$

for

$$\gamma > 0$$

Of course, γ is also expressed often as

$$\gamma = \frac{1}{2\sigma^2}$$

d. Kernel Basis Function: Homogenous Polynomial Basis =>

$$\kappa(\vec{x}_i, \vec{x}_j) = [\vec{x}_i \cdot \vec{x}_j]^d$$

e. Kernel Basis Function: Inhomogenous Polynomial Basis =>

$$\kappa(\vec{x}_i, \vec{x}_j) = [1 + \vec{x}_i \cdot \vec{x}_j]^d$$

f. Kernel Basis Function: Hyperbolic Tangent Radial Basis =>

$$\kappa(\vec{x}_i, \vec{x}_j) = \tanh(C + k\vec{x}_i \cdot \vec{x}_j)$$

for certain (not all)

$$C < 0; k > 0$$

13. SVM vs. Other Classifiers: SVM belongs to the family of generalized linear classifiers. They correspond to a special case of Tikhonov regularization that simultaneously minimized the empirical classification error and maximizes the geometric margin, and are therefore referred to maximum margin classifiers (Meyer, Leisch, and Hornik (2003)).
14. SVM Parameter Selection:
    a. Effectiveness of the SVM depends on the kernel basis function choice, the kernel parameters, and the soft margin parameter C. For instance, for the Gaussian kernel with parameter γ, the best combination of γ and C may be selected by a grid search of exponentially growing sequences of C/γ, i.e.,

$$C \in \{2^{-5}, 2^{-3}, \cdots, 2^{13}, 2^{15}\}$$

and

$$\gamma \in \{2^{-15}, 2^{-13}, \cdots, 2^{1}, 2^{3}\}$$

b. Further, each combination of the parameter choices is checked using cross validation, and the parameters with the cross-validation accuracy are picked. The final model, which is used for testing and classifying new data, is then trained on the entire training set using the selected parameters (Hsu, Chang, and Lin (2003)).

15. Drawbacks of the SVM Approach:
    a. Uncalibrated Class Membership Probabilities
    b. As is, it is directly applicable only to two-class classification problems. Additional algorithms to reduce the multi-class classification may need to be applied.
    c. Parameters of the solved model are hard to interpret.
16. Multiclass SVM: As outlined in Hsu and Lin (2002) and Duan and Keerthi (2005), the choices are:
    a. One vs. All => This is a winner take all strategy where the classifier with the highest score wins (for this, it is important the output functions be calibrated to produce comparable scores).
    b. One vs. One => Here, the classification is done by a max wins voting strategy in which every classifier assigns the instance to one of two classes, after which the assigned class is increased by one vote, and finally the class with most votes wins.
17. Other Multiclass Extensions:
    a. Directed Acyclic Graph SVM (Platt, Christianini, and Shawe-Taylor (2000)).
    b. Error correcting output codes (Dietterich and Bakiri (1995)).
    c. Crammer and Singer (2001) propose a multiclass SVM which casts the classification problem into a single optimization problem rather than decomposing it into multiple binary classification problems (Lee, Lin, and Wahba (2001), and Lee, Lin, and Wahba (2004)).
18. Transduction SVM: These extend the traditional SVM treatment so that they could handle labeled data in semi-supervised learning by following the principles of transduction. In

effect, the formulation extends the training data $\{\vec{D}\}$ with the unlabeled set $\{\vec{D}'\}$ and optimizes the calibration across both sets (Joachims (1999)).

a. Setup => The following primal optimization problem statement sets it up: Minimize $\frac{1}{2}\|\vec{w}\|^2$ in $(\vec{w}, b, \vec{y}^*)$ subject to the constraints

$$y_i(\vec{w} \cdot \vec{x}_i - b) \geq 1;\ \forall i$$

and

$$y_j{}^*\left(\vec{w} \cdot \vec{x}_j - b\right) \geq 1;\ \forall j$$

with

$$y_j{}^* \in [-1, +1]$$

19. SVM Regression (SVR): For the regression extension SVM, the model produced by SVR depends only on the subset of the training data (just like SVM classification), as the cost function for building the model ignores any training data close to the model prediction within a threshold $\mathcal{E}$ (Drucker, Burges, Kaufman, Smola, and Vapnik (1997)). Suykens and Vandewalle (1999) lay out a least-squares version of SVR.
20. Maximum Margin Hyperplane Solutions:
    a. There exist several specialized algorithms for quickly solving the QP problem that arises from the SVM formulation, mostly relying on heuristics for breaking the problem down into smaller, more manageable chunks. A common method is Platt's Sequential Minimal Optimizer (SMO) which breaks the problem down into 2D sub-problems that may be solved analytically, eliminating the need for a numerical optimization algorithm.
    b. Another approach is to use an interior-point method that employs Newton-like iterations to find a solution to the Karush-Kahn-Tucker conditions of the primal and the dual problems (Ferris and Munson (2002)). Instead of solving a sequence of

broken-down problems, this approach directly solves the problems as a whole. To avoid solving a linear system involving a large kernel matrix, a low rank approximation to the matrix is often used in the kernel trick.

References

- Aizerman, M. A., E. M. Braverman, and L. I. Rozonoer (1964): Theoretical Foundations of the Potential Function Method in Pattern Recognition Learning *Automation and Remote Control* **25** 821-837.
- Boser, B. E. I. M. Guyon, and V. N. Vapnik (1992): A Training Algorithm for Optimal Margin Classifiers *5th Annual ACM Workshop on COLT* **ACM Press** Pittsburgh, PA.
- Cortes, C., and V. N. Vapnik (1995): Support Vector Networks *Machine Learning* **20** 273-297.
- Crammer, K., and Y. Singer (2001): On the Algorithmic Implementation of Multiclass Kernel Based Vector Machines *Journal of Machine Learning Research* **2** 265-292.
- Dietterich, T. G. and G. B. Bakiri (1995): Solving Multiclass Learning Problems via Error-Correcting Output Codes *Journal of Artificial Intelligence Research* **2 (2)** 263-286.
- Drucker, H., C. J. C. Burges, L. Kaufman, A. J. Smola, and V. N. Vapnik (1997): Support Vector Regression Machines, in: *Advances in Neural Information Processing Systems 9 (NIPS 1996)* **MIT Press**.
- Duan, K. B. and S. S. Keerthi (2005): Which is the Best Multi-class SVM? An Empirical Study *Proceedings of the 6th International Workshop on Multiple Classifier Systems* **3541** 278.
- Ferris, M. C., and T. S. Munson (2002): Interior-Point Methods for Massive Support Vector Machines *SIAM Journal on Optimization* **13 (3)** 783-804.
- Hsu, C. W. and C. J. Lin (2002): A Comparison of Methods for Multiclass Support Vector Machines *IEEE Transactions on Neural Networks*.
- Hsu, C. W., C. C. Chang, and C. J. Lin (2003): A Practical Guide to Support Vector Classification

- Meyer, D., F. Leisch, and K. Hornik (2003): The Support Vector Machine Under Test *Neurocomputing* **55 (1-2)** 169-186.
- Joachims, T. (1999): Transductive Inference for Text Classification using Support Vector Machines *Proceedings of the 1999 International Conference on Machine Learning (ICML 1999)* 200-209.
- Lee, Y., Y. Lin, and G. Wahba (2001): Multi-Category Support Vector Machines *Computing Science and Statistics* **33**.
- Lee, Y., Y. Lin, and G. Wahba (2004): Multi-Category Support Vector Machines – Theory and Application to the Classification of Microarray Data and Satellite Radiance Data *Journal of the American Statistical Association* **99 (465)** 67-81.
- Platt, J., N. Christianini, and J. Shawe-Taylor (2000): Large Margin DAGs for Multiclass Classification *Advances in Neural Information Processing Systems* **MIT Press**.
- Press, A., S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery (2007): *Numerical Recipes – The Art of Scientific Computing 3rd Edition* **Cambridge University Press**, New York.
- Support Vector Machine (Wiki): Wikipedia Entry for Support Vector Machine.
- Suykens, J. A. K., and J. P. L. Vandewalle (1999): Least Squares Support Vector Machine Classifiers *Neural Processing Letters* **9 (3)** 293-300.

Gene Expression Programming (GEP)

1. Definition: GEP is an evolutionary algorithm that creates computer programs and models. The programs are complex tree structures that learn and adapt by changing their sizes, shapes, and composition.
2. GEP as a Genotype-Phenotype Algorithm: The GEP programs are encoded in simple linear chromosomes of fixed length; thus, just like a genotype-phenotype system, GEP benefits from a simple genome to keep and transmit genetic information, and a complex phenotype to explore the environment and adapt to it.
3. Evolutionary Algorithms: These use populations of individuals, select individuals according to fitness, introduce genetic variation one/more genetic operators, and have been used in optimization problems since the 1950's (Box (1957)), Friedman (1959), Rechenberg (1973), and Mitchell (1996)).
4. GEP as a Member of the Family of Evolutionary Algorithms: GEP is closely related to genetic algorithms and genetic programming (Ferreira (2001)). From genetic algorithms GEP inherits the notion of linear chromosomes of fixed length; and from genetic programming it inherits the expressive parse trees of various sizes and shapes.
5. Multigenic Genotype/Phenotype: The linear chromosomes work as genotypes, and the parse trees as phenotypes, thereby creating a genotype/phenotype system that is multigenic, thus encoding multiple parsed trees in each chromosome. This indicates the programs created by GEPs are composed of multiple-parse trees. Further, since these parse-trees themselves are the result of gene expression, they are called as expression trees.
6. The Encoding – The Genotype: Genotypes consist of one/more fixed length genes, which encode expression trees of different sizes and shapes. The expressions contain encoding for both the operators $(\times, \div, +, -, \cdots)$ and the variables/constants (e.g., $L + a - baccd \cdots$). They may also possess an implied execution pipeline order (BODMAS etc.)
7. Expression Trees - The Phenotype: These interpret the genome specified above and expand them out into expression trees. Thus, to move from the genotype (the coding) to the

phenotype (the expression), you need a pre-agreed dictionary, and possibly an assembly rule, since the terminals assembly is non-unique.

   a. The genome k-expressions above are also referred to as k-expressions (or Karva expressions).

8. k-Expressions: The k-expressions of GEP correspond to regions of genes that WILL get expressed – there may be sequences of the genes that may not be expressed at all, and the reason for that is to provide a buffer of terminals so that all k-expressions encoded in GEP genes always correspond to valid programs/expressions.
9. Gene Head/Tail: The gene head is used to encode the functions and variables needed to solve the problem at hand, whereas the tail is used to a) encode variables, and b) provide a reservoir of terminals that ensures that the fixed length gene is error free.
   a. GEP Genes Tail Length =>

$$t = h(n_{MAX} - 1) + 1$$

   where h is the length of the head, and n_{MAX} is its maximum arity.

10. Multigenic Chromosomes: Each gene is composed of one/more genes, the output of each of which can be combined in some way. Each output is referred to as a sub-ET (sub expression tree).
11. Gene Output Linkages: Although the output will have to be a primitive, there are no restrictions on them, thus they may be linked using any combination of primitives (mean, median, average etc.) in a number of ways. They can also be evolved, of chosen in an ad hoc manner (Ferreira (2002a), Ferreira (2006a)).
12. Homeotic Genes: These control the interactions of the different sub-ET's, i.e., determine which sub-ET's are called upon and in which order, in which cell (the driver program), as well as the nature of connections the ET's establish with each other.
   a. Homeotic Genes and Cellular Systems => The homeotic genes are organized identically to that of the other genes, except their heads contain linking functions and a special kind of terminal – the genic terminal – that represents normal genes. Expressions of the normal genes result in different sub-ET's (also referred to as ADF's – automatically defined functions), with different evolutionary linkages, etc.

13. Multicellular Programs: These are composed of more than one homeotic gene, and the multigenic output of each of homoeotic gene results in different sub-ET's and their linkages, each of which may be custom evolved.
14. Additional Gene Domains: In addition to the typical head/tail domains, you may have additional domains used for maintaining/learning/calibrating/process tuning the constants used in finding a solution.
15. Examples of the Numerical Constants Employed in Domain Usage:
    a. As weights/factors in a function approximation problem (the GEP-RNC algorithm)
    b. As weights/thresholds of a neural network (GEP-NN)
    c. Numerical Constants in a Decision Tree (GEP-DT algorithm)
    d. Weights used in a Polynomial Induction
    e. Random numerical constants to discover parameter values in a parameter optimization task
16. Basis Gene Expression Algorithm:
    a. Select Function Set
    b. Select Terminal Set
    c. Load Dataset for Fitness Evaluation
    d. Create Chromosomes of initial Population randomly
    e. For each Program in the Population:
        i. Express the Chromosome
        ii. Execute the Program
        iii. Evaluate the Fitness
    f. Verify the Termination Condition
    g. Select the Programs
    h. Replicate the Selected Programs to form the next Population
    i. Modify Chromosomes using Genetic Operators
    j. Go back to Step e.
17. Preparation for the Execution: Steps a through e are just the preparatory steps needed for e through i. The key step of initial population occurs using random elements of function/terminal sets.

18. Population of Programs: Like other evolutionary programs, GEP works with individual populations (i.e., populations of computer programs – the organism is treated as a program!) From the initial population, descendants are evolved via selection or genetic modification. In the genotype/phenotype system of GEP, only the simple linear chromosomes of individual programs need to be evolved, as this localizes the structural soundness and eventually syntactically correct programs.
19. Fitness Functions and the Selection Environment: Selection environments are akin to training sets, and the fitness functions determine the penalty cost. The fitness of a program depends on both the cost function and the training data.
20. Selection Environment: This contains the training data/records as well as the fitness cases, the selection environment should represent the problem cases well, as well being well-balanced, should not be too large, but large enough to enable good generalization and validation.
21. Fitness Functions for Regression Analysis (Continuous Inference): Due to the continuous nature, direct comparison metrics are possible. Any of the appropriate Bayes' loss functions would be a good candidate for fine granularity/smoothness to the solution space.
    a. Multi-target Regression Fitness Function => While fitness functions based purely on R^2 and correlation β are smooth, they work best as combinations of multiple metrics (i.e., ones that control coarseness of fit, say less than 10% of estimated samples out of the range, quality of approximation/shape preservation, etc.)
22. Fitness Functions for Classification:
    a. Fitness Functions Based on Confusion Matrix => Confusion matrix along with the appropriate smoothener creates efficient/effective fitness functions. Examples of smootheners include F-measures, Jaccard similarity, Matthews correlation coefficient, and cost/gain matrix that combines the costs/gains assigned to the 4 different confusion classifications.
    b. Fitness Functions based on the Solution Space => These fitness functions explore the structure of the classification model itself (which includes the domain, the range, the distribution of the model output, and the classifier margin). For instance, one can combine:

i. Measures based on confusion matrix with those based on the mean squared error between the rate model outputs and the actual values
ii. F-measure and the R^2 for the model and the target
iii. Cost/gain Matrix with the correlation coefficient
iv. Functions that expose model granularity with metrics based on the area under the ROC curve and the rank measure

23. Fitness Functions for Logistic Regression: Here the confusion matrix is combined with the joint metric that uses model probabilities and measures probabilities.
24. Fitness Functions for Boolean Problems: Here the cost/gain confusion matrix in conjunction with the deterministic/hard hit rates is the only real option
25. Selection and Elitism: Roulette-wheel selection according to fitness and the luck of the draw is the most popular. Combining roulette-selection with cloning the best program of each generation guarantees that the very best traits are not lost (this is called simple elitism).
26. Reproduction with Modification: Program reproduction first involves selection, then replication of the genome sequence. Genome modification is not required for reproduction, but without it adaptation and evolution won't take place.
27. Selection for Replication: Selection operator (say, a copy instruction) selects programs for the replication operator to copy (selection typically occurs via simple elitism). For evolution to occur, replication is implemented with a few random errors thrown in through genetic operators such as mutation, recombination, transposition, inversion, and many others.
28. Mutation: Mutation is the most important genetic operator (Ferreira (2002b)), and changes genomes by changing one element in it by another. Accumulation of many small changes over time creates great diversity.
    a. Unconstrained Mutation => In GEP, mutation is totally unconstrained, which means that in each gene domain, any domain symbol can be replaced by another. For example, in the heads of genes, any function can be replaced by a terminal or another function (regardless of the number of arguments in the new function), and a terminal may be replaced by a function or a new terminal.
29. Recombination: Recombination typically involves two parent chromosomes to create two new chromosomes by combining different parts from the parent chromosomes. As long as the parent chromosomes are aligned and the exchanged fragments are homologous (i.e., they

occupy the same position in the chromosomes), the new chromosomes created by recombination will always encode syntactically correct programs.

   a. Recombination crossover implementation types =>
      i. Changing the number/combination of parents
      ii. Changing the split points
      iii. Changing the order of the fragments to exchange

30. Transposition: This involves introduction of an insertion sequence into the chromosome. The insertion sequence may be chosen from anywhere in the chromosome, but is inserted only at the head. This ensures that all domains, including the tail domain sequences, result in error-free programs.
    a. Transposition Implementation Methods => Transposition needs to preserve the structure and the length, so it needs to be implemented in the following ways. Both methods can be implemented to operate within chromosomes, or within even a single gene.
       i. Create a shift at the insertion site, followed by a delete at the end of the head
       ii. Overwrite the local sequence at the target site (thus, making it easier to implement)
31. Inversion: Inversion inverts a small sequence within a chromosome, and is especially powerful for combinatorial optimization (Ferreira (2002c)). In GEP it can be easily implemented in all gene domains, and the produced off springs are always syntactically correct. For any gene domain, a sequence (ranging from at least two elements to as big as the domain itself) is chosen at random within that domain, and is then inverted.
32. Other Genetic Operators: The possibilities are endless – examples include one-point/two-point/gene/uniform recombinations, gene/root/domain-specific transposition, domain-specific mutation, etc. All are easily implemented and widely used.
33. GEP-RNC Algorithm: Here the GEP's use an extra domain for accommodating the random numerical constants (RNC) – the DC domain. Special DC-specific genetic operators allow for efficient calculation of the RNC's among the individual programs. Also, a special mutation operator allows for the permanent introduction of variation onto the RNC set.
34. GEP-NN:

a. Motivation => Typical neural networks consists of three different classes of units – the input units, the hidden units, and the output units. An activation pattern presented at the input spreads forward through the hidden layers to the output. The activation input is amplified by the link-specific weight set, and is then thresholded/transmitted through the link set to the output.
b. Parameters => The parameters are essentially the amplification weights and the corresponding thresholds, each of which may be populated in a GEP algorithm from a random initial population and then evolved.
c. The Algorithm => Here the network architecture is encoded in the head/tail domain (Ferreira (2006b)). The head contains special encodings (or functions in the GEP-NN terminology) that activate the hidden/output units, and terminals that represent the input units. The tail contains only of the terminal units.
d. Encoding the weights/thresholds => Besides the head/tail domains, GEP-NN uses two extra domains - D_W and D_t - the encode the weights/thresholds of the NN. D_W follows the tail with length

$$d_W = hn_{MAX}$$

and D_t follows D_W with length

$$d_t = t$$

35. GEP-DT:

a. Structure and Motivation => Decision Trees (DT) have three types of nodes – the root nodes, the internal nodes, and the leaf/terminal nodes. Roots/internal decision nodes represent the test conditions for the different attributes/variables in the dataset. Leaves specify the class labels that terminate across the tree paths (Ferreira (2006c)).
b. GEP-DT Algorithm => Select an attribute for the root node (starting with random initial population and an eventual evolution) and use the nominal/numeric DT genesis algorithm to drive the classification label output.

c. DT Encoding => Rules for encoding/evolving the attributes are no different than that of GEP-RNC domain localization etc. in that the extra domain encodes the numerical constants and input labels corresponding to the attributes.

36. Performance: As is evident, beyond the encoding representation and the pipeline evolution strategies, there are no significant performance (as you would think) with GEP, and this is borne out by tests on GEP-DT (Oltean and Grosan (2003)).

References

- Box, G. E. P. (1957): Evolutionary Operation: A Method for Increasing Industrial Productivity *Applied Statistics* **6** 81-101.
- Ferreira, C. (2001): Gene Expression Programming: A New Adaptive Algorithm for Solving Problems *Complex Systems* **13 (2)** 87-129.
- Ferreira, C. (2002a): *Gene Expression Programming: Mathematical Modeling by an Artificial Intelligence* **Angra do Heroismo** Portugal.
- Ferreira, C. (2002b): Mutation, Transposition, and Recombination: An Analysis of the Evolutionary Dynamics *Proceedings of the 6th Joint Conference on Information Sciences, 4th International Workshop on Frontiers in Evolutionary Algorithms* Research Triangle Park, NC.
- Ferreira, C. (2002c): Combinatorial Optimization by Gene Expression Programming: Inversion Revisited *Proceedings of the Argentine Symposium on Artificial Intelligence* Santa Fe, Argentina.
- Ferreira, C. (2006a): Automatically Defined Functions in Gene Expression Programming, in: *Genetic Systems Programming: Theory and Experiences, Studies in Computational Intelligence* (**13) Springer Verlag**.
- Ferreira, C. (2006b): Designing Neural Networks Using Gene Expression Programming, in: *Applied Soft Computing Technologies: The Challenge of Complexity* **Springer Verlag**.
- Ferreira, C. (2006c): *Gene Expression Programming: Mathematical Modeling by an Artificial Intelligence* **Springer Verlag**.

- Friedman, G. J. (1959): Digital Simulation of an Evolutionary Process *General Systems Yearbook* **4** 171-184.
- Mitchell, M. (1996): *An Introduction to Genetic Algorithms* **MIT Press** Cambridge MA.
- Oltean, M., and C. Grosan (2003): A Comparison of Several Genetic Programming Techniques *Complex Systems* **14 (4)** 285-314.
- Rechenberg, I. (1973): *Evolutionsstrategie* **Holzman-Froboog** Stuttgart.

Cluster Analysis

Introduction

1. Definition: Also called Clustering, it is the task of grouping a set of objects in such a way such that objects in the same group (cluster) are more similar to each other than to another in a different cluster.
2. Alternate Terms: Automatic Classification, Numerical Taxonomy, Botryology, and typological analysis (Tryon (1939), Bailey (1994)).
3. Cross Field Focus Differences: Differences in above terminology/focus is due to the eventual usage of the clustering results; in data mining, the resulting groups are the matter of interest, while in automatic classification, the resulting discriminative power is of interest (Cattell (1993)).

Cluster Models

1. Cluster Model/Conception: "Cluster Model" refers to the conceptual entity that defines a cluster unit (Estivill-Castro (2002), Cluster Analysis (Wiki)).
2. Types of Cluster Models: Following examples are given for the corresponding cluster model:
    a. Connectivity Models => Hierarchical Clustering builds models based on distance Connectivity.
    b. Centroid Models => *k*-Means represents each cluster by its mean vector.
    c. Distribution Models => Clusters are modeled using statistical distributions, e.g., multi-variate normal distributions used by the Expectation Maximization Algorithm.
    d. Density Models => DBSCAN and OPTICS define clusters as connected dense regions in the data space.

e. Sub-space Models (also referred to as bi-clustering, co-clustering, or 2 mode clustering) => Here the clusters are modeled using both the cluster member characteristics and their relevant attributes.
f. Group Models => These models do not provide a refined mathematical model for the clustering – instead just provide the grouping information.
g. Graph-Based Models => These represent a clique, i.e., a sub-set of nodes in the graph such that every 2 nodes in the subset are connected by an edge to form a proto-typical cluster. Relaxations of the complete connectivity requirement (e.g., it may be stipulated that a fraction of the edges may be missing) are known as quasi-cliques.

3. Hard vs. Soft Clusters: In hard clustering, each object belongs to at most a single cluster. In soft clustering (also called fuzzy clustering), there exists a probability of a object belonging to a cluster.
4. Cluster Categorization Based on the Strength of Membership:
    a. Strict Partitioning Cluster => Here each object belongs to precisely one cluster.
    b. Strict Partitioning Cluster with Outliers => Objects can also belong to no clusters at all, and are considered outliers.
    c. Overlapping Clustering (also called alternative clustering or multi-view clustering) => While usually a hard cluster, objects may belong to more than one cluster.
    d. Hierarchical Clustering => Objects that belong to a child cluster also belong to a parent cluster.
    e. Sub-space Clustering => While typically an overlapping cluster, inside of a uniquely defined sub-space, clusters are not expected to overlap.

Connectivity Based Clustering

1. Definition: Also referred to hierarchical clustering, connectivity-based clustering is based on the idea that “closer” objects form a cluster. Thus, these algorithms form clusters by connecting objects inside of a distance metric.

2. Specification of the Connectivity Cluster: This cluster is described largely by the distance required to connect different parts of the cluster. Different cluster form at different max distance thresholds.
3. Connectivity Clustering – Linkage Criterion: In addition to the choice of the distance functions, the linkage criterion (an additional metric that links the elements of the cluster) needs to be specified. Popular choices are single-linkage clustering (here the linkage criterion is the minimum of the object distances), complete linkage clustering (the linkage criterion is the maximum of the object distances), and UPGMA (Unweighted Pair Group Method with Arithmetic Mean, also known as average linkage clustering).
4. Hierarchical Cluster Formation Types: They are either agglomerative (starting with single elements and aggregating them into clusters) or divisive (starting with the complete data set and dividing them into partitions).
5. Hierarchical Clustering Robustness and Uniqueness: Hierarchical clustering does not result in unique clusters – it depends on the starting point, and only produces a hierarchy from which additional decisions need to be made to fix down the clusters. Further, it is not robust to outliers, which results in additional clusters, or causes clusters to merge (this is called chaining phenomenon, esp. in single linkage clustering).
6. Hierarchical Clustering Performance: In general, the complexity associated is $O(n^3)$, thus making it too slow to be practical. Specialized situations produce $O(n^2)$ – e.g., for SLINK that uses single-linkage clustering (Sibson (1973)), or for CLINK that uses complete linkage clustering (DeFays (1977)).

Centroid Based Clustering

1. Definition: Here the clusters are represented by a central vector, which in itself may not be a member of any data set. When the number of clusters is fixed to κ, κ-means clustering provides a formal definition as an optimization problem – find the κ clusters and assign the objects to the nearest center such that the distances (Euclidean/squared) are minimized.

2. κ-Means Optimization Problem: This is NP-hard, and only approximate solutions that uncover local optimum are used. The most well-known of these is the Lloyd's algorithm (Lloyd (1982)), and is often run multiple times with different random initializations.
3. κ-Means Algorithm: Lloyd's algorithm is also referred to as the κ-Means algorithm. Variations on it include choosing the best of multiple runs, restricting the centroids to members of the data sets (κ-Medoids), choosing medians instead (κ-Means++), or allowing a fuzzy cluster assignment (fuzzy c-means). All of these variations may be applied concurrently.
4. Drawback of the κ-Means: Requires κ to be extraneously specified, often in advance. Further, this has a preference to generate clusters of similar size, as they always assign the object to the nearest centroid. Since κ-Means optimizes for the cluster centers (and not the cluster borders), it can produce incorrect borders.
5. Theoretical properties of κ-Means: Partitions data space into Voronoi polyhedral, and is conceptually close to nearest neighbor classification.

Distribution Based Clustering

1. Definition: Clusters are defined statistically: the representation here suits objects most likely to probabilistically belong to a specified distribution. Since a more complex distribution will explain/fit the data better, overfitting can become a challenge in this methodology.
2. Gaussian Mixture Cluster Models: Here the data set is modeled with a fixed number of Gaussian distributions to avoid over-fitting. These distributions are initialized randomly, and the parameters are iteratively optimized via the expectation maximization algorithm. Convergence is to a local optimum. For hard clustering, objects are assigned to the Gaussian distribution they most likely belong to (this is not needed for soft clustering).

Density Based Clustering

1. Definition: Here the clusters are defined as areas of higher density than the remainder of the data set (Kriegel, Kroger, Sander, and Zimek (2011)). Objects in the sparse areas (these are required to separate the clusters) are considered to be noise and border points.
2. DBSCAN: This is the most popular density-based clustering method (Ester, Kriegel, Sander, and Xu (1996)). Similar to linkage-based clustering, it is based on connecting points within certain distance thresholds. However, it connects only points that satisfy a density criterion (defined in the original variant as the number of objects within a specified radius). A cluster consists of all density-connected objects (which can form an arbitrary shape, in contrast to many other methods) PLUS all the objects within the original objects' range.
3. Properties of DBSCAN: Low complexity, requires a linear number of range queries, and it discovers results essentially in each run so that it doesn't need to be run multiple times (it is deterministic in the core and the noise points, but not at the border points).
4. Enhancement to DBSCAN - OPTICS: OPTICS (Ankerst, Breuning, Kriegel, and Sander (1999)) is a generalization of DBSCAN that removes the need for choosing an appropriate value for ε, and produces a hierarchical cluster result related to linkage clustering.
5. Enhancement of DBSCAN - DeLiClu: Density Link Clustering (DeLiClu – Achtert, Bohm, and Kroger (2006)) combines ideas from single-linkage clustering and OPTICS, thereby eliminating the ε parameter entirely, and offering performance improvements over OPTICS by using an R-tree index.
6. Drawback of DBSCAN - OPTICS: These need density-drop to be able to detect the clusters, and end up performing poorly when the data is probabilistic (e.g., when the data uses Gaussian mixtures). A variant of DBSCAN called EnDBSCAN has been developed to detect intrinsic cluster structures prevalent in the majority of real-life data (Roy and Bhattacharyya (2005)).

Recent Clustering Enhancements

1. Performance Enhancements: Check out CLARANS (Ng and Han (1994)), BIRCH (Zhang, Ramakrishnan, and Livny (1996)) and Sculley (2010).

2. Clustering on Large Data Sets: With large data, there has been a willingness to trade semantic meaning of the generated clusters for performance. This has led to an array of pre-clustering techniques such as canopy clustering, which can process huge data sets efficiently. The resulting clusters are merely a pre-partitioning of the data that need to be analyzed using slower methods such as κ-Means (e.g., seed-based clustering (Can and Ozkarahan (1990))).
3. High-Dimensional Clustering: Curse of dimensionality renders particular distance functions problematic. Therefore high-dimension clustering employs sub-space clustering, correlation clustering etc. that search for arbitrary rotated features. Sample algorithms are given in CLIQUE (Agrawal, Gehrke, Gunopoulus, and Raghavan (2003)) and SUBCLU (Kaling, Kriegel, and Kroger (2004)).
4. Density Subspace Clustering: A combination of the above techniques has been adopted in hierarchical subspace clustering. Combination of subspace clustering and DBSCAN/OPTICS has been used in HiSC (Achtert, Bohm, Kriegel, Kroger, Muller-Gorman, and Zimek (2006)) and DiSH (Achtert, Bohm, Kriegel, Kroger, Muller-Gorman, and Zimek (2007). Correlation clustering variants of the above include HiCO (Achtert, Bohm, Kroger, and Zimek (2006). Other enhancements include correlation connectivity enhancements in 4C (Bohm, Kaling, Kroger, and Zimek (2004)) and exploration of hierarchical density-based correlation clusters (see ERiC – Achtert, Bohm, Kriegel, Kroger, and Zimek (2007)).
5. Enhancements using Mutual Information: Examples include variation of information metric (Meila (2003)) with applications to hierarchical clustering (Kraskov, Stogbauer, Andrzejak, and Grassberger (2003)). Genetic information enables testing a wide variety of mutual information based fit functions (Auffarth (2010)). Finally, message-passing algorithms based on statistical physics has opened up a variety of clustering techniques (Frey and Dueck (2007)).

Internal Cluster Evaluation

1. Internal Cluster Evaluation - Objective: This carries out an evaluation of the clustering result based on the data set that was clustered in itself. Drawbacks with internal cluster evaluations are: a) High scores on internal metrics do not necessarily result in effective information

retrieval approaches (Manning, Raghavan, and Schutze (2008)). Further, if the evaluation metric is similar to the clustering metric (e.g., say if both are distance based), this evaluation over-estimates the impact of the clustering algorithm.

2. Davies Bouldin Index: This is estimated using the formula

$$DB = \frac{1}{n}\sum_{i=1}^{n}\left\{\underset{i \neq j}{max}\left(\frac{\sigma_i + \sigma_j}{d(c_i, c_j)}\right)\right\}$$

where n is the number of clusters, c_i is the centroid of the cluster i, c_j is the centroid of the cluster j, σ_i is the average distance of all the elements in cluster i to its centroid c_i, and $d(c_i, c_j)$ is the distance between the centroids c_i and c_j. Algorithms that produce clusters with low intra-cluster distances (high intra-cluster similarity) and high inter-cluster distances (low inter-cluster similarity) have low index values. Thus, algorithms that produce the smallest index value are the most effective ones by this metric (it is important to remember that the notion of "distance" needs to match between the clustering and the evaluation schemes – beyond that it is left open).

3. Dunn Index (Dunn (1974)): The Dunn index aims to identify dense and well-separated clusters, and is defined as the ratio between the minimal inter-cluster distance and the maximal intra-cluster distance:

$$D = \underset{1 \leq i \leq n}{min}\left\{\underset{1 \leq j \leq n; i \neq j}{min}\left[\frac{d(i,j)}{max_{1 \leq k \leq n} d'(k)}\right]\right\}$$

where $d(i,j)$ is the distance between clusters i and j and $d'(k)$ is the measure of the intra-cluster distance in cluster k. Inter-cluster distance $d(i,j)$ may be measured in a number of ways – such as the distance between the centroids. Likewise, $d'(k)$ could be, say, the maximum distance between any pair of elements in cluster k. Thus, algorithms with a higher Dunn index value are more desirable.

External Cluster Evaluation

1. External Cluster Evaluation - Objective: This is more or less identical to cross validation or GCV, and the comparison is with hand labeled "cross validation" data set. Challenges with this approach include the choice of the cross-validation sample (to reveal sub-structures embedded in the data set) (Farber, Gunnermann, Kriegel, Kroger, Muller, Schubert, Seidl, and Zimek (2010)).
2. Rand Measure (Rand (1971)): This measures how similar the cluster results are to the benchmark classifications. It is the ratio of the correct decisions made by the algorithm to the total results (this comes from using a confusion matrix):

$$RI = \frac{TP + TN}{TP + TN + FP + FN}$$

   One challenge with this approach is that its weight FP and FN equally – the F-measure removes that.
3. F-Measure: Defining Precision Rate and the recall rate as

$$P = \frac{TP}{TP + FP}$$

   and

$$R = \frac{TP}{TP + FN}$$

   respectively, the F-measure weights FN through a

$$\beta \geq 0$$

   parameter as

$$F_\beta = \frac{(\beta^2 + 1).P.R}{\beta^2.P.R}$$

When $\beta \to 0$, $F_0 \to P$, thus recovering the Rand index.

4. Pair Counted F-Measure: This is the F-Measure applied to the set of object pairs, where the objects are paired with each other when they are part of the same cluster. This measure is able to compare clusterings with different numbers of clusters.
5. Fowlkes-Mallows Index (Fowlkes and Mallows (1983)): FM computes the similarity between clusters returned by the clustering algorithm and the benchmark classifications. Higher the FM, greater the similarity between the result cluster and the benchmark cluster:

$$FM = \sqrt{\frac{TP}{TP + FP} \cdot \frac{TP}{TP + FN}}$$

In effect, FM is the geometric mean of the P and the R rates, while the F-measure may be viewed as their harmonic mean (Hubert and Arabie (1985)). P and R are also referred to as Wallace's B^I and B^{II} indices (Wallace (1983)).

6. Jaccard Index: This is simply the number of unique elements common to the two sets (the cluster set and the benchmark set) divided by the total number of unique elements in both sets:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{TP}{TP + FP + FN}$$

Thus, $J(A, B)$ ranges from 0 (no common elements) to 1 (the sets are identical).

7. Mutual Information Based External Evaluation: This is an information theoretic measure of how much information is shared between a cluster and its ground truth classification that can detect a non-linear similarity between the two clusterings. Adjusted mutual information is the adjusted-for-chance variant of this that has reduced bias for varying cluster numbers.

Clustering Axiom

1. Formalism: A partition function F acts on a set S with

$$n \geq 2$$

points (the points being labeled as $\{1,2,\ldots,n\}$ along with a number of clusters integer

$$k > 0$$

and pair-wise distance among S. (These are the ONLY set of information available on S). The distance function

$$d: S \times S \to R$$

has the property that for $i, j \in S$,

$$d(i,j) \geq 0$$

and

$$d(i,j) = d(j,i)$$

These properties ensure that the distance function is non-negative, symmetric, and that two points are identical ONLY if the distance between is zero.

2. Clustering Function: The clustering/partition function F takes a distance function d on $S \times S$ and an integer

$$k \geq 1$$

to return a k-partition of S, a collection of non-empty disjoint subsets of S whose union is S. Two clustering functions are equivalent if and only if they output the same partitioning on all values of d AND k.

3. Clustering Axiom #1 – Scale Invariance: For any distance function d, number of cluster k, and a scalar

$$\alpha > 0$$

we should have

$$F(d,k) = F(\alpha.d,k)$$

This simply states that the clustering function is immune to stretching/shrinking.

4. Clustering Axiom #2 – k-Richness: For a fixed S and k, let $Range[F(\cdot,k)]$ be the set of all possible outputs while varying d. Then, for any number of clusters k, $Range[F(\cdot,k)]$ is equal to the set of all the k-partitions of S. This property ensures that any partition $\mathbb{\Gamma}$ has associated with it (or extractable using a) distance metric d such that

$$F(d,k) = \mathbb{\Gamma}$$

5. Clustering Axiom #3 - Consistency: If the like partitions are blended together with their distances shrunk (expanded) and "unlike partitions" are altered such that their distances expand (shrink), the clustering function $F(d',k)$ should get back to the same partition as $F(d,k)$ (Kleinberg (2002)).
6. Sample Clustering Applications:
    a. In field robotics to track objects and detect outliers for situational awareness (Bewley, Shekhar, Leonard, Upcroft, and Lever (2011))
    b. To find structural similarities by clustering chemical compounds into topological indices (Basak, Magnuson, Niemi, and Regal (1988))
    c. To find weather regimes or preferred sea-level pressure atmospheric patterns (Huth, Beck, Philipp, Demuzere, Ustrnul, Cahynova, Kysely, and Tveito (2008))

References

- Achtert, E., C. Bohm, H. P. Kriegel, P. Kroger, I. Muller-Gorman, and A. Zimek (2006): Finding Hierarchies of Subspace Clusters *LNCS: Advances in Knowledge Discovery in Databases* **4213** 446-453.
- Achtert, E., C. Bohm, and P. Kroger (2006): DeLiClu: Boosting Robustness, Completeness, Usability, and Efficiency of Hierarchical Clustering by a Closest Pair Ranking *LNCS: Advances in Knowledge Discovery and Data Mining* **3918** 119-128.
- Achtert, E., C. Bohm, P. Kroger, and A. Zimek (2006): Mining Hierarchies of Correlation Clusters *Proceedings of the 18th International Conference on Scientific and Statistical Database Management (SSDBM)* 119-128.
- Achtert, E., C. Bohm, H. P. Kriegel, P. Kroger, I. Muller-Gorman, and A. Zimek (2007): Detection and Visualization of Subspace Cluster Hierarchies *LNCS: Advances in Databases: Concepts, Systems, and Applications* **4443** 152-163.
- Achtert, E., C. Bohm, H. P. Kriegel, P. Kroger, and A. Zimek (2007): On Exploring Complex Relationships of Correlation Clusters *19th International Conference on Scientific and Statistical Database Management (SSDBM 2007)* ***7***.
- Agrawal, R., J. Gehrke, D. Gunopulos, and P. Raghavan (2003): Automatic Subspace Clustering of High-Dimensional Data *Data Mining and Knowledge Discovery* **11** 5.
- Ankerst, M., M. Breuning, H. P. Kriegel, and J. Sander (1999): OPTICS: Ordering Points to Identify the Clustering Structure *ACM SIGMOD International Conference on Management of Data* 49-60.
- Auffarth, B. (2010): Clustering a Genetic Algorithm with Biased Mutation Operator WCCI CEC.
- Bailey, K. (1994): *Numerical Taxonomy and Cluster Analysis* Typologies and Taxonomies.
- Basak, S. C., V. R. Magnuson, C. J. Niemi, and R. R. Regal (1988): Determining Structural Similarity of Chemicals using Graph Theoretic Indices *Discrete Applied Math* **19** 17-44.

- Bewley, A., R. Shekhar, S. Leonard, B. Upcroft, and B. Lever (2011): Real-time Volume Estimation of a Dragline Payload *IEEE International Conference on Robotics and Automation (ICRA '11)* 1571-1576.
- Can, F., and E. A. Ozkarahan (1990): Concepts and Effectiveness of the Cover-Coefficient-Based Clustering Methodology for Text Databases *ACM Transactions on Database Systems* **15 (4)** 483.
- Cattell, R. B. (1943): The Description of Personality: Basic Traits resolved into Clusters *Journal of Abnormal and Social Psychology* **38** 476-506.
- Cluster Analysis (Wiki): Wikipedia Entry for Cluster Analysis.
- Defays, D. (1977): An Efficient Algorithm for a Complete-Link Method *Computer Journal* **20 (4)** 364-366.
- Dunn, J. (1974): Well Separated Clusters and Optimal Fuzzy Partitions *Journal of Cybernetics* **4** 95-104.
- Ester, M., H. P. Kriegel, J. Sander, and X. Xu (1996): A Density-Based Algorithm for Creating Clusters in Large Spatial Databases with Noise *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)* 226-231.
- Estivill-Castro, V. (2002): Why so many Clustering Algorithms – A Position Paper *ACM SIGKDD Explorations Newsletter* **4 (1)** 65-75.
- Farber, I., S. Gunnemann, H. P. Kriegel, P. Kroger, E. Muller, E. Schubert, T. Seidl, and A. Zimek (2010): On Using Class Labels on Evaluation of Clusterings *MultiClust: Discovering, Summarizing, and Using Multiple Clusterings ACM SIGKIDD*.
- Fowlkes, E. B., and C. L. Mallows (1983): A Method for Comparing Two Hierarchical Clusterings **78** 553-569.
- Frey, B. J. and D. Dueck. (2007): Clustering by Passing Messages between Data Points *Science* **315 (5814)** 972-976.
- Hubert, L., and P. Arabie (1985): Comparing Partitions *Journal of Classification* **2 (1)**.
- Huth, R., C. Beck, A. Philipp, M. Demuzere, Z. Ustrnul, M. Cahynova, J. Kysely, and O. E. Tveito (2008): Classification of Atmospheric Circulation Patterns – Recent Advances and Applications *Annals of NY Academy of Sciences* **1146** 105-152.
- Kleinberg, J. (2002): An Impossibility Theorem for Clustering *Proceedings of the Neural Information Processing Systems Conference*.

- Kraskov, A., H. Stogbauer, R. G. Andrzejak, and P. Grassberger (2003): Hierarchical Clustering based on Mutual Information **arXiV**.
- Meila, M. (2003): Comparing Clusterings by the Variation of Information *Learning Theory and Kernel Machines* **2777** 173-187.
- Kriegel, H. P., P. Kroger, J. Sander, and A. Zimek (2011): Density-Based Clustering *WIREs Data Mining and Knowledge Discovery* **1 (3)** 231-240.
- Lloyd, S. (1982): Least Squares Quantization in PCM *IEEE Transactions in Information Theory* **28 (2)** 129-137.
- Manning, C. D., P. Raghavan, and H. Schutze (2008): *Introduction to Information Retrieval* **Cambridge University Press**.
- Ng, R., and J. Han (1994): Efficient and Effective Clustering Method for Spatial Data Mining *Proceedings of the 20th VLDB Conference* 144-155.
- Rand, W. M. (1971): Objective Criteria for the Evaluation of Clustering Models *Journal of the American Statistical Association* **66 (336)** 846-850.
- Roy, S., and D. K. Bhattacharyya (2005): An Approach to Finding Embedded Clusters Using Density Based Techniques *LNCS* **3816** 523-535.
- Sculley, D. (2010): Web-scale k-Means Clustering Proceedings of the 19th International Conference on the World Wide Web.
- Sibson, R. (1973): SLINK: An Optimally Efficient Algorithm for the Single-Link Cluster Method *Computer Journal* **16 (1)** 30-34.
- Tryon, R. C. (1939): *Cluster Analysis: Correlation Factor and Orthometric (Factor) Analysis for the Isolation of Unities in Mind and personality* **Edwards Brothers**.
- Wallace, D. L. (1983): Comment *Journal of the American Statistical Association* **78** 569-579.
- Zhang, T., R. Ramakrishnan, and M. Livny (1996): An Efficient Data Clustering Method for Very Large Databases *Proceedings of International Conference on Management of Data* 103-114.

Mixture Model

Introduction

1. Definition: In statistics, a mixture model is a probabilistic model for representing the presence of sub-populations within an overall population, without requiring that an observed data set should identify the sub-population to which an individual observation belongs.
2. Components of the Mixture Model: See Mixture Model (Wiki):
    a. N random variables corresponding to observations, each assumed to be distributed according to a mixture of k components with each component belonging to the same parameter family of distributions (e.g., all normal, all Zipfan, etc.) but with different parameters.
    b. N corresponding random latent variables specifying the identity of the mixture components of each observation, each distributed according to a k-dimensional categorical distribution.
    c. A set of k mixture weights, each of which is a probability, and the sum of all of which come to 1
    d. A set of k parameters, each set specifying the parameter of the corresponding mixture component. For example, observations distributed according to a mixture of one-dimensional Gaussian distribution will have a mean and a variance for each component. Observations distributed according to a V-dimensional categorical distribution will have a vector of V probabilities, collectively summing to 1.

Generic Mixture Model Details

1. Bayesian Setting: In a Bayesian setting, the mixture parameters and the weights will themselves be random variables, and prior distributions are placed over these variables. The

weights are typically viewed as a k-dimensional random vector drawn from a Dirichlet distribution (the conjugate prior of the categorical distribution), and the parameters will be distributed according to their respective conjugate priors.

2. The Basis Generic Parameter Mixture Model:
    a. k - The Number of Mixture Components
    b. N - The Number of Observations
    c. $\theta_{i=1,\ldots,k}$ - Parameters of the Distribution of the Observation associated with Component i
    d. $\varphi_{i=1,\ldots,k}$ - Mixture Weight, i.e., Prior Probability of a particular Component i
    e. $\vec{\emptyset}$ - k-dimensional vector composed of all the individual $\varphi_{i=1,\ldots,k}$; must sum to unity
    f. $x_{i=1,\ldots,k}$ - Observation i
    g. $z_{i=1,\ldots,k}$ - Component of Observation i
    h. $F(x|\theta)$ - Probability Distribution of an Observation parametrized on θ
    i. $x_{i=1,\ldots,k}$ - $Categorical(\emptyset)$
    j. $z_{i=1,\ldots,k} - F(\theta_{z_i})$
3. Bayesian Parametric Mixture Model:
    a. $k, N, \theta_{i=1,\ldots,k}, \varphi_{i=1,\ldots,k}, x_{i=1,\ldots,k}, z_{i=1,\ldots,k}, F(x|\theta)$ - As Above
    b. α - Shared Hyper-parameter for the Component Parameters
    c. β - Shared Hyper-parameter for the Mixture Weights
    d. $H(\theta|\alpha)$ - Prior Probability Distribution of Component Parameters, parametrized on α
    e. $\theta_{i=1,\ldots,k}$ - $H(\alpha)$
    f. φ - $Symmetrical\_Dirichlet_k(\beta)$
    g. $x_{i=1,\ldots,k}$ - $Categorical(\emptyset)$
    h. $z_{i=1,\ldots,k}$ - $F(\theta_{z_i})$
4. Nature of F and H: The above characterization uses F and H to describe arbitrary distributions over observations and parameters, respectively. Typically, H will be a conjugate prior of F. Most common choices for F are Gaussian for real-valued observations, and categorical for discrete observations.
5. Alternate Mixture Component Distributions:

a. Binomial Distribution => Used to model the number of positive occurrences (successes, yes votes, etc.) given the total number of occurrences
b. Multinomial Distribution => Similar to binomial distribution, but for counts of multi-way occurrences (e.g. yes/no/maybe in a survey)
c. Negative binomial distribution, for binomial type observations, but where the quantity of interest is the number failures before a given number of successes occurs
d. Poisson Distribution => For the number of occurrences of an event in a given period of time, for an event that is characterized by a fixed rate of occurrence
e. Log Normal Distribution => For positive real numbers that are assumed to grow exponentially
f. Multi-variate Normal Distribution (aka multivariate Gaussian distribution) => For vectors of correlated outcomes that are individually Gaussian distributed
g. A vector of Bernoulli distributed values => Corresponding to, for e.g., a black-white image, with each value representing a pixel.

Specific Mixture Models

1. Gaussian Mixture Model:
    a. k, N, $\varphi_{i=1,\ldots,k}$, $\vec{\emptyset}$, $x_{i=1,\ldots,k}$, $z_{i=1,\ldots,k}$ - As above
    b. $\mu_{i=1,\ldots,k}$ and ${\sigma^2}_{i=1,\ldots,k}$ - Mean and Variance of Component i
    c. $\mu,\ \lambda,\ \nu, {\sigma_0}^2$ - Shared hyper-parameters
    d. $\mu_{i=1,\ldots,k} \sim \aleph\left(\mu_0, \lambda {\sigma_i}^2\right)$; ${\sigma_{i=1,\ldots,k}}^2 \sim Inverse\_Gamma\left(\nu, {\sigma_0}^2\right)$
    e. $\vec{\emptyset}$ - $Symmetrical\_Dirichlet_k(\beta)$; $z_{i=1,\ldots,N}$ - $Categorical\left(\vec{\emptyset}\right)$
    f. $x_{i=1,\ldots,N} \sim \aleph\left(\mu_{z_i}, {\sigma^2}_{z_i}\right)$
2. Multivariate Bayesian Gaussian Mixture Model: In a multivariate distribution (one modeling $\vec{x}$ with N random variables) one may model a vector of parameters (such as several observations of a signal or patches within an image) using a Gaussian mixture model with a prior distribution on the vector of estimates given by

$$p(\vec{\theta}) = \sum_{i=1}^{k} \varphi_i \aleph(\vec{\mu_i}, \vec{\sigma_i}^2)$$

3. Multivariate Bayesian Gaussian Estimation: To incorporate the above prior into a Bayesian estimation, the prior is multiplied with a known distribution $p(\vec{x}|\vec{\theta})$ of the data set $\vec{x}$ conditioned on $\vec{\theta}$ to be extended, thus resulting in

$$p(\vec{\theta}|\vec{x}) = \sum_{i=1}^{k} \tilde{\varphi}_i \aleph\left(\vec{\tilde{\mu}_i}, \vec{\tilde{\sigma}_i}^2\right)$$

4. Bayesian Multivariate Parameter Estimation: The new parameter set $\tilde{\varphi}_i, \vec{\tilde{\mu}_i}, \vec{\tilde{\sigma}_i}^2$ are then updated using the Expectation-Maximization algorithm (Yu (2012)). Although EM-based parameter updates are well-established, providing initial estimates for the parameters above is an area of active research. The formulation above yields a closed-form solution to the complete prior, and estimates of $\vec{\theta}$ are obtained by one of several standard Bayesian estimators (such as mean or maximization of the posterior distribution).
5. Patch-wise Multivariate Bayesian Gaussian: Such Bayesian estimates of Gaussian multivariates are useful for assuming path-wise shapes of images and clusters. For image representation, each Gaussian may be tilted, expanded, and warped according to $\vec{\tilde{\sigma}_i}^2$. A single Gaussian distribution of the set is fit to each patch (say 8×8 pixels) of image. Notably, any distribution of points around a cluster (e.g., k-means) may be accurately determined given enough Gaussian components, but rarely are $k > 20$ such components needed to accurately model a given image distribution or cluster of the data.
6. Categorical Mixture Model:
    a. k, N, $\varphi_{i=1,\ldots,k}$, $\vec{\emptyset}$, $x_{i=1,\ldots,k}$, $z_{i=1,\ldots,k}$ - As above
    b. V - Dimension of the Categorical Observations (e.g., size of the Word Volabulary)
    c. $\theta_{i=1,\ldots,k;j=1,\ldots,V}$ - Probability of Component i observing Item j
    d. $\vec{\theta}_{i=1,\ldots,k}$ - Vector of Dimension k, composed of $\theta_{i;1,\ldots,V}$; sums to unity
    e. $x_{i=1,\ldots,k}$ - $Categorical(\emptyset)$

f. $z_{i=1,\ldots,k}$ - $Categorical(\theta_{z_i})$

7. Bayesian Categorical Mixture Model:
    a. $k, N, \varphi_{i=1,\ldots,k}, \vec{\emptyset}, V, \theta_{i=1,\ldots,k;j=1,\ldots,V}$ - As represented as above in the non-Bayesian mixture model case
    b. α - Shared Concentration Hyper-parameter of $\vec{\theta}$ for each Component
    c. β - Concentration Hyper-parameter for $\vec{\emptyset}$
    d. $\vec{\emptyset} \sim Symmetrical\_Dirichlet_k(\beta)$; $\theta_{i=1,\ldots,k} \sim Symmetrical\_Dirichlet_V(\alpha)$
    e. $z_{i=1,\ldots,k} \sim Categorical(\vec{\emptyset})$; $x_{i=1,\ldots,k} \sim Categorical(\theta_{z_i})$

Mixture Model Samples

1. Financial Returns: Financial returns often behave differently in normal situations, and during crisis times. A mixture model (Dinov (2005)) for returns data seems reasonable. Sometimes the model used is a jump-diffusion model, or a model that is a mixture of two normal distributions.
2. Home Prices: This can assume that the prices are accurately described by a mixture model with K different components in one area, each distributed as a normal distribution with a corresponding unknown mean and variance.
3. Document Topics: This is also called a topic model. This assumes that a document is composed of N different words from a total vocabulary of size V where each word corresponds to one of K possible topics.
    a. Parameter Estimation => EM is applied to model tails and produce realistic results due to the excessive number of parameters. Additional assumptions are needed, e.g., a prior distribution is placed over the parameters describing the topic distribution where only a small number of words have significantly non-zero probabilities.
    b. Topic Identity Constraint => Additional constraints placed over the topic identities of words to take advantage of the natural clustering include:

i. Placing a Markov chain (i.e., the latent variables specifying the mixture component for each observation) on the topic identities (which are really just the cluster labels) represents the fact that nearby words belong to similar topics. This results in a hidden Markov model – specifically one where a prior distribution is placed over the state transitions that favor transitions that stay in the same state.
ii. Natural Clustering Constraints => Another possibility to specify additional constraints is to model the topic identities using the latent Dirichlet allocation model, which divides the word set into D different documents and assumes that in each only a small number of topics can occur with any frequency.

4. Handwriting Recognition: Consider a $N \times N$ black-and-white pixel (Bishop (2006)) that is a scan of a hand-written digit, i.e., $0 - 9$. We create a mixture model with

$$k = 10$$

different components (one per digit number) where each component is a vector of N^2 Bernoulli distributions (one per pixel). This model can be trained with EM on an unlabeled set of hand-written digits, thus creating the clustering. The same model can then be used to recognize the digit of another image simply by holding the parameters constant, computing the probability of realization of the new image for each possible digit, and returning the digit with the highest possibility.

5. Fuzzy Image Segmentation: In fuzzy/soft segmentation, any pattern can have "ownership" over any single pixel. If the patterns are Gaussian, fuzzy segmentation naturally results in Gaussian mixtures. Combined with other analytic/geometric tools (e.g., phase transitions over diffusive boundaries), such regularized mixture models could lead to more realistic and computationally efficient segmentation models (Shen (2006)).

Identifiability

1. Definition: Identifiability refers to existence of a unique characterization for any one of the models in the class (family) being considered. Estimation procedure may not be well-defined, and asymptotic theory may not hold if a model is not identifiable.
2. Mathematical Specification: Consider a mixture of parametric distributions belonging to the same class. Let

$$J = \{f(\blacksquare, \theta) : \theta \in \Omega\}$$

be the class of all the component distributions. Then the convex hull K of J defines the class of all the finite mixture distributions in J:

$$K = \left\{p(\blacksquare) : p(\blacksquare) = \sum_{i=1}^{n} a_i f_i(\blacksquare, \theta_i)\,;\; a_i > 0;\; \sum_{i=1}^{n} a_i = 1\,;\; f_i(\blacksquare, \theta_i) \in J\ \forall\ i, n\right\}$$

K is said to be identifiable if all its members are unique, that is, given two members p and p' in K being mixtures of k distributions and k' distributions respectively in J, we have

$$p = p'$$

if and only if, first,

$$k = k'$$

and second, we can re-order the summations such that

$$a_i = a_i'$$

and

$$f_i = f_i'$$

for all i.

3. Parameter Estimation and System Identification: Typical approaches that use EM or MAP consider separately the questions of parameter estimation and system identification, that is to say, a distinction is made between the determination of the number and the functional forms of the components within the mixture, and the estimation of the corresponding parameter values.
4. Joint Parameter Estimation and System Determination: Notable departures to the separated system determination/parameter estimation methods are the graphical methods of Tarter and Lock (Tarter (1993)), more recently the minimum message length (MML) techniques (Figueiredo and Jain (2002)), and to some extent, the moment matching pattern analysis routines suggested by McWilliam and Loh (2008).

Expectation Maximization

1. Definition: This is seemingly the most popular technique used in determining the parameters of the mixture with an a priori given number of components. EM/MLE is of particular appeal for finite normal mixtures where closed-form expressions are possible, an example of which is provided below (this is the Dempster-Shafer iterative algorithm):

$$w_s^{j+1} = \frac{1}{N}\sum_{t=1}^{N} h_S^j(t)$$

$$\mu_S^{j+1} = \frac{\sum_{t=1}^{N} x(t) h_S^j(t)}{\sum_{t=1}^{N} h_S^j(t)}$$

$$\rho_S^{j+1} = \frac{\sum_{t=1}^{N} \left[x(t) - \mu_S^j\right]\left[x(t) - \mu_S^j\right]^T h_S^j(t)}{\sum_{t=1}^{N} h_S^j(t)}$$

The corresponding posterior probabilities are

$$h_s^j(t) = \frac{w_s^j p_s\left(x(t), \mu_S^j, \rho_S^j\right)}{\sum_{i=1}^{n} w_i^j p_i\left(x(t), \mu_i^j, \rho_i^j\right)}$$

2. State Evolution Updates: Using the above, on the basis of the current estimate for the parameters, the conditional probability for a given observation $x(t)$ being generated from state s is determined for each

$$t = 1, \ldots, N$$

N being the sample size. The parameters are then updated such that the new component weights correspond to the average conditional probability, and component mean and co-variance is the component specific weighted average of the mean and the covariance of the entire sample.

3. EM Steps:
    a. The Expectation Step => With initial Guesses for the parameters of our mixture model, the "partial membership" in each constituent distribution is computed by calculating the expectation values for the membership variables of each data point. Thus, given the data point x_j and a distribution Y_i, the membership value is given as

$$y_{ij} = \frac{a_i f_Y\left(x_j;\ \theta_i\right)}{f_X\left(x_j\right)}$$

    b. The Maximization Step => The mixing coefficients a_i and the means of the membership values over N data points are given as

$$a_i = \frac{1}{N}\sum_{j=1}^{N} y_{ij}$$

The corresponding computed model parameters are computed using

$$\theta_i = \frac{\sum_{j=1}^{N} x_j y_{ij}}{\sum_{j=1}^{N} y_{ij}}$$

With new estimates for a_i and θ_i, the expectation step is repeated until model parameters converge.

4. Advantages of the EM Approach: Dempster, Laird, and Rubin (1977) showed that each successive EM iteration will not decrease the likelihood, a property not shared by other gradient based maximization techniques. EM also embeds within itself the constraints on the probability vector, and, for sufficiently large sample sizes, also maintains the positive definiteness of the covariance iterates. This is a key advantage, since explicitly constrained methods incur extra computational costs to check and maintain appropriate values.
5. Drawbacks of the EM Approach:
    a. EM is a first-order algorithm, so convergence is slow. Superliner, second-order Newton, and quasi-Newton methods are therefore more preferred (although it needs to be noted that even if the convergence likelihood is rapid – which is what really counts – the convergence of the estimated parameters need not (Xu and Jordan (1996))).
    b. EM is a local maximizer (McLaughlan (2000)), and displays sensitivity to initial values. Approaches to overcoming these include evaluating EM at several points in the parameter space, but this can become computationally expensive too. Thus, approaches such as annealing EM may be preferred – here the initial components are essentially forced to overlap, providing a less heterogenous basis for the initial guesses.
6. Minimum Message Length (MML) Algorithm: Figueiredo and Jain (2002) note that the convergence to meaningless parameter values obtained at the boundary (where regularity conditions breakdown) is frequently observed when the number of model components exceeds the optimal/true one. On this basis, they suggest a unified approach to estimation and identification in which the initial n is chosen to greatly exceed the optimal value. This optimization routine is constructed via a minimum message length criterion that effectively

eliminates a candidate component if there is insufficient information to support it. In this way it is possible to systematize reductions in n and consider estimation and identification jointly.

Alternatives to EM

1. Monte Carlo Markov Chain Alternative to EM: Here the mixture model parameters can be deduced using posterior sampling. Essentially this will be an incomplete data problem whereby membership of the data points is the missing data. A 2-step iterative procedure known as Gibbs' sampling may be used.
    a. 2-step Gaussian MCMC => As in the EM case initial guesses for the parameters of the mixture model are made. Instead of computing partial memberships for each elemental distribution, a membership value for each data point is drawn from a Bernoulli distribution (that is, it will be assigned to either the first or the second Gaussian distribution). The Bernoulli parameter θ for the draw is determined for each data point on the basis of one of the constituent distributions. Draws from the distribution generate membership associations for each data point. Plug-in estimators can them be used in the Maximization step of the EM to generate a new set of mixture model parameters, and the binomial draw is repeated.
2. Moment Matching Methods: In this approach, the parameters of the mixture are determined such that the composite distribution has moments matching the given value set (Wang (2001)). However, solutions to the moment equations may pose non-trivial computation challenges and other inefficiencies compared to EM (Day (1969)).
3. Spectral Methods - Applicability: Mixture model estimations are amenable to spectral methods if the data points x_i are points in high-dimensional real space, but the hidden distributions are known to be log-concave (such as Gaussian distribution or exponential distribution).
    a. Estimation Techniques => Spectral methods of learning mixture models are based on the use of SVD of a matrix containing the data set. The idea of the technique is to consider the top k singular vectors where k is the number of distributions to be learned. The projection of each data point to a linear subspace spanned by those

vectors groups those points originating from the same distribution close together, while the points from different distributions stay far apart.

   b. Robustness Property => One distinctive feature of the spectral method is that if the distributions satisfy a certain separation condition (e.g., they are not too close), the estimated mixture will be very close to the true one with very high probability.

4. Graphical Methods: In graphical approach to mixture identification a kernel function is applied to an empirical frequency plot to reduce the intra-component variance (Tarter (1993)). This helps more readily identify components with differing means. While this method does not require knowledge of the number or the basis functional form for the components, its success does rely on the choice of the kernel parameters, which to some extent implicitly embeds assumptions about the component structure.
5. Other non-EM Methods: Some of the methods can even learn mixtures with heavy tailed distributions, including those with infinite variance. In this setting the EM approaches won't work, as the expectation step diverges due to the presence of the outliers.

Mixture Model Extensions

1. Bayesian Extensions: Additional Bayesian levels can be added to the model defining the mixture model. For example, in the common latent Dirichlet allocation topic model, the observations are sets of words drawn from D different documents, and the K mixture components represent topics that are shared across the documents. Each document has a different set of mixture weights which specify the topics prevelant in those documents. All sets of mixture weights share common hyper parameters.
2. Hidden Markov Extensions: Another common extension is to connect the latent variables defining the mixture component identities into a Markov chain instead of assuming that they are independent identically distributed random variables. This results in a hidden Markov model.
3. Application Areas: The advent of processing power has popularized EM/ML techniques (McLaughlan (1988)), and is now used vastly in areas such as agriculture, botany,

economics, electrophoresis, finance, fisheries, genetics, geology, medicine, paleontology, psychology, sedimentology, and zoology (Titterington, Smith, and Makov (1985)).

References

- Bishop, C. M. (2006): *Pattern Recognition and Machine Learning* **Springer** New York.
- Day, N. E. (1969): Estimating the Components of the Mixture of a Normal Distribution *Biometrika* **56 (3)** 463-474.
- Dempster, A. P., N. M. Laird, and D. B. Rubin (1977): Maximum Likelihood from Incomplete Data via the EM Algorithm *Journal of the Royal Statistical Society Series B* **39 (1)** 1-38.
- Dinov, I. D. (2008): Expectation Maximization and Mixture Modeling Tutorial *California Digital Library Statistics Online Computational Resource*.
- Figueiredo, M. A. T., and A. K. Jain (2006): Unsupervised Learning of Finite Mixture Models **24 (3)** 381-396.
- McLaughlan, G. J. (2000): *Finite Mixture Models* **Wiley**.
- McWilliam, N., and K. Loh (2008): Incorporating Multi-Dimensional Tail Dependencies in the Valuation of Credit Derivatives (Working paper).
- Mixture Model (Wiki): Wikipedia Entry for Mixture Model.
- Shen, J. (2006): A Stochastic Variational Model for Soft Mumford-Shah Segmentation *International Journal of Biomedical Imaging* **2006** 2-16.
- Tarter, M. E. (1993): *Model Free Curve Estimation* **Chapman & Hall**.
- Titterington, D. M., A. F. M. Smith, and U. E. Makov (1986): *Statistical Analysis of Finite Mixture Distributions* **Wiley**.
- Wang, J. (2001): Generating Daily Changes in Market Variables Using a Multivariate Mixture of Normal Distributions *Proceedings of the 33rd Winter Conference on Simulation, IEEE Computer Society* 283-289.
- Xu, L., and M. I. Jordan (1996): On the Convergence Properties of the EM Algorithm for Gaussian Mixtures *Neural Computation* **8 (1)** 129-151.

Deep Learning

Introduction

1. Definition: Deep learning comprises of a set of algorithms in machine learning that attempt to model high level abstractions in data by using architectures composed of multiple non-linear transformations (Deep Learning (Wiki), Bengio, Courville, and Vincent (2013)).
2. Architectures for Learning Representations: Deep learning is part of a broader family of machine learning methods based on learning representations. These architectures include deep neural, convolutional deep neural networks, and deep belief networks.
3. Earlier Architectures: Original deep-learning architectures based on the standard ANN back-propagation algorithms (Werbos (1974)) and the neocognitron (Fukushima (1980)) fell out of favor due to the speed issues. Vanishing gradient problem was identified to be the key issue (Hochreiter (1991), Hochreiter, Bengio, Frasconi, and Schmidhuber (2001)), and ANN approaches were replaced by SVMs etc.
4. Many Layered Feed Forward ANN: Hinton (2007) showed how a many layered feed-forward neural network can be effectively pre-trained one layer at a time, treating each layer in turn as an unsupervised Boltzmann machine, and then using supervised back-propagation for fine-tuning.
5. Unsupervised Deep Hierarchies: Schmidhuber (1992) had already implemented a very similar idea for the more general case of unsupervised hierarchies of recurrent neural networks, demonstrating the benefits for speeding up supervised learning (Schmidhuber (2013)).
6. Usage of Deep Learning: Application areas such as computer vision and automatic speech recognition (ASR) use evaluation data sets such as MNIST and TIMIT respectively to improve deep learning applications. For instance, convolutional neural networks are widely used - although more in computer vision than in ASR.

7. Impact of Computational Advances in Deep Learning: Powerful GPUs highly suited for number crunching and matrix/vector math have enormously enhanced deep learning by reducing the training times from weeks to hours (Raina, Madhavan, and Ng (2009), Ciresan, Meier, Gambardella, and Schmidhuber (2010)).

Unsupervised Representation Learner

1. Deep Learning as a Representation Learner: Deep learning algorithms are based on distributed representations – whose underlying idea is that the observed data is generated as a result of many different factors on different levels. Deep learning adds the assumption that these factors are organized into multiple levels, corresponding to different levels of abstraction or composition. Variable numbers of layers and layer sizes can be used to provide different amounts of abstraction (Bengio, Courville, and Vincent (2013)).
2. Hierarchical Explanatory Factors: Deep learning algorithms exploit the idea of hierarchical explanatory factors. Concepts are learned from other concepts, with the more abstract, higher level concepts being learned from the lower level ones. These architectures are often constructed using a greedy layer-by-layer method that models this data. Deep learning helps disentangle these abstractions and pick out the features that are useful for learning (Bengio, Courville, and Vincent (2013)).
3. Deep Learning as Unsupervised Learning: Many deep learning algorithms are actually framed as unsupervised learning problems. This ease of use in unlabeled inputs (very prevalent in real life situations) makes deep learning techniques widely applicable.

Deep Learning Using ANN

1. Motivation: Some of the most successful deep learning methods involve ANN (which was inspired by the 1959 biological model of the primary visual cortex as being composed of 2 types of cells – the simple and the complex). Many ANN's may be viewed as cascading models (Reisenhuber and Poggio (1999)) of these cell types.

2. Deep Learning using Convolutional Neural Networks: Fukushima's Neocognitron (Fukushima (1980)) introduced the convolutional NN that is partially trained by unsupervised learning. LeCun, Boser, Denker, Henderson, Howard, Hubbard, and Jackel (1989) applied supervised back propagation to such architectures.
3. Challenges Training Deep ANN's: Hochreiter (1991) and Hochreiter, Bengio, Frasconi, and Schmidhuber (2001) identified the cause for the failure to be able to train deep ANN's from scratch as being due to the "vanishing gradient problem", which not only affects many-layered feed-forward networks, but also recurrent neural networks. The latter are trained by unfolding them into very deep feed-forward networks, where a new layer is created for each time step of the input sequence processed by the network. As errors propagate layer-to-layer, they shrink exponentially with the number of layers.
4. Training Deep ANN's: Multi level Hierarchy of Networks: In this approach, the multi-level hierarchy of networks are pre-trained one level at-a-time through unsupervised learning, and are fine-tuned through back-propagation (Schmidhuber (1992)). Each level learns a compressed representation of the observations that is fed to the next level.
5. Training Deep ANN's - LSTM: Deep, multi-dimensional Long-Short Term Memory (LSTM) networks have been shown to be powerful in deep learning with many non-linear layers (Hochreiter and Schmidhuber (1997)). Applications to Connected Handwriting Recognition without any prior knowledge of the language to be learned for three different languages have also been demonstrated (Graves and Schmidhuber (2009), Graves, Liwicki, Fernandez, Bertolami, Bunke, and Schmidhuber (2009)).
6. Gradient Sign Learning: Sven Behnke (2003) relied only on the sign of the gradient (R prop) when training his Neural Abstraction Pyramid to solve problems like image re-construction and face localization.
7. Alternate Methods to Structure ANN Learning: These use unsupervised pre-training to structure a neural network, making it first learn generally useful feature detectors. Then the network is trained further by supervised back-propagation to classify labeled data.
8. Hinton's Deep Learning Model: The deep learning model of Hinton, Osindero, and Teh (2006) involves learning the distribution of the high-level representation using successive layers of binary/real-valued latent variables. It uses a restricted Boltzmann machine (Smolensky (1986)) to model each new layer of high-level features. Each new layer

guarantees an increase on the lower-bound of the log-likelihood of the data if trained properly, thus improving the model. Once sufficient number of layers have been learnt, the deep architecture maybe used as a generative model by reproducing the data when sampling down the model (an "ancestral pass") from the top-level feature activations. Hinton (2009) reports that these are more effective feature extractors over high-dimensional, structured data.

9. High Level Concept Learning in Action: The Google Brain team led by Andrew Ng and Jeff Dean created a neural network that learned to recognize high-level concepts such as cats purely by watching unlabeled images taken from youtube videos (Markoff (2012), Ng and Dean (2012)).
10. Learning with Compute Power: Ciresan, Meier, Masci, Gambardella, and Schmidhuber (2010) showed that, despite the "vanishing gradient problem", superior processing power of GPUs makes back-propagation feasible for deep feed-forward neural networks with many layers. Their method outperformed all other learning techniques on the old famous MNIST handwritten digits problem of LeCun.
11. State-of-the-Art in Deep Learning: Currently the state-of the-art in feed forward networks alternates convolutional layers and max-pooling layers (Ciresan, Meier, Masci, Gambardella, and Schmidhuber (2011), Martines, Bengio, and Yannakakis (2013)) topped by several pure classification layers. Training is done without supervised pre-training. Applications that use this architecture are described in Ciresan, Meier, Masci, Gambardella, and Schmidhuber (2011), Ciresan, Meier, Masci, and Gambardella (2012), and Ciresan, Giusti, Gambardella, and Schmidhuber (2012).
12. Human Competitive ANN's: Above architecture was also among the first artificial pattern recognizer to achieve human-competitive performance on certain tasks (Ciresan, Meier, and Schmidhuber (2012a)).

Deep Learning Architectures

1. Deep Neural Network: A Deep Neural Network (DNN) is an ANN with more than one hidden layer of units between the input and the output layers. Similar to shallow ANN's, a

DNN can model complex non-linear relationships. The extra layers give it added levels of abstraction, thereby increasing its modeling capability. DNN's are typically designed as feed-forward networks, but recent research has successfully applied the deep learning architectures to recurrent neural networks for applications such as language modeling (Mikolov, Karafiet, Burget, Cernocky, and Khudanpur (2010)).

2. Convolution DNN: Convolutional DNN (CNN) is used in computer vision where its success is well-documented (LeCun, Bottou, Bengio, and Haffner (1998)). More recently, CNN's are used for acoustic modeling of ASR with considerable improvements over previous models (Sainath, Mohamed, Kingsbury, and Ramabhadran (2013)).
3. DNN Formulation: A DNN can be discriminatively trained with the standard back-propagation algorithm. The weight updates can be done via stochastic gradient descent using the following equation

$$\Delta w_{ij}(t+1) = \Delta w_{ij}(t) + \eta \frac{\partial C}{\partial w_{ij}}$$

where η is the learning rate, and C is the cost function.

4. DNN Parameters: The choice of the cost function depends on factors such as the learning types (supervised, unsupervised, reinforcement etc.) and the activation function. For example, when performing supervised learning on a multi-class classification problem, common choice for the activation function and the cost function are the softmax and the cross-entropy function, respectively.

Challenges with the DNN Approach

1. Overfitting: Overfitting happens when the DNN is allowed to model rare dependencies between units that occur in the training data. DNN's are very prone to overfitting because of their added layers of abstraction, which can allow them to model rare dependencies in the training data. Regularization methods that weight decay (l_2-regularization) or sparsity regularization (l_1-regularization) can be applied during training to help combat overfitting

(Bengio, Boulanger-Lewandowski, and Pascanu (2013)). Finally, in dropout regularization, some units are randomly omitted from the hidden layers during training – this helps break those rare dependencies that occur in the data (Dahl, Sainath, and Hinton (2013)).

2. Computation Time: Back propagation and gradient descent have been the preferred training method for training these structures due to the ease of implementation and their tendency to converge to better local optima in comparison with other training methods. However, they can be computationally intensive, esp. when applied to DNN's.
3. Computation Details: There are many parameters to be considered with a DNN – such as the size (the number of layers, and the number of units per layer), the learning rate, the initial weights, etc. Sweeping through the parameter space may not be feasible for many tasks due to the time cost (Hinton (2010)). Various tricks such as 'mini-batching" (where gradients on several training examples are computed at once, rather than individual examples (Hinton (2010))) have been shown to speed up computation.
4. GPU Power Deployment: The sheer processing power of GPU's has had the most significant impact, particularly in the matrix and the vector computation space. However, it is hard to make use of large cluster machines for training DNN's, so better methods of parallelizing the training are necessary.

Deep Belief Networks (DBN)

1. Definition: A DBN is a probabilistic generative model made up of multiple layers of hidden units. It can be looked at as a combination of simple learning modules that make up each layer (Hinton (2009)).
2. Training a DBN: A DBN can be used of generatively pre-training a DNN by using the learned weights as initial weights. Back propagation or other discriminative algorithms can then be used for fine-tuning these weights. This is particularly useful in situations where limited training data is available, as poorly initialized weights can have significant impact on the performance of the final model. These pre-trained weights are in a region of weight space closer to optimal (than just random initialization). This enables improved modeling and faster

convergence during fine-tuning (La Rochelle, Erhan, Courville, Bergstra, and Bengio (2007)).

3. DBN as Layers of RBM: A DBN can be effectively trained in an unsupervised, layer-by-layer manner where the layers are typically made up of restricted Boltzmann machines (RBM's). An RBM is an undirected, generative energy-based model with an input layer and a single hidden layer. Connections only exist between the visible units of the input layer and the hidden units of the hidden layer; there are no visible-visible or hidden-hidden connections.
4. RBM Training: This is also known as Contrastive Divergence (CD, Hinton (2002)). This method provides an approximation to ML that would ideally be applied for learning the weights of a RBM (Hinton (2010)).
5. Weight Updates in RBM Training: In training a single RBM, weight updates are performed using gradient descent via the equation

$$\Delta w_{ij}(t+1) = \Delta w_{ij}(t) + \eta \frac{\partial log[p(\vec{v})]}{\partial w_{ij}}$$

Here $p(\vec{v})$ is the probability of the visible vector $\vec{v}$, and is given by

$$p(\vec{v}) = \frac{1}{Z}\sum_{h} e^{-E(\vec{v},h)}$$

where Z is the partition function normalizer, $E(\vec{v}, h)$ is the energy function assigned to the state of the network – lower energy indicates that the network is in a more desirable configuration.

6. Gradient in RBM Training: The gradient $\frac{\partial log[p(\vec{v})]}{\partial w_{ij}}$ has the simple form $\langle v_i h_j \rangle_{Data} - \langle v_i h_j \rangle_{Model}$ where the expectation are with respect to $p(\vec{v})$. The practical problem in sampling $\langle v_i h_j \rangle_{Model}$ is that this requires running alternating Gibbs' sampling for a long time. CD addresses this by running Gibbs' sampling for a smaller number of steps –

$$n = 1$$

has been shown to perform well. After n steps, the data is sampled and used in place of $\langle v_i h_j \rangle_{Model}$ (Hinton (2010)).

7. The CD Procedure:
    a. Initialize the visible units to a training vector.
    b. Update the hidden units in parallel given the visible units from

$$p(h_j = 1|\vec{v}) = \sigma\left(b_j + \sum_i v_i w_{ij}\right)$$

Here, σ represents the sigmoid function, whereas b_j is the bias of h_j.

    c. Update the visible units in parallel given the hidden units from

$$p(v_j = 1|\vec{h}) = \sigma\left(a_j + \sum_i h_i w_{ij}\right)$$

a_i is the bias of v_i, and this step is called the reconstruction step.

    d. Re-construct the hidden units in parallel given the visible reconstructed units as in the previous step.
    e. Finally perform the weight update from

$$\Delta w_{ij} \propto \{\langle v_i h_j \rangle_{Data} - \langle v_i h_j \rangle_{Model}\}$$

with the proportionality constant being the ANN update rate.

8. Stacking up RBM's: Once an RBM is trained, another can be stacked on top to create a multi-layer model. With each stack, the corresponding visible input layer can be initialized to a training vector, and values for the units in the already trained RBM layers are assigned using the current weights and biases. The final layer of the already trained layers is used as

an input to the new RBM. The new RBM is trained as above, and the whose process is repeated until a stopping criterion is achieved (Bengio (2009)).

9. Effectiveness of the CD Algorithm: Despite CD's crude similarity to ML (CD has been shown to not follow the gradient descent of any function), empirical results have shown CD to be an effective method for use with deep training architectures (Hinton (2010)).

Convolutional Neural Networks (CNN)

1. Definition: A CNN is composed of one/more convolutional layers with fully connected layers on the top. The CNN uses tied weights and pooling layers. This architecture enables CNN's to take advantage of the 2D data of the input layer.
2. Advantages of a CNN: The CNN has demonstrated strengths in both image and speech applications. They may also be trained with the standard back propagation techniques. CNNs are easier to train than the other regular, deep, feed-forward ANN's and have much fewer parameters to estimate.

Deep Learning Evaluation Data Sets

1. TIMIT - ASR Evaluation Data Set: TIMIT is a common data set used for the evaluation of the Deep Learning Architecture. It contains 630 speakers from 8 major dialects of American English, with each speaker reading 10 different sentences (TIMIT (1993)). The small size allows many different configurations to be tried effectively.
2. MNIST Image Classification Data Set: MNIST is composed of hand-written digits and is composed of 60,000 training examples and 100,000 test examples. Similar to TIMIT, its smaller size allows multiple configurations to be tested. A comprehensive list of results on this set are presented in MNIST (2013), and the current best result is an error rate of 0.23% (Ciresan, Meier, and Schmidhuber (2012a, 2012b)).

Neurological Basis of Deep Learning

1. Neurological Computational Brain Learning: The neocortex appears to employ computational deep learning models using a hierarchy of filters, where each layer captures some of the information of the operating environment and the passes the remainder (along with a modified base signal) to layers further up in the hierarchy (Blakeslee (1995), Elman, Bates, Johnson, Karmiloff-Smith, Parisi, and Plunkett (1996), Shrager and Johnson (1996), Quartz and Sejnowski (1997), Utgoff and Stracuzzi (2002)).
2. Deep Learning as a Result of Human Cognition: The theory of deep learning sees the co-evolution of culture and cognition as a fundamental condition of human evolution (Shrager and Johnson (1995), Bufill, Agusti, and Blesa (2011)).

Criticism of Deep Learning

1. Theoretical Foundations: Many criticisms of deep learning stem from the lack of theory surrounding many of the methods. Most of the deep learning architecture are built around some form of gradient descent. Theoretical treatment of related algorithms such as CD are unclear (does it converge? Is it fast? Exactly what does it approximate?). Most confirmations are empirical, not theoretical.

References

- Behnke, S. (2003): Hierarchical Neural Networks for Image Interpretation *Lecture Notes in Computer Science* **2766 Springer**.
- Bengio, Y. (2009): Learning Deep Architectures for AI *Foundations and Trends in Machine Learning* **2 (1)** 1-127.
- Bengio, Y., N. Boulanger-Lewandowski, and R. Pascanu (2013): Advances in Optimizing Recurrent Networks **arXiV**.

- Bengio, Y., A. Courville, and P. Vincent (2013): Representation Learning: A Review and new Perspectives *IEE Transactions PAMI, Special Issue Learning Deep Architectures*.
- Blakeslee, S. (1995): In Brain's Early Growth, Time Table may be Critical *The New York Times Science Section* B5-B6.
- Bufill, E. J. Agusti, and R. Blesa (2011): Human Neoteny Revisited: The Case of Synaptic Plasticity *American Journal of Human Biology* **23 (6)** 729-739.
- Ciresan, D. C., U. Meier, L. M. Gambardella, and J. Schmidhuber (2010): Deep Big Simple Neural Nets for Handwritten Digit Recognition *Neural Computation* **22** 3207-3220.
- Ciresan, D. C., U. Meier, J. Masci, L. M. Gambardella, and J. Schmidhuber (2011): Flexible, High Performance Convolutional Neural Networks for Image Classification *International Joint Conference on Artificial Intelligence (IJCAI-2011 Barcelona).*
- Ciresan, D. C., A. Giusti, L. M. Gambardella, and J. Schmidhuber (2012): Deep Neural Networks Segment Neuronal Membranes in Electron Microscopy Images, in: *Advances in Neural Information Processing Systems (NIPS 2012)* **Lake Tahoe**.
- Ciresan, D. C., U. Meier, J. Masci, and J. Schmidhuber (2012): Multi-column Deep Neural Network for Image Classification *IEEE Conference on Computer Vision and Pattern Recognition CVPR 2012*.
- Ciresan, D. C., U. Meier, and J. Schmidhuber (2012a): Multi-column Deep Neural Network for Traffic Sign Classification *Neural Networks*.
- Ciresan, D. C., U. Meier, and J. Schmidhuber (2012b): Multi-column Deep Neural Network for Image Classification *Technical Report No. IDSIA-04-12'*.
- Dahl, G., T. N. Sainath, and G. E. Hinton (2013): Improving Deep Neural Networks for LVCSR using Rectified Linear Units and Dropout *ICASSP '13*.
- Elman, J. L., E. A. Bates, M. H. Johnson, A. Karmiloff-Smith, D. Parisi, and K. Plunkett (1996): *Rethinking Innateness* **MIT Press**.
- Graves, A., and J. Schmidhuber (2009): Offline Handwriting Recognition in Multidimensional Recurrent Neural Networks, in: *Advances in Neural Information Processing Systems 22 (NIPS '22): Neural Information Processing Systems (NIPS) Foundation: Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta (eds.)*

- Graves, A., M. Liwicki, S. Fernandez, R. Bertolami, H. Bunke, and J. Schmidhuber (2009): A Novel Connectionist System for Improved Unconstrained Handwriting Recognition *IEEE Transactions on Pattern Analysis and Machine Recognition* **31 (5)**.
- Deep Learning (Wiki): Wikipedia Entry for Deep Learning.
- Fukushima, K. (1980): Neocognitron: A Self-Organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by a Shift in Position *Biol. Cybern.* **36** 193-202.
- Hinton, G., E. (2002): Training Products of Experts by Minimizing Contrastive Divergence *Neural Computation* **14** 1771-1800.
- Hinton, G., E., S. Osindero, and Y. Teh (2006): A Fast Learning Algorithm for Deep Belief Nets *Neural Computation* **18 (7)** 1527-1554.
- Hinton, G., E. (2007): Learning Multiple Layers of Representation *Trends in Cognitive Sciences* **11** 428-434.
- Hinton, G., E. (2009): Deep Belief Networks *Scholarpedia* **4 (5)** 5947.
- Hinton, G., E. (2010): A Practical Guide to Training Restricted Boltzmann Machines *Technical Report UTML TR 2010-003* **Department of Computer Science, University of Toronto**.
- Hochreiter, S. (1991): *Untersuchungen zu Dynamischen Neurolanen Netzen* Diploma Thesis **Institute fur Informatik, Technische Universitat Munchen**.
- Hochreiter, S., and J. Schmidhuber (1997): Long Short-Term Memory *Neural Computation* **9 (8)** 1735-1780.
- Hochreiter, S., Y. Bengio, P. Frasconi, and J. Schmidhuber (2001): Gradient Flow in Recurrent Nets: The Difficulty od Learning Long-Term Dependencies, in: *A Field Guide to Dynamic Recurrent Neural Networks: S. C. Kremer, and J. F. Kolen (eds.)* **IEEE Press**.
- LeCun, Y., B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel (1989): Back-propagation Applied to Handwritten Zip Code Recognition *Neural Computation* **1 (4)** 541-551.
- LeCun, Y., L. Bottou, Y. Bengio, and P. Haffner (1998): Gradient-based Learning applied to Document Recognition *Proceedings of the IEEE* **86 (11)** 2278-2324.

- La Rochelle, H., D. Erhan, A. Courville, J. Bergstra, and Y Bengio (2007): An Empirical Evaluation of Deep Architectures on Problems with many Factors of Variation *Proceedings of the 24th International Conference on Machine Learning* 473-480.
- Markoff, J. (2012): How Many Computers to Identify a Cat? 16,000.
- Martines, H., Y. Bengio, and G. N. Yannakakis (2013): Learning Deep Psychological Models of the Affect *IEEE Computational Intelligence* **8 (2)** 20.
- Mikolov, T., M. Karafiat, L. Burget, J. Cernocky, and S. Khudanpur (2010): Recurrent Neural Network Based Language Model *Interspeech*.
- MNIST (2013): THE MNIST DATABASE of handwritten digits.
- Ng, A., and J. Dean (2012): Building High-level Features Using Large Scale Unsupervised Learning **arXiV**.
- Quartz, S. R., and T. J. Sejnowski (1997): The Neural Basis for Cognitive Development: A Constructivist Manifesto *Behavioral and Brain Sciences* **20 (4)** 537-556.
- Raina, R., A. Madhavan, and A. Ng (2009): Large Scale Deep Unsupervised Learning Using Graphics Processors *Proceedings of the 26th International Conference on Machine Learning*.
- Riesenhuber, M., and T. Poggio (1999): Hierarchical Models of Object Recognition in Cortex *Nature Neuroscience* **11** 1019-1025.
- Sainath, T. N., A. Mohamed, B. Kingsbury, and B. Ramabhadran (2013): Deep Convolutional Networks for LVCSR *Proceedings ICASSP*.
- Schmidhuber, J. (1992): Learning Complex, Extended Sequences Using the Principle of History Compression *Neural Computation* **4** 234-242.
- Schmidhuber, J. (2013): My First Deep Learning System of 1991 + Deep Learning Timeline 1962-2013 **arXiV**.
- Shrager, J., and M. H. Johnson (1995): Timing in the Development of the Cortical Function: A Computational Approach, in: *B. Julesz and I. Kovacs (eds.): Maturational Windows and Adult Cortical Plasticity*.
- Shrager, J., and M. H. Johnson (1996): Dynamical Plasticity influences the Emergence of Function in a Simple Cortical Array *Neural Networks* **9 (7)** 1119-1129.
- Smolensky, P. (1986): Information Processing in Dynamical System: Foundations of Harmony Theory, in: *Explorations of the Microstructure of Cognition: D. E. Rumelhart, J. L. McClelland, and the PDP Research Group, Parallel Distributed Processing* **1** 194-281.

- TIMIT (1993): Acoustic-Phonetic Continuous Speech Corpus *Linguistic Data Consortium* Philadelphia.
- Utgoff, P. E., D. A. Stracuzzi (2002): Many Layered Learning *Neural Computation* **14** 2497-2529.
- Werbos, P. (1974): *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences* PhD Thesis **Harvard University**.

Hierarchical Clustering

1. Definition: Hierarchical Clustering is a Method of Cluster Analysis that seeks to build a hierarchy of clusters (Hierarchical Clustering (Wiki)). Strategies for hierarchical clustering fall into 2 categories:
    a. Agglomerative Clustering => This is a bottoms-up approach; each element starts in its own cluster, and pairs of clusters are merged as one moves up the hierarchy.
    b. Divisive Clustering => This is a top down approach; all observations start in one cluster, and splits are done recursively as one moves down the hierarchy.
2. Complexity: General agglomeration complexity is $\sim O(n^3)$, making it too slow for large data sets. Divisive Clustering is worse, consuming $\sim O(2^n)$ for an exhaustive search. In certain special cases optimal agglomeration of $\sim O(n^2)$ is known, e.g., SLINK (Sibson (1973)) for single linkage clustering and CLINK (Defays (1977)) for complete linkage clustering.
3. Clustering Dissimilarity Metrics: In order to decide which clusters need to be combined (for agglomerative clustering) or where a cluster needs to be split (for divisive clustering), a measure of dissimilarity between the sets of observations is required. In most methods of hierarchical clustering this is achieved by using an appropriate metric (via a measure of distance between a pair of observations), and a linkage criterion that specifies dissimilarity of sets as a function of pairwise distances of observations between the sets. Some common distance metrics are given in DISTANCE (2009).
    a. Euclidean Distance =>

$$\|a - b\|_2 = \sqrt{\sum_i (a_i - b_i)^2}$$

    b. Squared Euclidean Distance =>

$$\|a-b\|_2{}^2 = \sum_i (a_i - b_i)^2$$

c. Manhattan Distance =>

$$\|a-b\| = \sum_i |a_i - b_i|$$

d. Maximum Distance =>

$$\|a-b\|_\infty = \max_i (|a_i - b_i|)$$

e. Mahalanobis Distance =>

$$\sqrt{(a-b)S(a-b)^T}$$

where S is the covariance matrix.

f. Cosine Similarity =>

$$\frac{a.b}{\|a\|\|b\|}$$

4. Metrics for Text/Non-Numeric Data: In this case metrics such as the Hamming distance or the Levenshtein distance or used.
5. Clustering Dissimilarity Linkage Criterion: This determines the distance between sets of observations as a function of the pairwise distances between the observations. Some commonly used linkage criterion are (Szekely and Rizzo (2005), CLUSTER (2009)):
    a. Maximum or Complete Linkage Clustering =>

$$max\{d(a,b): a \in A, b \in B\}$$

b. Minimum or Single Linkage Clustering =>

$$min\{d(a,b): a \in A, b \in B\}$$

c. Mean or Average Linkage Clustering (UPGMA) =>

$$\frac{1}{|A||B|}\sum_{a\in A}\sum_{b\in B} d(a,b)$$

d. Minimum Energy Clustering =>

$$\frac{2}{nm}\sum_{i,j=1}^{n,m}\left\|a_i - b_j\right\|_2 - \frac{1}{n^2}\sum_{i,j=1}^{n}\left\|a_i - a_j\right\|_2 - \frac{1}{m^2}\sum_{i,j=1}^{m}\left\|b_i - b_j\right\|_2$$

6. Other Linkage Criterion:
   a. Sum of all inter-cluster Variance
   b. Decrease in Variance for the Cluster being Merged – the Ward's Criterion (Ward (1963))
   c. Probability that the candidate clusters spawn from the same distribution function (V-linkage)
   d. The product of the in-degree and the out-degree in the k-nearest-neighbor graph (graph degree linkage (Zhang, Wang, Zhao, and Tang (2012))
   e. The increment of some Cluster descriptor (i.e., the metric defined for some measurement of the quality of the cluster) after the merge/separation (Ma, Derksen, Hing, and Wright (2007), Zhao and Tang (2008), Zhang, Zhao, and Wang (2013)).

References

- CLUSTER (2009): The CLUSTER Procedure: Clustering Methods *SAS/STAT 9.2 Users Guide* **SAS Institute**.
- Defays, D. (1977): An Efficient Algorithm for a Complete-Link Method *Computer Journal* **20 (4)** 364-366.
- DISTANCE (2009): The DISTANCE Procedure: Proximity Measures *SAS/STAT 9.2 Users Guide* **SAS Institute**.
- Hierarchical Clustering (Wiki): Wikipedia Entry for Hierarchical Clustering.
- Ma, Y., H. Derksen, W. Hong, and J. Wright (2007): Segmentation of Multivariate Mixed Data via Lossy Data Coding and Compression *IEEE Transactions on Pattern Analysis and Machine Intelligence* **29 (9)** 1546-1562.
- Sibson, R. (1973): SLINK: An Optimally Efficient Algorithm for the Single-Link Cluster Method *Computer Journal* **16 (1)** 30-34.
- Szekely, G. J., and M. L. Rizzo (2005): Hierarchical Clustering vis Joint Between-Within Distances: Extending Ward's Minimum Variance Method *Journal of Classification* **22** 151-183.
- Ward, J. H. (1963): Hierarchical Grouping to Optimize and Objective Function *Journal of the American Statistical Association* **58 (301)** 236-244.
- Zhang, W., X. Wang, D. Zhao, and X. Tang (2012): Graph Degree Linkage: Agglomerative Clustering on a Directed Graph **arXiV**.
- Zhang, W., D. Zhao, and X. Tang (2013): Agglomerative clustering via maximum incremental path integral *Pattern Recognition* **46 (11)** 3056-3065.
- Zhao, D., and X. Tang (2008): Cyclizing Clusters viz Zeta Function of a Graph *Advances in Neural Information Processing Systems*.

k-Means Clustering

Introduction

1. Definition: k-Means Clustering is a method of vector quantization that aims to partition n observations into k clusters in which each observatione belongs to the cluster with the nearest mean, serving as a prototype of the cluster.
2. k-Means Complexity: It is NP-hard – however there exist efficient heuristic algorithms that converge quickly to a local optimum. These algorithms are similar to EM on Gaussian mixture Distributions.
3. Differences between k-Means and EM Algorithms: While both k-means and the EM algorithms cluster centers to model the data, k-means finds clusters of comparable spatial extent, while EM allows clusters to have different shapes.
4. Development History: The standard algorithm (Steinhaus (1957), MacQueen (1967)) was originally used in Bell Labs for pulse code modulation (Lloyd (1957), Forgy (1965)). An efficient version of this was created later by Hartigan (1975) and Hartigan and Wong (1979).

Mathematical Formulation

1. Specification: Given a set of observations $\{\overrightarrow{x_i}\}_{i=1}^{n}$ where each observation is a d-dimensional real vector, k-means clustering aims to partition the n observations into k sets $(k < n)$ $\{\vec{S}\}_{j=1}^{k}$ so as to minimize the within-cluster sum of squares (WCSS):

$$\arg\min_{\vec{S}} \sum_{i=1}^{k} \sum_{x_j \in S_i} \left\| \overrightarrow{x_j} - \overrightarrow{\mu_i} \right\|^2$$

where $\overrightarrow{\mu_i}$ is the mean of the points in $\overrightarrow{S_i}$.

The Standard Algorithm

1. The Layout: This is also referred to as the Lloyd's algorithm. Given an initial set of k-means $m_1^{(1)}, \ldots, m_k^{(1)}$, the algorithm proceeds by alternating between 2 steps – the assignment step and the update step (MacKay (2003)).
2. The Assignment Step: Assign each contribution to the cluster whose assignment yields the least WCSS. Since the sum-of-the-squares is the Euclidean distance, this algorithm intuitively chooses the nearest mean. Mathematically, this partitions the observations according to the Voronoi diagram generated by the means

$$S_i^{(t)} = \left\{x_p : \left\|x_p - m_i^{(t)}\right\|^2 \leq \left\|x_p - m_j^{(t)}\right\|^2 \forall\, i, j; i \leq j \leq k\right\}$$

   x_p is assigned to exactly one $S_i^{(t)}$ (even if it is suitable to be assigned to more than to one).
3. The Update Step: Calculate the new means to be centroids of the observations in the new cluster

$$m_i^{(t+1)} = \frac{1}{\left|S_i^{(t)}\right|} \sum_{x_j \in S_i^{(t)}} x_j$$

   Since the arithmetic mean is a least squares estimator, this step also minimizes the WCSS objective.
4. k-Means Convergence: The algorithm is deemed to have converged when the assignments no longer change. Since both the steps optimize the WCSS objective, and there exist only a finite number of partitions, the algorithm must converge to a local optimum (no guarantees of the global optimum).

   a. Convergence to the local optimum may depend on the initial clusters, so it is common to run it multiple times with different starting conditions (esp. since the algorithm is very fast). In the worst case, it can be very slow; in particular, it has been that exist certain set of starting points, even in 2 dimensions, on which the k-means takes an exponential time, i.e., $2^{\Omega(n)}$ to converge (Vattani (2011)).

5. Comparison with EM Algorithm: The assignment step is also referred to the expectation step, while the update step is referred to as the maximization step, making the k-means a variant of the generalized EM algorithm.
6. The Distance Metric: Given that the WCSS minimization occurs in both the steps, naively using other distance metrics (such as non-Euclidean metrics) will cause the algorithm to not converge. Various modifications to k-means such as spherical k-means and k-medoids have been proposed to allow for using alternate distance metrics.
7. Drawbacks:
   a. Euclidean distance is used as the distance metric and variance is used as a measure of the cluster scatter (i.e., extraneously specified distribution scatter measures such as higher moments etc. are not specifically accommodated)
   b. The number of clusters k is an input parameter, and an inappropriate choice of k may lead to poor results. This makes it important to run diagnostic checks for determining the appropriate k.
   c. Convergence to a local minimum may produce counter-intuitive/wrong results.
8. Performance: Use of k-means has been successfully combined with simple, linear classifiers for semi-supervised learning in NLP (specifically for named entity recognition) (Lin and Wu (2009)), and in computer vision. Object recognition tests indicate that k-means has comparable performance with more sophisticated feature learners such as auto-encoders and RBMs (Coates, Lee, and Ng (2011)). However, it requires more data than for these methods to produce equivalent performance, since each data point only contributes to feature than multiple (Coates and Ng (2012)).

k-Means Initialization Schemes

1. The Standard Schemes: Commonly used initialization methods are the Forgy method and the Random Partition Method (Hamerley and Elkan (2002)). The Forgy method randomly chooses k observations from the data set, and uses them as the initial means. The random partition method first randomly assigns a cluster to each observation and then proceeds to the update step, thus computing the initial means to be centroid of the randomly computed points.
2. Comparison of Forgy and Random Partition: In effect, the Forgy method tends to spread the initial means out, while the random partition method places all of them close to the center of the data set. The random partition method is preferred to the extensions of the standard k-means such as k-harmonic means or fuzzy k-means. For standard k-means and EM, the Forgy method is preferred.

k-Means Complexity

1. Generics: In general, finding the optimal solution to k-means clustering problems in d dimensions is:
    a. NP-Hard in the general Euclidean space even for 2 clusters (Aloise, Deshpande, Hansen, and Popat (2009), Dasgupta and Freund (2009))
    b. NP-Hard for k clusters even in a single plane (Mahajan, Nimbhorkar, and Varadarajan (2009))
    c. If k and d are fixed, the time for the exact solution is $\sim O(n^{dk+1} \log n)$ where n is the number of entries to be clustered (Inaba, Katoh, and Imai (1994))
    d. It is important to recall that all of these are heuristic algorithms that seek out a local optimum
2. Lloyd's k-mean Complexity: Lloyd's k-means algorithm has polynomial smoothed running time. It is shown (Arthur, Manthey, and Roeglin (2009)) that for an arbitrary set of n points in $[0, 1]^d$, if each point is independently perturbed with a mean θ and a variance σ^2, the expected running time of k-means is bounded by $O\left(\frac{n^{34}k^{34}d^8 log^4(n)}{\sigma^2}\right)$, which is a polynomial in n, k, d and $\frac{1}{\sigma^2}$.

3. Other k-means Complexity Estimates: Estimates for other specific k-means configurations have been worked out, for e.g., Arthur and Bhowmick (2009) have shown that the running time for k-means on n points in the lattice $\{1, \ldots, M\}^d$ is bounded by $O(dn^4M^2)$.

k-Means Variations

1. k-Medians Clustering: k-medians clustering uses the median in each dimension instead of the means, and in this way minimizes the L_1 norm (the Taxicab Geometry).
2. k-Medoids Clustering: Also called Partitioning Around Medoids (PAM), this uses the medoids instead of the means, and in this way minimizes the sum of distances for arbitrary distance functions.
3. Fuzzy C-Means Clustering: This is a softer version of the k-means, where each entry has a "degree" of belonging to each cluster.
4. Gaussian Mixtures: Clustering done with Gaussian mixture models trained using EM maintains the probabilistic assignments to clusters (instead of the deterministic assignments) along with multivariate distributions instead of a single set of means.
5. Starting Clusters: Several methods have been proposed to choose better starting clusters (e.g., k-means++).
6. Speeding up Iteration Steps: Kanungo, Mount, Netanyahu, Piatko, Silverman, and Wu (2002) propose a filtering algorithm that uses kd-trees to speed up each k-means step. Other methods to speed up k-means use the coresets (Frahling and Sohler (2006)) or the triangle inequality (Elkan (2003)).
7. Escaping Local Maxima: Escape local maxima by swapping points between the clusters (Hartigan and Wong (1979)).
8. Spherical k-means: The spherical k-means clustering is suitable for directional data (Dhillon and Modha (2001)).
9. Minkowski Weighting: The Minkowski weighted k-means deals with each irrelevant feature by assigning cluster specific weights to each feature (Amorim and Mirkin (2012)).

k-Means Applications

1. Application Range: Given that heuristic algorithms such as Lloyd's are easy to implement even on large data sets, k-means is used widely in problems ranging from market segmentation, computer vision, geostatistics (Honarkhah and Caers (2010)), astronomy, agriculture, etc. It is often used as a pre-processing step for other algorithms, for example, to find a starting configuration.
2. Vector Quantization: In computer graphics color quantization is the task of reducing the color palette to a fixed number of colors k. The k-means algorithm can be used for such quantization tasks and produces very competitive results. Other uses of vector quantization include non-random sampling, as k-means can be easily used to choose k different but prototypical objects from a large data set (possibly for further analysis).
3. Feature Learning: k-means clustering has been used as a feature learning (or dictionary learning) step in semi-supervised or unsupervised learning (Coates and Ng (2012)). The basic approach is to train a k-means clustering representation using the input training data that need not be labeled. Then, to project the input datum into the new feature space, we have a choice of encoding functions such as:
    a. Thresholded matrix-product of the datum with centroid locations
    b. Distance from the datum to each centroid
    c. Indicator function for the nearest centroid (Csurka, Dance, Fan, Willamowski, and Bray (2004), Coates and Ng (2012))
    d. Smooth transformation of the computed "distance" (Coates, Lee, and Ng (2011))
    e. Alternatively, by transforming the sample-cluster distance through a Gaussian RBF, one obtains the hidden layer of a radial basis function network (Schwenker, Kestler, and Palm (2001))

Alternate k-Means Formulations

1. Generalized k-Means: k-means clustering, and its associated EM algorithm, is a special case of Gaussian mixture model – specifically, where the limit of covariances are diagonal, equal,

and small (Press, Teukolsky, Vetterling, and Flannery (2007)). Another generalization is the k-SVD algorithm at estmates the data points as sparse linear combination of "codebook" vectors. k-means corresponds to the special case of using a single codebook vector with unit weight (Aharon, Elad, and Bruckstein (2006)).

2. Mean-Shift Clustering: The basic mean-shift clustering maintains a set of points the same size as the input data set. Initially, this set is copied from the input set. This set is then iteratively replaced by the mean of those points in the set that are within a given distance of that point. By contrast, k-means restricts this updated set to k points usually much less than the number of points in the input data set, and replaces each point in this set by the means of all points in the INPUT SET that are closer to the point than any other (e.g., within the Voronoi partition of each updating point).
    a. Variation => A mean shift algorithm that is more similar to k-means (called likelihood mean shift) replaces the set of points undergoing replacement by the mean of all points in the input set that are within a given distance of the changing set (Little and Jones (2011)).
    b. Properties => One of the advantages of mean shift clustering over k-means is that there is no need to choose the number of clusters, since mean shift is likely to find only a few clusters if only a small number exist. However, mean shift clustering can be slower than k-means, and still requires the selection of a bandwidth parameter. Mean shift also has soft variants like k-means.
3. PCA vs. k-means: It has been asserted (Zha, Ding, Gu, He, and Simon (2001), Ding and He (2004)) that the relaxed solution to k-means clustering, specified by the cluster indicators, is given by the PCA's principal components, and the PCA sub-space spanned by the principal directions is identical to the cluster centroid subspace. However, PCA being a useful relaxation of k-means is not a new result (Drineas, Frieze, Kannan, Vempala, and Vinay (2004)); further, it is straightforward to uncover counter-examples to the claim that the cluster centroid subspace is spanned by the principal components.
4. Bilateral Filtering: k-means implicitly assumes that the ordering of the inputs does not matter. The bilateral filter is similar to k-means and mean shift in that it maintains a set of data points that are iteratively replaced by their means. However, bilateral filtering restricts the calculation of the kernel-weighted mean to include only those points that are close in the

ordering of the input data (Little and Jones (2011)). This makes it applicable to problems such as image de-noising where spatial arrangements of pixels in an image is of critical importance.

5. Other similar Problem Spaces: The set of squared error minimizing cluster functions includes the *k*-medoid algorithm, an approach that forces the center point of each cluster to be one of the actual points, i.e., it uses medoids in place of centroids.

References

- Aharon, M., M. Elad, and A. Bruckstein (2006): k-SVD: An Algorithm for Designing Over-complete Dictionaries for Sparse Representation *IEEE Transactions on Signal Processing* **54 (11)** 4311-4322.
- Aloise, D., A. Deshpande, P. Hanson, and P. Popat (2009): NP-Hardness of Euclidean Sum-of-Squares Clustering *Machine Learning* **75** 245-249.
- Amorim, R. C., and B. Mirkin (2012): Minkowski, Metric, Feature Weighting, And Anomalous Cluster Initializing using K-Means Clustering *Pattern Recognition* **45 (3)** 1061-1075.
- Arthur, D., A. Bhowmick (2009): A Theoretical Analysis of the Lloyd's Algorithm for k-Means Clustering.
- Arthur, D., B. Manthey, and H. Roeglin (2009): *k*-Means has Polynomial Smoothed Complexity *Proceedings of the 50th Symposium of the Foundations of Computer Science (FOCS).*
- Coates, A., H. Lee, and A. Y. Ng (2011): An Analysis of Single-Layer Networks in Unsupervised Feature Learning *International Conference on Artificial Intelligence and Statistics (AISTATS).*
- Coates, A., and A. Y. Ng (2012): Learning Feature Representations with k-Means, in: Montavon, Orr, and Muller (eds.) *Neural Networks: Tricks of the Trade* **Springer**.
- Csurka, G., C. C. Dance, L. Fan, J. Willamowski, and C. Bray (2004): Visual Categorizations with Bags of Key Points *ECCV Workshop on Statistical Learning in Computer Vision*.

- Dasgupta, S., and Y. Freund (2009): Random Projection Trees for Vector Quantization *IEEE Transactions On* **55** 3229-3242.
- Dhillon, I. S., and D. M. Modha (2001): Concept Decomposition for Large Sparse Text Data using Clustering *Machine Learning* **42 (1)** 143-175.
- Ding, C., and X. He (2004): k-Means Clustering via Principal Component Analysis *Proceedings on the International Conference on Machine Learning (ICML 2004)* 225-232.
- Elkan, C. (2003): Using the Triangle Inequality to Accelerate *k*-Means *Proceedings of the 20th International Conference on Machine Learning (ICML)*.
- Forgy, E. W. (1965): Cluster Analysis of Multi-variate Data: Efficiency versus Interpretability of Classifications *Biometrics* **21** 768-769.
- Frahling, G., and C. Sohler (2006): A Fast *k*-Means Implementation using Coresets *Proceedings of the 22nd Annual Symposium on Computational Geometry (SoCG)*.
- Hamerley, G., and C. Elkan (2002): Alternatives to the k-means Algorithms that find better Clusterings *Proceedings of the 11th International Conference on Information and Knowledge Management*.
- Hartigan, J. A. (1975): *Clustering Algorithms* **John Wiley & Sons**.
- Hartigan, J. A., and M. A. Wong (1979): Algorithm AS 136: A k-Means Clustering Algorithm *Journal of the Royal Statistical Society* **C28 (1)** 100-108.
- Honarkhah, M., and J. Caers (2010): Stochastic Simulation of Patterns Using Distance-Based Pattern Modeling *Mathematical Geosciences* **42** 487-517.
- Inaba, N., M. Katoh, and J. Imai (1994): Applications of Weighted Voronoi Diagrams and Randomization to Variance-Based k-Clustering *Proceedings of the 10th ACM Symposium on Computational Geometry* 332-339.
- Kanungo, T., D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu (2002): An Efficient k-Means Clustering Algorithm: Analysis and Implementation *IEEE Transaction on Pattern Recognition and Machine Intelligence* **24** 881-892.
- Lin, D., and X. Wu (2009): Phrase Clustering for Discriminative Learning *Annual Meeting of the ACL and IJCNLP* 1030-1038.
- Little, M. A., and N. S. Jones (2011): Generalized Methods and Solvers for Piece-wise Constant Signals: Part I *Proceedings of the Royal Society A*.

- Lloyd, S. (1982): Least Squares Quantization in PCM *IEEE Transactions in Information Theory* **28 (2)** 129-137.
- MacKay, C. (2003): *Information Theory, Inference, and Learning Algorithms* **Cambridge University Press**.
- MacQueen, J. B. (1967): Some Methods for Classification and Analysis of Multi-variate Observations *Proceedings of the 5th Berkeley Symposium of Mathematical Sciences and Probability* **University of California Press** 281-297.
- Mahajan, M., P. Nimbhorkar, and K. Varadarajan (2009): The Planar k-Means Problem is NP-Hard *Lecture Notes in Computer Science* **5431** 274-285.
- Press, A., S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery (2007): *Numerical Recipes – The Art of Scientific Computing 3rd Edition* **Cambridge University Press**, New York.
- Schwenker, F., H. A. Kestler, and G. Palm (2001): Three Learning Phases for Radial-Basis-Function Networks *Neural Networks* **14** 439-458.
- Steinhaus, H. (1957): Sur le Division des Corps Materiels en Parties *Bull. Acad. Polon. Sci.* **4 (12)** 801-804.
- Zha, H., C. Ding, M. Gu, X. He, and H. D. Simon (2001): Spectral Relaxation for K-Means Clustering *Neural Information Processing Systems (NIPS 2001)* **14** 1057-1064.

Correlation Clustering

1. Definition: Correlation Clustering provides a method for clustering a set of objects into the optimum number of clusters without specifying that number in advance (Becker (2005)).
2. Problem Description: In machine learning correlation clustering or cluster editing operates in a scenario where the representation of the relationships between the objects is what is known (in place of their actual representations). Given a signed graph

$$G = (V, E)$$

   where the edge labels indicate similarity/dissimilarity $(+/-)$ of the nodes, the task is to cluster the vertices so that similar objects are grouped together. Unlike other clustering algorithms, this does not require choosing the number of clusters in advance because the objective (which is to minimize the net disagreements) is independent of the number of clusters.
3. Imperfect Correlation Clustering: It may not be possible to find a perfect clustering where all the similar items are in one cluster, and the dissimilar ones are in different clusters. The problem of maximizing the agreements is NP-complete – the multi-way problem reduces to maximizing the weighted agreements, and the problem of partitioning into triangles can be reduced to the unweighted one (Garey and Johnson (2000)).
4. Constant Factor Approximation Algorithm: Bansal, Blum, and Chawla (2004) discuss the NP-completeness proof and present constant factor algorithm and polynomial time approximation scheme to find the clusters in this setting. Ailon, Charikar, and Newman (2005) propose a randomized 3-approximation algorithm for the above (Correlation Clustering (Wiki) contains the pseudo-code for this).
5. Optimal Number of Clusters Evaluation: Optimization of the correlation clustering functional is closely related to the well-known discrete optimization methods (Bagon and Galun (2011)). Bagon and Galun (2011)) also propose a probabilistic solution of the underlying

implicit model that allows the correlation clustering to evaluate the underlying number of clusters. The analysis suggests that the functional assumes a uniform prior over all partitions regardless of the number of clusters – thus, a uniform prior over the number of clusters emerges.

   a. Optimal Number of Clusters under specific scenarios => Bagon and Galun (2011) propose several discrete optimization algorithms that scale gracefully with the number of elements (which can be over 100,000 variables for certain applications). They also evaluate the effectiveness of the recovery of the underlying number of clusters in a number of applications.
   b. They also evaluate the effectiveness of the recovery of the underlying number of clusters in a number of applications.

6. Feature Vector Correlation: In certain scenarios in data mining, correlations among feature vectors in high-dimensional space guide the clustering process. Since these correlations can be different in different clusters, a global de-correlation cannot reduce this to typical uncorrelated clustering.
   a. Feature Vector Correlation Clustering => Correlation among the subsets of attributes results in different shapes of clusters. Therefore, the similarity between cluster objects needs to be defined by taking into account local correlation patterns. Thus, different types of correlation clustering (esp. high-dimensional clustering) and their relationships to standard correlation clustering techniques have been discussed in Bohm, Kailing, Kroger, and Zimek (2004), Zimek (2008), and Kriegel, Kroger, and Zimek (2009).

References

- Ailon, N., M. Charikar, and A. Newman (2005): Aggregating inconsistent Information: Ranking and Clustering *STOC '05: Proceedings of the 37th Annual ACM Symposium on Theory of Computing* 684-693.
- Bagon, S., and M. Galun (2011): Large Scale Correlation Clustering Optimization **arXiV**.

- Bansal, N., A. Bloom, and S. Chawla (2004): Correlation Clustering *Machine Learning Journal (Special Issue on Theoretical Advances in Data Clustering)* 86-113.
- Becker, H. (2005): A Survey of Correlation Clustering.
- Bohm, C., K. Kailing, P. Kroger, and A. Zimek (2004): Computing Clusters of Correlation Connected Objects *Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data – SIGMOD '04* 455-467.
- Correlation Clustering (Wiki): Wikipedia Entry for Correlation Clustering.
- Kriegel, H. P., P. Kroger, and A. Zimek (2009): Clustering High-Dimensional Data: A Survey on Subspace Clustering, Pattern-based Clustering, and Correlation-Clustering *ACM Transactions on Knowledge Discovery from Data (TKDD)* **1 (3)** 1-58.
- Zimek, A. (2008): Correlation Clustering.

Kernel Principal Component Analysis (Kernel PCA)

1. Definition: Kernel PCA is an extension of the standard PCA using the techniques of the kernel methods. With kernel PCA, the original linear operations of PCA are done in a reproducing kernel Hilbert space with a non-linear mapping (Kernel PCA (Wiki)).
2. Motivation: In general, a set of N data points cannot be separated if

$$d < N$$

They can almost always be separated in

$$d \geq N$$

dimensions. That is, given N points $\{\vec{x_i}\}$, if we map them to an N-dimensional space with $\Phi(\vec{x_i})$ where

$$\Phi: R^d \to R^N$$

it is easy to construct a hyperplane that divides the points into arbitrary clusters. This Φ creates a linearly independent set of vectors, so there is no covariance on which to perform eigen-decomposition explicitly as we would in linear PCA (Scholkopf, Smola, and Muller (1996)).
3. The "Kernel" Trick: In kernel PCA a non-trivial, arbitrary function Φ is "chosen" but never calculated explicitly, allowing for the possibility to use very high dimensional Φ's if we never have to actually evaluate the data in that space. Since we generally try to avoid working in the Φ-space (we call this the "feature space"), we create the $N \times N$ kernel

$$K = k(x, y) = [\Phi(x), \Phi(y)] = \Phi(x)\Phi^T(y)$$

which represents the inner product space (i.e., the Gramian matrix) of an otherwise intractable feature space.

4. Feature Space Projections: Since we never actually work in the feature space, we do not need to compute its covariances and eigenvectors. This also implies that we do not explicitly compute the principal components themselves, but only compute the projections of our data onto these components. To evaluate the projection from a point in the feature space $\Phi(\vec{x})$ onto the k^{th} principal component, we compute it using

$$V_k^T \Phi(\vec{x}) = \left[\sum_{i=1}^{N} a_{i,k} \Phi(\vec{x_i})\right] \Phi(\vec{x})$$

5. Computation of the Kernel Coefficients: $\Phi(\vec{x_i})\Phi^T(\vec{x})$ is just the dot product, i.e., the elements of the K matrix. Eigenization of $\Phi(\vec{x_i})\Phi^T(\vec{x})$ and the eventual normalization results in $a_{i,k}$.
6. Feature Space Mean Centering: Care must be taken regarding the fact that, whether or not $\vec{x}$ has zero-mean in the original space, it is not guaranteed to be centered in the feature-space (which we never compute explicitly anyway). Since centering is required for PCA, we centralize $K \rightarrow K'$ using

$$K' = K - 1_N K - K 1_N + 1_N K 1_N$$

where 1_N denotes a $N \times N$ matrix for which each element takes the value $\frac{1}{N}$. K' is used for the PCA above.

7. Kernel PCA Ranking: In linear PCA we use the eigenvalues to rank the eigenvectors based on how much variance is captured by each principal component on the given set of inputs points – this is used, amongst other things, for dimensionality reduction. No such ranking exists in kernel PCA, however.
8. Kernel PCA on Large Datasets: In practice, a large dataset leads to a large K, and storing those K components can become a problem. One way to deal with this is to cluster the large

dataset, and replace the kernel with the means of the clusters. Since even this method may yield a large K, it is common to compute the top P eigen-vectors and eigen-values of K (bearing in mind that these do NOT necessarily translate into the corresponding maximal variance explainers of the original points).

9. Applications: Kernel PCA has been demonstrated to be useful for novelty reduction (Hoffmann (2007)) and image de-noising (Mika, Scholkopf, Smola, Muller, Scholz, and Ratsch (1999)).

References

- Hoffmann, H. (2007): Kernel PCA for Novelty Detection *Pattern Recognition* **40** 863-874.
- Mika, S., B. Scholkopf, A. Smola, K. R. Muller, M. Scholz, and G. Ratsch (1999): Kernel PCA and De-noising in Feature Spaces *Proceedings of the 1998 Conference on Advances in Neural Information Processing Systems II* 536-542.
- Scholkopf, B., A. Smola, and K. R. Muller (1996): Nonlinear Component Analysis as a Kernel Eigenvalue Problem.

Ensemble Learning

Introduction

1. Definition: In statistics and machine learning ensemble methods use multiple learning algorithms to obtain better predictive performance than could be obtained from any one of the constituent learning algorithms (Opitz and Maclin (1999), Polikar (2006), Rokach (2010)).
2. Differences with Statistical Mechanics: Unlike a statistical ensemble in statistical mechanics, which is usually infinite, a machine learning ensemble refers only to a concrete set of alternative models, but typically allows for much more flexible structure to exist between these alternatives.

Overview

1. Weak Learners vs. Strong Learners: Ensembles combine multiple hypotheses to form a better hypothesis – thereby presenting a technique for combining many weak learners to form a strong learner.
2. Ensembles vs. Multiple Classifier Systems: The term ***ensemble*** is reserved here for methods that generate multiple hypotheses using the same base learner. The broader term ***multiple classification systems*** also covers hybridization of the hypotheses not induced by the same base learner.
3. Motivation for Usage in Supervised Learning: Supervised learning algorithms search through a hypothesis space to locate a suitable hypothesis that has the appropriate prediction capabilities. Even if the hypothesis space contains hypotheses that are well-suited for a particular problem, it may be difficult to find a good (i.e., well-balanced) one.

4. Performance Advantages: Evaluating the performance of an ensemble typically requires more computation that evaluating the performance of a single model, so ensembles may be thought of as a way to compensate for poor learning algorithms at the expense of more computation. Fast algorithms such as decision trees are commonly used with ensembles (e.g., random forest), although slower algorithms can benefit from ensemble techniques as well.

Theoretical Underpinnings

1. Ensemble Hypotheses Space: Since an ensemble itself can be trained for predictions, it is a supervised algorithm in itself. However, as a trained ensemble, it represents a single hypothesis. The ensemble hypothesis space, therefore, is not necessarily contained within the space of the models from which it is built.
2. Ensemble Predictive Representation Flexibility: The above-mentioned representation ability results in greater representation flexibility, at the cost of over-fitting the training data more than that for a single model. In practice, certain ensemble techniques (esp. bagging) tend to reduce problems related to over-fitting of the training data.
3. Predictor Diversity: Empirically, ensembles tend to yield better results when there is significant diversity among the models (Sollich and Krogh (1996), Kuncheva and Whitaker (2003)). Many ensemble methods therefore seek to promote diversity among the models they combine (Brown, Wyatt, Harris, and Yao (2005), Garcia-Adeva, Cervino, and Calvo (2005)). More random algorithms such as random decision trees can be used to produce a stronger ensemble than very deliberate algorithms such as entropy-reducing decision trees (Ho (1995)).
4. Strong Learners: Using a variety of strong learning algorithms, however, has been shown to be more useful than using techniques that “dumb down” the models to promote diversity (Gashler, Giraud-Carrier, and Martinez (2008)).

Ensemble Aggregator Types

1. Type 1: This is based purely on Bayes' Optimal Classification (see below). Aggregation here occurs exclusively across the hypothesis ensemble space, and training set remains fixed between classification runs. Examples of this type are Ensemble Averaging Techniques, BMA, BMC, certain BOM variants, and stacking.
2. Type 2: These are based off of cross-validation philosophy, and therefore involve generating subsets of re-samples and their eventual re-combination. Aggregation occurs purely across the re-sampling space and not the hypotheses space (often there will only be a single model). Examples of this type include bagging and boosting.
3. Type 3: This type is an extension of the two types above, and the expectation is that the aggregation should occur across both the hypotheses space as well as the re-sample set according to

$$y = \underset{c_j \in C}{\arg max} \sum_{T_k \in T} \sum_{h_i \in H} P(c_j|h_i) P(T_k|h_i) P(h_i)$$

   The cross-validation based variants of BOM (e.g., gating based cross-validation selectors) could be constructed by applying the above.

Bayes' Optimal Classifier

1. Definition: This refers to an ensemble that contains all the hypotheses in the hypothesis space. On average (and over sufficient sample sizes) no other ensemble can outperform it (by construction), since this is the ideal ensemble that others attempt to mirror (Mitchell (1997)).
2. Conception Philosophy: Each hypothesis is given a vote proportional to the likelihood that the training data set would be sampled from the system if that corresponding hypothesis were true. This setup facilitates evaluation under the world where the hypothesis is valid. To accommodate training data set of finite size, the vote of each hypothesis is also multiplied by the prior probabilities of that hypothesis.
3. Formulation:

$$y = \underset{c_j \in C}{\arg max} \sum_i P(c_j|h_i)P(T|h_i)P(h_i)$$

where y is the predicted class, c_j is the class instance j that is part of the bigger set of classes C, h_i is the hypothesis i that is part of the hypothesis universe H; T is the set of all observations; $P(c_j|h_i)$ and $P(T|h_i)$ are the probabilities of realization of the class c_j and the observation set T given hypothesis h_i; and $P(h_i)$ is the probability of realization of the hypothesis h_i.

4. Hypothesis Space Optimality: The hypothesis represented by the Bayes Optimal Classifier is the optimal hypothesis in the ensemble space (the space of all possible ensembles consisting only of those hypotheses in H).
5. Implementation Challenges:
    a. Most interesting hypotheses space can be too large to iterate over, as required by $\arg max$
    b. Many hypotheses yield only a predicted class rather than the probability term $P(c_j|h_i)$
    c. Computing the unbiased estimate of the probability of training set given hypothesis (i.e., $P(T|h_i)$) is rarely feasible
    d. Estimating $P(h_i)$ is rarely possible

Bagging and Boosting

1. Boostrap Aggregating (Boosting): Bagging involves each model in the ensemble vote with equal weight (this could be $P(T|h_i)$ or $P(h_i)$ or both – i.e., a constant independent of h_i).
2. Model Diversity in Bagging: In order to promote model variance, bagging trains each model in the ensemble using a randomly drawn subset of the training data. As an example, random forest trees combine random decision trees with bagging to achieve very high classification accuracy (Breiman (1996)). Sahu, Runger, and Apley (2001) provide an interesting application of bagging in the unsupervised learning context.

3. Boosting: Boosting involves incrementally building an ensemble by training each new model instance to emphasize the training instances that the previous model set mis-classified. In some cases, boosting has been shown to yield better accuracy than bagging – however, it also contains a tendency to over-fit. The most common boosting implementation is AdaBoost, although some newer algorithms are reported to achieve better results.

Bayesian Model Averaging (BMA)

1. Philosophy: BMA is an ensemble technique that seeks to approximate Bayes' Optimal Classifier by sampling hypotheses from the hypothesis space, and combining them using Bayes' law (Hoeting, Madigan, Raftery, and Volinksy (1999)).
2. Methodology: Unlike the Bayes' Optimal Classifier, BMA implementation is practically achieved. Hypotheses are usually sampled (i.e., $P(h_i)$ determined) using a Monte Carlo sampling technique such as MCMC. Further, Gibbs' sampling may be used to draw hypothesis that is consistent with the distribution (i.e., $P(T|h_i)$). Wiki (Ensemble Learning) contains illustrative pseudo-code.
3. Asymptotic Error Rate: It has been shown that under certain circumstances, when the hypotheses are drawn according to BMA and averaged in accordance with Bayes' law, BMA has an error rate that is bounded to be within twice the error rate of the Bayes' Optimal Classifier (Haussler, Kearns, and Schapire (1994)).
4. Caveats: Despite the empirical correctness, BMA has been found empirically to promote more over-fitting in comparison to simpler techniques such as bagging (Domingos (2000)), thereby motivating the development of techniques such as Bayesian Model Combination (Minka (2002)).
5. Conceptual Shortcoming of BMA: As seen earlier, the use of Bayes' law to compute model weights necessitates the computation of each $P(T|h_i)$. Typically, none of the models in the ensemble is the precise model from the training data is generated, so all of them correctly receive a value close to zero for this term ($P(T|h_i)$). Thus, in principle BMA would work well if the hypotheses ensemble set was wide enough to span the entire model space, but that is rarely possible. Consequently, each pattern in the training data causes the ensemble weight

to shift towards the model in the ensemble that is closes to the distribution of the training data.

6. Amelioration of this BMA Shortcoming: The possible weightings for an ensemble can be visualized as lying along the vertices of a simplex. In BMA, at each vertex of the simplex, all of the weights are given to the single corresponding model of the ensemble. BMA converges toward the vertex that is closest to the training data – this is what can be changed.

Bayesian Model Combination (BMC)

1. BMC as an Enhancement to BMA: Unlike BMA, BMC converges towards the point where the weightings distribution projects onto the simplex. In other words, instead of selecting the model closest to the distribution, BMC seeks the combination of models that is closest to the generating distribution.
2. BMC Algorithm: Instead of sampling each model in the ensemble individually as in BMA, BMC samples the space of possible ensembles (the space has model weightings drawn randomly from a Dirichlet distribution having uniform parameters). This modification overcomes the tendency of BMA to converge toward giving all of the weight to a single model.
3. BMC Effectiveness: Although BMC is computationally more expensive than BMA, it tends to yield dramatically better results over it, as well as over bagging (Montieth, Carroll, Seppi, and Martinez (2011).
4. BMC in Practice: The results of BMA can often be approximated by using cross-validation to select the best models from a bucket of models. Likewise, the results of BMC may be approximated by using cross-validation to select the best ensemble combination from a random sampling of possible weightings. Ensemble Learning (Wiki) contains sample BMC pseudo-code.

Bucket of Models (BOM)

1. Introduction: A "Bucket of Models" is an ensemble in which a model selection algorithm is used to choose the best model for each problem. When tested with only one problem, the BOM can produce no better results than the best model in the hypotheses set, but when evaluated across many problems, it will typically produce much better results on the average than any single model in the set.
2. BOM Cross-Validation Selection: The most common approach used for model selection is cross-validation (also called a "bake-off contest" – Ensemble Learning (Wiki) contains sample pseudo-code). This may be summed as: "Try them all with the training set, and pick the one that works best" (Zenko (2004)).
3. Gating Cross-Validation Selection: Gating is a generalization of the cross-validation based sample selection. It involves using a separate learning model to decide which of the models in the bucket is best-suited for the problem.
4. Gating Perceptrons: Often, a perceptron is used for the gating model. The perceptron is used to pick the best model, or it can be used to assign a set of linear weights to the predictions from each model in the basket.
5. Landmark Cross-Validation: When a bucket of models is used with a large set of problems, it may be desirable to avoid training some of the models that take a long time to train. Landmark learning is a meta-approach that seeks to address this problem. It involves first training only the fast but imprecise algorithms in the bucket, and then using the performance of these algorithms to determine which slow but accurate algorithm is most likely to do best (Bensusan and Giraud-Carrier (2000)).

Stacking

1. Introduction: Also called a stacking generalization, this involves training a learning algorithm to combine the predictions of several other training algorithms. First, all of the other algorithms are trained using the training set, then a combiner algorithm is used to make a final prediction using all the predictions of the other algorithms as inputs.

2. Stacking as an Ensemble Generalizer: If an arbitrary stacking combiner algorithm is used, then stacking can theoretically represent any of the ensemble techniques described above. In practice, a single-layer logistic regression model is often used as the combiner.
3. Stacking Performance: Stacking typically yields performance better than any single one of the individual trained models (Wolpert (1992)). It has been successfully used on both supervised learning tasks (e.g., regression, see Breiman (1996)) as well as unsupervised learning tasks (e.g., density estimation, Smyth and Wolpert (1999)). It has been used to outperform BMA (Clarke (2003)) and to estimate bagging's error rate (Wolpert and MacReady (1999), Rokach (2010)). The two top performers in the Netflix competition used BLENDING which may be considered to be a form of stacking (Sill, Takacs, Mackey, and Lin (2009)).

Ensemble Averaging vs. Basis Spline Representation

1. Parallel between Spline Representation and Ensemble Averaging Techniques: From a real-valued inference (i.e., regression + transformed classification) point-of-view, there are significant similarities between ensemble averaging techniques and basis spline representation techniques. While ensemble averaging attempts to aggregate over hypotheses set, multi-spline basis representation attempts to represent the splines over the set of the basis spline representation function set. Further, there are also parallels in the way in which the weights are inferred; for basis splines, this is achieved by a combination of best-fit and penalization, whereas for the ensemble aggregator this done via the variance-bias optimization mechanism.
2. Difference: Ensemble averaging may be viewed as a dual pass training exercise effectively – the first training pass trains the individual response basis functions themselves, and the second pass trains the weights across the basis function set.
3. Focus of the Training Passes: Focus of the first pass in ensemble averaging is simply bias reduction, i.e., enhancing the closeness of fit and reduction of Bayes' risk. The second pass performs the ensemble averaging with a view to reducing the variance – this is comparable to the smoothening pass.

4. Ensemble Averaging vs. Multi-Pass State Inference: Bias reduction is comparable to the shape preservation pass, whereas the ensemble averaging/variance reduction is comparable to the smoothening pass.
5. Differences between Ensemble Averaging and Multi-Pass State Inference:
    a. The shape preserving pass ends up emitting a sequence of parameters that correspond to the stretches across the univariate predictor ranges to represent a SINGLE latent state.
    b. The calibration run during the shape preserving pass produces a set of calibrated parameters for the full set of basis splines for shape preservation (i.e., error minimization).
    c. Training each basis function separately during the bias reduction phase of ensemble averaging does not really purport to "infer" any latent state.
    d. The second phase of ensemble aggregation simply re-works the basis weights across all the "low bias" basis functions, again no notion of "re-inferring of states" involved.
    e. The smoothing phase of the multi-phase latent-state construction may work on a latent state quantification metric that is different from the shape preserving pass; while this does loosen the shape preservation impact, it is no more different to the equivalent step in ensemble averaging.

References

- Bensusan, H., and C. G. Giraud-Carrier (2000): Discovering Task Neighborhoods through Landmark Learning Performances *Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery* 325-330.
- Breiman, L. (1996): Bagging Predictors *Machine Learning* **24 (2)** 123-140.
- Brown, G., J. Wyatt, R. Harris, and X. Yao (2005): Diversity Creation Methods: A Survey and Categorization *Information Fusion* **6 (1)** 5-20.
- Clarke, B. (2003): Bayes Model Averaging and Stacking when Model Approximation Error cannot be ignored *Journal of Machine Learning Research* 683-712.

- Domingos, P. (2000): Bayesian Averaging of the Classifiers and the Over-fitting Problem *Proceedings of the 17th International Conference on Machine Learning (ICML)* 223-230.
- Ensemble Learning (Wiki): Wikipedia Entry for Ensemble Learning.
- Garcia-Adeva, J. J., U. Cervino, and R. Calvo (2005): Accuracy and Diversity in Ensembles of Text Categorizers *CLEI Journal* **8 (2)** 1-12.
- Gashler, M., C. Giraud-Carrier, and T. Martinez (2008): Decision Tree Ensemble – Small Heterogenous is better than large Homogenous *The 7th International Conference on Machine Learning and Applications* 900-905.
- Haussler, D., M. Kearns, and R. E. Schapire (1994): Bounds on the Sample Complexity of Bayesian Learning using Information Theory and the VC Dimension *Machine Learning* **14** 83-113.
- Ho, T. (1995): Random Decision Forests *Proceedings of the 3rd International Conference on Document Analysis and Recognition* 278-282.
- Hoeting, J. A., D. Madigan, A. E. Raftery, and C. T. Volinsky (1999): Bayesian Model Averaging: A Tutorial *Statistical Science* **14 (4)** 382-401.
- Kuncheva, C., and L. Whitaker (2003): Measures of Diversity in Classifier Ensembles *Machine Learning* **51** 181-207.
- Minka, T. (2002): Bayesian model averaging is not model combination.
- Mitchell, T. (1997): Machine Learning
- Montieth, K., J. Carroll, K. Seppi, and T. Martinez (2011): Turning Bayesian Model Averaging into Bayesian Model Combination *Proceedings of the International Joint Conference on Neural Networks IJCNN '11* 2657-2663.
- Opitz, D., and R. Maclin (1999): Popular Ensemble Methods: An Empirical Study *Journal of Artificial Intelligence Research* **11** 169-198.
- Polikar, R. (2006): Ensemble based Systems in Decision Making *IEEE Circuits and Systems Magazine* **6 (3)** 21-45.
- Rokach, L. (2010): Ensemble-based Classifiers *Artificial Intelligence Review* **33 (1-2)** 1-39.
- Sahu, A., G. Runger, and D. Apley (2011): Image denoising with multi-phase kernel principal component approach and an Ensemble Version *IEEE Applied Imagery Pattern Recognition Workshop* 1-7.

- Sill, J., G. Takacs, L. Mackey, and D. Lin (2009): Feature-Weighted Linear Stacking **arXiV.**
- Smyth, P., and D. H. Wolpert (1999): Linearly Combining Density Estimators via Stacking *Machine Learning Journal* **36** 59-83.
- Sollich, P., and A. Krogh (1996): Learning with Ensembles: How Over-fitting can be Useful *Advances in Neural Information Processing Systems* **8** 190-196.
- Wolpert, D. (1992): Stacked Generalization *Neural Networks* **5 (2)** 241-259.
- Wolpert, D. H., and RW. G. Macready (1999): An Efficient Method to Estimate Bagging's Generalization Error *Machine Learning Journal* **35** 41-55.
- Zenko, B. (2004): Is Combining Classifiers better than selecting the best One *Machine Learning* 255-273.

ANN Ensemble Averaging

Overview

1. Committee Machines: ANN Ensemble averaging one of the simplest types of committee machines. Along with boosting, it is one of the 2 types of static committee machines (Haykin (1999)). In contrast to standard neural network design techniques in which many networks are generated but only one is retained, ANN ensemble averaging holds on to the less satisfactory networks, but with lower weights (Hashem (1997)).
2. ANN Properties Manipulated by the Ensemble: This comes here from Naftaly, Intrator, and Horn (1997) via Wiki (Ensemble Averaging):
    a. In any ANN, the bias can be reduced at the cost of increased variance, AND
    b. In a group of networks, the variance can be reduced at no cost to bias.
3. ANN Variance-Bias Optimization: Ensemble averaging creates a group of ANN each with low bias and high variance, then combines them to form an ANN that has low bias and low variance, thereby offering an optimal bias-variance resolution (Clemen (1989), Geman, Bienenstock, and Doursat (1992)).

Techniques and Results

1. Methodology: The bias-variance optimization idea above provides an obvious strategy: create a set of experts with low bias and high variance and average them. From an ANN PoV, what this means is the creation of a set of experts with varying parameters; often these are the initial synaptic weights, although other factors such as learning rate, momentum etc. may be varied as well (the drawbacks of varying weight decay and early stopping is discussed in Naftaly, Intrator, and Horn (1997)).
2. The Algorithm:

   a. Generate N experts, each with their own initial values (the initial values are usually chosen randomly from a distribution).
   b. Train each ANN separately.
   c. Use the ensemble to combine the experts and average their values.
3. Classes of ANN Domain Experts: Alternatively, domain knowledge may be used to generate several classes of experts. An expert from each class is trained, and then combined.
4. Linear Combination of Experts: In place of averaging, one may also weight them using

$$\tilde{y}(\vec{x}, \alpha) = \sum_{j=1}^{p} \alpha_j y_j(\vec{x})$$

   where $y_j(\vec{x})$ correspond to the individual experts, and $\{\alpha\}$ to the set of weights. The optimization problem of extracting $\{\alpha\}$ is readily solved through standard neural network approaches. Thus, a meta-network where each neuron is an ANN in itself can be trained, and the synaptic weights on the final network is the weight applied to each ANN expert (Hashem (1997)).
5. Negative Correlation Learning: A more recent ANN ensemble averaging method called negative correlation learning (Liu and Yao (1999)) has been widely used in evolutionary computing.
6. Benefits of ANN Ensemble Averaging:
   a. The resulting committee is less complex than the single machine that achieves the same level of performance (Pearlmutter and Rosenfeld (1990)).
   b. The resulting committee can be trained more easily on smaller input sets (Haykin (1999)).
   c. The resulting committee typically has improved performance over any single network (Hashem (1997)).
   d. The risk of over-fitting is lessened, since there are fewer parameters (weights) that need to be set (Haykin (1999)).

References

- Clemen, R. T. (1989): Combining Forecasts: Review and Annotated Bibliography *International Journal of Forecasting* **5 (4)** 559-583.
- Geman, S., E. Bienenstock, and R. Doursat (1992): Neural Networks and the Bias/Variance Dilemma *Neural Computation* **4** 1-58.
- Hashem, S. (1997): Optimal Linear Combinations of Neural Networks *Neural Computation* **10 (4)** 599-614.
- Haykin, S. (1999): *Neural Networks – A Comprehensive Foundation 2nd Edition* **Prentice Hall** Upper Saddle River, NJ.
- Liu, Y., and X. Yao (1999): Ensemble Learning via Negative Correlation *Neural Networks* **12 (20)** 1399-1404.
- Pearlmutter, B. A., and R. Rosenfeld (1990): Chaitkin-Kolmogorov Complexity and Generalization in Neural Networks *Proceedings of the 1990 Conference in Advances in Neural Information Processing Systems* **3** 931.
- Naftaly, U., N. Intrator, and D. Horn (1997): Optimal Ensemble Averaging of Neural Networks *Networks: Computation in Neural Systems* **8 (3)** 283-296.

Boosting

Overview

1. Definition: Boosting is a machine learning meta-algorithm for reducing bias in supervised learning (Boosting (Wiki)). Boosting techniques were developed in response to the question (Kearns (1988)): Can a set of weak learners create a strong learner (Schapire (1990))?
2. Weak Learner vs. Strong Learner: A weak learner is defined to be a classifier that is only slightly correlated with the true classification (i.e., it can label examples better than random guessing). In contrast, a strong learner is arbitrarily well-correlated with true classification.
3. Fast Hypotheses Boosters: Algorithms that achieve boosting quickly simply become known as "boosting algorithms". The ARC technique (Freund and Schapire (1997) – adaptive re-sampling and combination), as a general technique, quickly became more or less synonymous with boosting (Breiman (1998)).

Philosophy behind Boosting Algorithms

1. Principle: While boosting is not algorithmically constrained, most boosting algorithms consist of iteratively learning weak classifiers with respect to a distribution, and eventually combining them to make a strong classifier. The weights of addition are related to the weak learners' accuracy.
2. Sample Re-weighting: For purposes of re-weighting, the examples that are mis-classified gain weight, and those that are correctly classified lose weight (some boosting algorithms reduce the weights of repeatedly misclassified samples – e.g., boost by majority and BrownBoost). Thus, future weak learners focus more on the examples that previous weak learners misclassified.

3. The Original Boosting Algorithms: The original ones, proposed by Schapire (1990) (a recursive majority gate formulation algorithm) and Freund (boost by majority – Llew, Baxter, Bartlett, and Frean (2000)) were not adaptive, and could not take full advantage of the weak learners.
4. Probably Approximately Correct Boosters: Only algorithms that are provably correct on the probably approximately correct framework are called boosting algorithms. Other algorithms that are similar in spirit to boosting are called "leveraging algorithms", although they are sometimes incorrectly referred to as boosting algorithms (Llew, Baxter, Bartlett, and Frean (2000)).

Popular Boosting Algorithms and Drawbacks

1. Variations among Boosting Algorithms: The main variation among the many boosting algorithms is their method of weighting the training data points and the corresponding hypotheses.
2. Popular Boosters: The first one was AdaBoost; recent ones include LPBoost, TotalBoost, BrownBoost, MadaBoost, and LogitBoost. Many boosting algorithms fit into the AnyBoost framework (Llew, Baxter, Bartlett, and Frean (2000)), which demonstrates that boosting essentially performs a gradient descent slide in the function space using a convex cost function.
3. Application: Boosting algorithms are used in Computer Vision, where individual classifiers detecting contrast changes can be combined to identify facial features (OpenCV Cascade Classifier).
4. Boosting under Random Classification Noise: Long and Servedio (2010) suggest that many of the boosting algorithms listed above are probably flawed. They conclude that convex potential boosters cannot withstand random classification noise, thus rendering the applicability of such algorithms for noisy, real world data questionable.
5. Causes for the Boosting Flaw: Long and Servedio (2010) show that if any non-zero fraction of the training data is mislabeled, the boosting algorithm tries extremely hard to correctly classify the training examples, and fails to produce a model with accuracy better than 0.5.

This result does not apply to branching program-based boosters, but does apply to AdaBoost, LogitBoost, and others.

References

- Boosting (Wiki): Wikipedia Entry for Boosting.
- Breiman, L. (1998): Arcing Classifier *Annals of Statistics* **26 (3)** 801-849.
- Freund, Y., and R. E. Schapire (1997): A Decision-Theoretic Generalization of On-line Learning and an Application to Boosting *Journal of Computer and System Sciences* **55 (1)** 119-139.
- Kearns, M. (1988): Thoughts on Hypothesis Boosting.
- Llew, M., J. Baxter, P. Bartlett, and M. Frean (2000): Boosting Algorithms as Gradient Descent *Advances in Neural Information Processing Systems* (S. A. Solla, T. K. Leen, and K. R. Muller, editors) **12** 512-518.
- Long, P. M., and R. A. Servedio (2010): Random Classification Noise Defeats all Convex Potential Boosters *Machine Learning* **78 (3)** 287-304.
- Schapire, R. E. (1990): The Strength of Weak Learnability *Machine Learning* **5 (2)** 197-227.

Bootstrap Aggregating

Overview and Sample Generation

1. Definition: Bootstrap aggregating (bagging) is a machine learning ensemble meta-algorithm designed to improve the stability and the accuracy of machine learning algorithms used in statistical classification and regression (Bootstrap Aggregating (Wiki)). It also reduces variance and helps avoid over-fitting. Although it is usually applied to decision tree techniques, it can be used with any type of method. Bagging is a special case of BMA.
2. Bagging Bootstrap Samples: Given a training set D of size n, bagging generates m new training sets D_i each of size n' by sampling from D uniformly and with replacement. This is called a bootstrap sample.
3. Sampling with Replacement: By sampling with replacement, some observations may be repeated in D_i; for

$$n = n'$$

and large n, D_i is expected to have

$$1 - \frac{1}{e}$$

(i.e., ~ 63%) of the unique samples of D, the rest being duplicates (more generally, when drawing with replacement n' values from a set of n (different and equally likely), the expected number unique draws is

$$n\left(1 - e^{-\frac{n'}{n}}\right)$$

(Aslam, Popa, and Rivest (2003)).

4. Bagging Ensemble Combiners: The m models are fit using the above m bootstrap samples and combined by averaging the output (for regression) or voting (for classification).
5. Applications: Bagging leads to improvements for unstable procedures (Breiman (1996), Sahu, Runger, and Apley (2011)) which includes neural nets, classification and regression trees, and subset selection in linear regression.
6. Caveat with kNN: On the other hand, blind application of bagging can mildly degrade performance of stable methods such as kNN (Breiman (1996)).

Bagging with 1NN – Theoretical Treatment

1. 1NN Improvements with Bagging: It is well known that the error-rate of a 1NN classifier is at most twice that of the Bayes' classifier, but there are no guarantees that this classifier will be consistent. By careful choice of the re-samples, bagging can lead to substantial improvements on the performance of the 1NN classifier.
2. 1NN Enhancement Algorithm: By taking a large number of re-samples of the data of size n' each, the bagged nearest neighbor classifier will be consistent provided

$$n' \to \infty$$

diverges, but

$$\frac{n'}{n} \to 0$$

as the sample size

$$n \to \infty$$

3. Bagging 1NN Formulation: Under infinite simulation, the bagged nearest neighbor classifier may be treated as a weighted nearest neighbor classifier. Suppose that the feature space is d-dimensional. Denote by $C_{n,n'}^{bnn}$ the bagged nearest neighbor classifier based on a training set of size n, with resamples each of size n'. In the infinite sampling case, under certain regularity conditions on the class distributions, the excess risk (i.e., excess over Bayes', the perfect classifier) has the following asymptotic expansion (Samworth (2012)):

$$R_R\left(C_{n,n'}^{bnn}\right) - R_R\left(C_{Bayes}\right) = \left[B_1 \frac{n'}{n} + B_2 \frac{1}{n'^{4/d}}\right]\{1 + O(1)\}$$

for some constants B_1 and B_2. The optimal choice of n' that balances the two terms in the asymptotic expansion is given by

$$n' = Bn^{\frac{d}{d+4}}$$

for some constant B.

References

- Aslam, J. A., R. A. Popa, and R. L. Rivest (2007): On Estimating the Size and Confidence of a Statistical Audit *Proceedings of the Electronic Voting Technology (EVT '07)* **Boston MA**.
- Bootstrap Aggregating (Wiki): Wikipedia Entry for Bootstrap Aggregating.
- Breiman, L. (1996): Bagging Predictors *Machine Learning* **24 (2)** 123-140.
- Sahu, A., G. Runger, and D.Apley (2011): Image denoising with multi-phase kernel principal component approach and an Ensemble Version *IEEE Applied Imagery Pattern Recognition Workshop* 1-7.
- Samworth, R. J. (2012): Optimal Weighted Nearest Neighbor Classifiers *Annals of Statistics* **40 (5)** 2733-2763.

Tensors and Multi-linear Subspace Learning

Tensors

1. Definition: Tensors are geometric objects used to describe linear relationships between scalars/vectors/other tensors. Order/Degree/Rank of a tensor is the dimensionality needed to represent the tensor (e.g., scalars are tensors of rank zero). See Hamilton (1854-1855), Voigt (1898), Goodstein (1982), Reich (1994), Pais (2005), and Tensor (Wiki) for the origin and the varied uses of tensors.
2. Families of Predictor Ordinates: Each family of the spanning set of predictor ordinates of the tensor corresponds to a single predictor ordinate dimension of the tensor. For example, 3D pixel predictor ordinate families may consist of a) x, y, and z location co-ordinate family, and b) RGB color family of predictors.
3. Covariant vs. Contravariant Tensor Components: If a tensor component transforms as the tensor itself, it is called a covariant component, and is represented using a subscript index. If a component transforms as the inverse of the tensor, it is called a contravariant component of the tensor, and is represented using a superscript. For instance, in a 2D tensor matrix, the row indices would be covariant and the column indices would be contra-variant.
    - Family Spanning Set #1 <-> Family Spanning Set #2 Transforming Tensor => This is a transforming tensor, with source family spanning set #1 acting as the covariant basis component tensor, and the destination family spanning set #2 acting as the contra-variant basis component tensor.
4. Expressing Covariant and Contravariant Tensors as Multi-dimensional Arrays: The transformation law for an order m tensor with n contra-variant indices and $m - n$ covariant indices is given as (Marion and Thornton (1995), Sharpe (1997), Griffiths (1999))

$$\hat{T}^{i_1,\cdots,i_n}_{i_{n+1},\cdots,i_m} = (R^{-1})^{i_1}_{j_1} \cdots (R^{-1})^{i_n}_{j_n} R^{i_{n+1}}_{j_{n+1}} \cdots R^{i_m}_{j_m} \hat{T}^{j_1,\cdots,j_n}_{j_{n+1},\cdots,j_m}$$

This expression uses the Einstein notation to sum over $j_1, \cdots, j_n$ (i.e., the summation is implicit).

5. Expressing Tensors as Field Transformation Jacobians (or Tensor Fields): Here, the transformation law is expressed in terms of the partial derivatives of the co-ordinate functions $\vec{x} = (x_1, \cdots, x_k)$ thereby defining the transformation in terms of the Jacobian (Curbastro (1892), Klein (1972)) as

$$\hat{T}^{i_1,\cdots,i_n}_{i_{n+1},\cdots,i_m}(\bar{x}_1, \cdots, \bar{x}_k) = \frac{\partial \bar{x}_{i_1}}{\partial x_{j_1}} \cdots \frac{\partial \bar{x}_{i_n}}{\partial x_{j_n}} \frac{\partial x_{j_{n+1}}}{\partial \bar{x}_{i_{n+1}}} \cdots \frac{\partial x_{j_m}}{\partial \bar{x}_{i_m}} \hat{T}^{j_1,\cdots,j_n}_{j_{n+1},\cdots,j_m}(x_1, \cdots, x_k)$$

6. Expression as Tensor Products or Multi-linear Maps: A type (m, n) tensor is defined as an element in the tensor product of vector spaces (Hazewinkel (2001)) as

$$T \in \{V \times \cdots \times V\} \bigotimes \{V^* \times \cdots \times V^*\}$$

where the first term in the bracket spans n copies of the vector space V, and the second term in the bracket spans m copies of the vector space V^*. Thus, if $\hat{v}_i$ is the basis of $\vec{V}$ and $\hat{w}_j$ is the basis of $\vec{W}$, then $\vec{V} \otimes \vec{W}$ has the natural basis $\hat{v}_i \otimes \hat{w}_j$.

7. Expressed using Penrose Graphical Notation: In the diagrammatic notation (Penrose (2007), Wheeler, Misner, Thorne (1973)), tensor symbols are replaced by shapes, and indices by lines and curves!
8. Tensor Operations:
   - Tensor Product =>

$$S(l, k) \bigotimes T(n, m) = S \bigotimes T(l + n, k + m)$$

   - Tensor Contraction => This reduces the tensor order by 2, i.e.,

$$S(l, k) \Longrightarrow S(l - 1, k - 1)$$

   - Raising an Index => Changes a Covariant Index to a Contravariant Index.

- Lowering an Index => Changes a Contravariant Index to a Covariant Index.

9. Infinite Dimensional Tensors: Infinite dimensional tensors may be generalized via the tensor product of Hilbert spaces (Segal (1956)), or by extending multi-maps that employ infinite-dimensional vector spaces and their algebraic duals (this is automatically achieved by using infinite-dimensional Banach manifolds and their continuous duals (Lang (1972), Abraham, Marsden, and Ratiu (1988))).
10. Tensor Density: A tensor with density r transforms like an ordinary tensor under co-ordinate transformations (Hazewinkel (2001)), with the exception that is scaled by the determinant of Jacobian to the power r (i.e., $\left\|\frac{\partial \hat{r}}{\partial \bar{x}}\right\|^{r}$).

Multi-linear Subspace Learning

1. Definition: Multi-linear Subspace Learning (MSL) aims to learn a specific part of a large space of multi-dimensional objects having a desired property. Essentially, MSL is a dimension reduction technique for finding low dimension representation with preferred characteristics, without resorting to vectorization (Multi-linear Subspace Learning (Wiki), He, Cai, and Niyogi (2005), Lu, Plataniotis, and Venetsanopoulus (2011)).
2. MSL as a Generalization of PCA: MSL may also be viewed as a higher order PCA/CCA/LDA (Vasilescu and Terzopoulus (2007)).
3. Challenges with Vectorized Linear Subspace Learners: Since they represent input data as vectors and attempt to solve for optimal mapping to the lower dimensional space, vectorized subspace learners become:
   - Inadequate when dealing with massive multi-dimensional data.
   - Need estimation of a large number of parameters.
   - Break down the structure and the correlation inherent in the original data (Yan, Xu, Yang, Zhang, Tang, and Zhang (2005), Lu, Plataniotis, and Venetsanopoulus (2008)).
4. MSL vs. Tensor Decomposition: Both are similar in that both use multi-linear algebra (Kolda and Bader (2009)), but differ in that while tensor decomposition focuses on factor analysis, MSL focuses on dimensionality reduction.

5. Multi linear Subspace Definition: This simply refers to the multi-linear projection that maps the input tensor data from a higher dimensional space to a lower dimensional space (Hitchcock (1927)).
6. Tensor-to-Tensor Projection (TTP): This is a direct projection of a high dimensional tensor to a low dimensional tensor of the same order, using N projection matrices for the order N tensor – it is simply an extension of HOSVD (Tucker (1966), Lathauwer, Moor, and van de Walle (2000a)).
7. Tensor-to-Vector Projection (TVP): This is a projection from a high dimension tensor to a low dimension vector, and is also referred to as a Rank-1 projection. A TVP of a tensor to a P-dimensional vector consists of P projections from the tensor to a scalar – each projection is called an EMP (elementary multi-linear projection) (Carroll and Chang (1970), Harshman (1970)).
8. MSL Solution Approach: N set of parameters need to solved, one per each mode. The following sub-optimal approach is followed (Kroonenberg and de Leeuw (1980), Lathauwer, Moor, and van de Walle (2000b)):
    - Initialize a set of projections in each mode.
    - Fix all but one projection, and solve for that fixed projection.
    - Perform mode-wise optimization until convergence.
9. Multi-linear Extensions to PCA:
    - TTP based MPCA (Lu, Plataniotis, and Venetsanopoulus (2008))
    - TVP based uncorrelated MLPCA (UMPCA) (Lu, Plataniotis, and Venetsanopoulus (2009b)).
10. Multi-linear Extensions to LDA:
    - TTP based Discriminant Analysis with Tensor Representation (Yan, Xu, Yang, Zhang, Tang, and Zhang (2005)).
    - TTP based General Tensor Discriminant Analysis (GTDA) (Tao, Li, Wu, and Maybank (2007)).
    - TTP based Uncorrelated Multi-linear Discriminant Analysis (UMLDA) (Lu, Plataniotis, and Venetsanopoulus (2009a)).
11. Multi-linear Extensions to CCA:
    - TTP based Tensor Canonical Correlation Analysis (TCCA) (Kim and Cipolla (2009)).

- TVP based Multi-linear Canonical Correlation Analysis (MCCA) (Lu (2013)).

Multi-linear PCA

1. Definition: MPCA is a mathematical procedure that uses multiple orthogonal transformations to convert a set of multi-dimensional objects into another set of multi-dimensional objects of lower dimension.
2. Purpose: The aim is to capture as high a variance as possible, accounting for as much of the variability in the data as possible, subject to the constraint of mode-wise orthogonality.
3. PCA as Response De-convolution: Given an array of responses, one way to look at PCA/ICA is as a process to extract/infer the individual, independent predictor drivers from the observation responses.
4. MPCA vs. Regular PCA: PCA needs to reshape the multidimensional object into a vector, while MPCA works directly on the multi-dimensional objects through mode-wise processing. For example, PCA converts a 100×100 image into a vector of size $100{,}000 \times 1$, while MPCA processes the same 100×100 using 2 100×1 vectors. Thus, MPCA results in savings of 50 in processing time in this case.
5. Use of MSL Techniques in MPCA: Like MSL, MPCA uses tensor based dimensionality reduction techniques (Lu, Plataniotis, and Venetsanopoulus (2008)), therefore MSL approaches such as Tucker decomposition (Tucker (1966)), HOSVD (Lathauwer, Moor, and van de Walle (2000a)), and best-rank $(R_1, R_2, \cdots, R_N)$ approximation of higher-order tensors (Lathauwer, Moor, and van de Walle (2000b)) are all applicable here.
6. The MPCA Algorithm: MPCA performs feature extraction by determining the multi-linear projection that captures most variations in a given mode. It works on centered data, and typically uses the altering least squares (ALS) approach (Kroonenberg and de Leeuw (1980)), by alternating across each mode.
    - ALS decomposes the original tensor into multiple projection sub-problems, each of which is a classical PCA, and therefore easily solved.
7. Retention of the Feature Correlations: Because of the tensor-to-tensor nature of the transformation, MPCA features are not uncorrelated in general, although the transformation

in each mode is orthogonal (to generate uncorrelated features UMPCA (Lu, Plataniotis, and Venetsanopoulus (2009b)) is used).

8. Feature Selection in MPCA: While conventional classifiers often use only vector features, specialized tensor feature selection to enhance/improve MPCA performance maybe used in specific situations (examples are supervised discriminant MPCA feature selection for object recognition (Lu, Plataniotis, and Venetsanopoulus (2008)), and unsupervised MPCA feature selection for visualization tasks (Lu, Eng, Thida, and Plataniotis (2010)).
9. MPCA Extensions:
    - Uncorrelated MPCA (UMPCA) (Lu, Plataniotis, and Venetsanopoulus (2009b))
    - Boosting + MPCA (Lu, Plataniotis, and Venetsanopoulus (2009c))
    - Non-negative MPCA (NMPCA) (Panagakis, Kotropoulus, and Arce (2010))
    - Robust MPCA (RMPCA) (Inoue, Hara, and Urahama (2009))
    - More extensions are detailed in (Lu, Plataniotis, and Venetsanopoulus (2011))

References

- Abraham, R., J. E. Marsden, and T. S. Ratiu (1988): *Chapter 5 => Manifolds, Tensor Analysis, Applications,* in *Applied Mathematical Sciences* **Springer-Verlag**, New York.
- Carroll, J. D., and J. Chang (1970): Analysis of Individual Differences in Multi-dimensional Scaling via an n-way Generalization of the Eckert-Young Decomposition *Psychometrika* **35** 283-319.
- Curbastro, R. (1892): Resume de quelques travaux sur les systemes variables de fonctions associes a une forme differentielle quadratique *Bulletin des Sciences Mathematiques* **2 (16)** 167-189.
- Goodstein, J. R. (1982): The Italian Mathematics of Relativity *Centaurus* **26 (3)**: 241-261.
- Griffiths, D. J. (1999): *Introduction to Electrodynamics 3rd Edition* **Saunders College Publishing**.
- Hamilton, W. R. (1854-1855): On some Extensions of Quaternions *Philosophical Magazine* **7-9**: 492-499, 125-137, 261-269, 46-51, 280-290.

- Harshman, R. A. (1970): Foundations of the PARAFAC Procedure: Models and Conditions for an Explanatory Multi-Model Factor Analysis *UCLA Working Papers in Phonetics* **16** 1-84.
- Hazewinkel, M. (2001): Tensor Density *Encyclopedia of Mathematics* **Springer**.
- He, X., D. Cai, and P. Niyogi (2005): Tensor Subspace Analysis *Advances in Neural Information Processing Systems 18 (NIPS)*.
- Hitchcock, F., L. (1927): The Expression of a Tensor or a Polyadic as a Sum of Products *Journal of Mathematics and Physics* **6** 164-189.
- Inoue, K., K. Hara, and K. Urahama (2009): Robust Principal Component Analysis *Proceedings of IEEE Conference on Computer Vision* 591-597.
- Kim, T. K., and R. Cipolla (2009): Canonical Correlation Analysis for Video Volume Tensors for Action Categorization and Detection *IEEE Transactions on Pattern Analysis and Machine Intelligence* **31 (8)** 1415-1428.
- Klein, M. (1972): *Mathematical Thought from Ancient to Modern Times* **3** 1122-1127.
- Kolda, T. G., and B. W. Bader (2009): Tensor Decompositions and Applications *SIAM Review* **51 (3)** 455-500.
- Kroonenberg, P. M., and J. de Leeuw (1980): Principal Component Analysis of Three-Mode Data by means of Alternating Least Squares Algorithms *Psychometrika* **45** 69-97.
- Lang, S. (1972): *Differential Manifolds* **Addison-Wesley Publisher Co** Reading, Massachusetts.
- Lathauwer, L. D., B. D. Moor, and J. van de Walle (2000a): A Multilinear Singular Value Decomposition *SIAM Journal of Matrix Analysis and Applications* 2**1 (4)** 1253-1278.
- Lathauwer, L. D., B. D. Moor, and J. van de Walle (2000b): On the best Rank-1 and Rank-$(R_1, \cdots, R_N)$ Approximation of Higher-order Tensors *SIAM Journal of Matrix Analysis and Applications* 2**1 (4)** 1324-1342.
- Lu, H., K. N. Plataniotis, and A. N. Venetsanopoulus (2008): MPCA: Multilinear Principal Component Analysis of Tensor Objects *IEEE Transactions on Neural Networks* **19 (1)** 18-39.
- Lu, H., K. N. Plataniotis, and A. N. Venetsanopoulus (2009a): Uncorrelated Multilinear Discriminant Analysis with Regularization and Aggregation for Tensor Object Recognition *IEEE Transactions on Neural Networks* **20 (1)** 103-123.

- Lu, H., K. N. Plataniotis, and A. N. Venetsanopoulus (2009b): Uncorrelated Multilinear Principal Component Analysis for Unsupervised Multilinear Subspace Learning *IEEE Transactions on Neural Networks* **20 (11)** 1820-1836.
- Lu, H., K. N. Plataniotis, and A. N. Venetsanopoulus (2009c): Boosting Discriminant Learners for Gait Recognition using MPCA Features *EURASIP Journal on Image and Video Processing*.
- Lu, H., H. L. Eng, M. Thida, and K. N. Plataniotis (2010): Visualization and Clustering of Crowd Video Content in MPCA Subspace *Proceedings of the 19th ACM Conference on Information and Knowledge Management* **Toronto**, ON, Canada.
- Lu, H., K. N. Plataniotis, and A. N. Venetsanopoulus (2011): A Survey of Multilinear Subspace Learning for Tensor Data *Pattern Recognition* **44 (7)** 1540-1551.
- Lu, H. (2013): Learning Canonical Correlations of Paired Tensor Sets vis Tensor-to-Vector Projection *Proceedings of the 23rd International Joint Conference on Artificial Intelligence* **Beijing**, China.
- Marion, J. B., and S. T. Thornton (1995): *Classical Dynamics of Particles and Systems 4th Edition* **Saunders College Publishing**.
- Multi-linear Subspace Learning (Wiki): Wikipedia Entry for Multi-linear Subspace Learning.
- Pais, A. (2005): *Subtle is the Lord: The Science and the Life of Albert Einstein* **Oxford University Press**.
- Panagakis, W., C. Kotropoulus, G. R. Arce (2010): Non-negative multilinear Principal Component Analysis of Auditory Temporal Modulations for Music Genre Classification *IEEE Transactions on Audio, Speech, and Language Processing* **18 (3)** 576-588.
- Penrose, R. (2007): *The Road to Reality* **Vintage Books**.
- Reich, K. (1994): *Die Entwicklung des Tensorkalkuls, Science Networks Historical Studies, vol. 11* **Birkhauser**.
- Segal, I. E. (1956): Tensor Algebras over Hilbert Spaces I *Transactions of the American Mathematical Society* **81 (1)**: 106-134.
- Sharpe, R. W. (1997): *Differential Geometry: Cartan's Generalization of Klein's Erlangen Problem* **Springer-Verlag**, Berlin, New York.
- Tensor (Wiki): Wikipedia Entry for Tensor.

- Tucker, L. D. (1966): Some Mathematical Notes on Three-Mode Factor Analysis *Psychometrika* **31 (3)** 279-311.
- Vasilescu, M. A. O., and D. Terzopoulus (2007): Multilinear Projection for Appearance-based Recognition in the Tensor Framework *IEEE 11th International Conference on Computer Vision* 1-8.
- Voigt, W. (1898): *Die Fundamentalen Physikalicshen Eigenschaften der Krystalle in Elementarer Derstellung* **Von Veit**, Leipzig.
- Wheeler, J. A., C. Misner, and K. S. Thorne (1973): *Gravitation* **W. H. Freeman & Co**.
- Yan, S., D. Xu, Q. Yang, L. Zhang, X. Tang, and H. J. Zhang (2005): Discriminant Analysis with Tensor Representation *IEEE Conference on Computer Vision and Pattern Recognition* **I** 526-532.

Kalman Filtering

1. Original Formulation: Despite the name, Kalman filtering had already been formulated by others – see Stratonovich (1959a, 1959b, 1960a, 1960b), Lauritzen (1981, 2002). Kalman Filter (Wiki) contains additional references.
2. Applications: In addition to the most applied areas such as guidance, navigation, and control of vehicles such as spacecraft/aircraft, and computer vision, Kalman filtering also applied in structural macroeconomic models (Andreasen (2008), Strid and Walentin (2009)), tracking and vertex fitting of charged particles in particle detectors (Fruhwirth (1987)), and human sensorimotor processing (Wolpert (1996)). Of particular relevance to illiquid trading may be the application of Kalman Filtering for the recovery of sparse, dynamic signals using restricted isometry and probabilistic recovery (Vaswani (2008) and Carmi, Gurfil, and Kanevsky (2010)).
3. Kalman Filtering vs. Hidden Markov Models: While there is a lot of similarity, there are a few critical differences. First, the hidden state variables in Kalman filtering are continuous, whereas in HMM they are discrete. However, the HMM can represent an arbitrary distribution for the state variables, whereas Gaussian noise models are used in Kalman filtering. The parallel between the state equations using HMM and the Kalman filtering treatments is compared in Hamilton (1994) and Roweis and Ghahramani (1999).
    - Markov Processes vs. Filtering => Although filtering processes seem to be treated in conjunction with HMM, filtering processes do not need to be Markov at all. However, there may be an impact on real-time applicability for non-Markov systems.
    - Extended Markov Systems => Perhaps using a few additional past observations may still not overtly compromise the speed of computation of what was a Markov process, but may improve the estimation quality (for e.g., using the limited Volterra expansions). This is NOT the same as Kalman smoothing, however.

4. Dempster-Shaefer Theoretical Underpinning: Under the Dempster-Shaefer theory, each state equation/observation is the result of a linear belief function, and the Kalman filter results under the special case of combining the linear belief functions on a Markov tree.
5. Definition: The Kalman Filter, also known as linear quadratic estimation (LQE), is an algorithm that uses a series of measurements observed over time, containing noise (random variations), and produces estimates of unknown state variables that tend to be more precise than those based on single measurement alone.
6. State Definitions:
   - $\hat{x}_{k|k-1}$: This is the a priori state estimate at time k given all the observations to the instant $k-1$.
   - $\hat{x}_{k|k}$: This is the a-posteriori state estimate at time k given all the observations to the instant k, and after applying the appropriate gain adjustment to the measurement.
   - x_k The actual state at time k.
7. Covariance Definitions:

$$P_{k|k} = Covariance\left(x_k - \hat{x}_{k|k}\right)$$

$$P_{k|k-1} = Covariance\left(x_k - \hat{x}_{k|k-1}\right)$$

$$S_k = Covariance(\tilde{y}_k)$$

   $\tilde{y}_k$ will be defined later.
8. A Priori State Estimation Model of the Kalman Filter:

$$\hat{x}_{k|k-1} = F_k\hat{x}_{k-1|k-1} + B_k u_k + w_k$$

   - F_k is the state transition model applied to the previous a posteriori state estimate $\hat{x}_{k-1|k-1}$.
   - B_k is the control-input model that is applied to the control vector u_k.
   - w_k is the process noise with zero-mean multivariate normal with co-variance Q_k.

- The corresponding a priori error covariance matrix estimate is

$$P_{k|k-1} = F_k P_{k-1|k-1} {F_k}^T + Q_k$$

9. Measurement of the True State:

$$z_k = H_k x_k + v_k$$

- H_k is the observation model operator that maps the true state space into the observed space.
- v_k is the observation noise that is a zero-mean Gaussian white noise with co-variance R_k.
- z_k is the observation/measurement of the true state x_k.

10. Assumption of the Variable Independence in the Kalman Filter: The initial state and the noise vector at each step $\{x_0, w_1, \cdots, w_k, v_1, \cdots, v_k\}$ are all mutually independent.
11. Innovation/Update Phase of the Kalman Filter: The innovation of the measurement residual is

$$\tilde{y}_k = z_k - H_k \hat{x}_{k|k-1}$$

This residual is generated by applying the measurement operator on the a priori state estimate $\hat{x}_{k|k-1}$.

12. Methodology Separation in the Kalman Filter: The Kalman Filtering methodology seeks to determine the a priori state estimate and the a priori error covariance estimate separately from the a-posteriori state estimate and the a-posteriori error covariance estimate.
13. A Posteriori State Estimation: The a-posteriori state estimate is typically expressed in terms of the Kalman gain K_k, i.e.,

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k \tilde{y}_k$$

- Observation weighting vs. Model weighting through Kalman Gain => Expanding out $\tilde{y}_k$, we get

$$\hat{x}_{k|k} = (I - K_k H_k)\hat{x}_{k|k-1} + K_k z_k$$

To uncover the intuition behind this, set

$$H_k = I$$

to get

$$\hat{x}_{k|k} = (I - K_k)\hat{x}_{k|k-1} + K_k z_k$$

Thus, here the unit gain

$$K_k = 1$$

weights $\hat{x}_{k|k}$ toward the measurement z_k, whereas the zero gain

$$K_k = 0$$

weights the a-posteriori estimate towards the model.

14. Kalman Filter Invariants:

$$\langle \hat{x}_{k|k} - \hat{x}_{k|k} \rangle = \langle \hat{x}_{k|k} - \hat{x}_{k|k-1} \rangle = 0$$

$$\langle \tilde{y}_k \rangle = 0$$

15. The Optimization Step in Kalman Gain: Optimization is really only applied to the a-posteriori error covariance, because that is where you can optimize using the Kalman gain.
16. A Posteriori Covariance Formulation:

$$P_{k|k} = Covariance\left[x_k - \hat{x}_{k|k}\right] = Covariance\left[x_k - \left\{\hat{x}_{k|k-1} + K_k\tilde{y}_k\right\}\right]$$
$$= Covariance\left[x_k - \left\{\hat{x}_{k|k-1} + K_k\left(z_k - H_k\hat{x}_{k|k-1}\right)\right\}\right]$$
$$= Covariance\left[x_k - \left\{\hat{x}_{k|k-1} + K_k\left(H_k\hat{x}_{k|k} + v_k - H_k\hat{x}_{k|k-1}\right)\right\}\right]$$
$$= Covariance\left[\left(I - K_kH_k\right)\left(x_k - \hat{x}_{k|k-1}\right)\right] + Covariance[K_kv_k]$$

Here we have used the fact that

$$Covariance(A - B) = Covariance(A) + Covariance(B)$$

if

$$Covariance(AB) = 0$$

$$P_{k|k} = (I - K_kH_k)Covariance\left(x_k - \hat{x}_{k|k-1}\right)(I - K_kH_k)^T + K_kCovariance[v_k]{K_k}^T$$

$$P_{k|k} = (I - K_kH_k)P_{k|k-1}(I - K_kH_k)^T + K_kR_k{K_k}^T$$

17. <u>A Priori/A Posteriori Prediction/Update Summary</u>:

- A Priori State Prediction =>

$$\hat{x}_{k|k-1} = F_k\hat{x}_{k-1|k-1} + B_ku_k + w_k$$

- A Priori State Error Covariance Estimation =>

$$P_{k|k-1} = F_kP_{k-1|k-1}{F_k}^T + Q_k$$

- A Posteriori State Prediction/Update =>

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k\tilde{y}_k$$

- A Posteriori State Error Covariance Estimate =>

$$P_{k|k} = (I - K_k H_k) P_{k|k-1} (I - K_k H_k)^T + K_k R_k {K_k}^T$$

This form of a posteriori state error covariance is called the Joseph's form. It is true for any Kalman system, not just the optimal system.

18. Re-expansion and Formulation of the Kalman Gain:

$$P_{k|k} = P_{k|k-1} - K_k H_k P_{k|k-1} - P_{k|k-1} {H_k}^T {K_k}^T + K_k H_k P_{k|k-1} {H_k}^T {K_k}^T + K_k R_k {K_k}^T$$

$$P_{k|k} = P_{k|k-1} - K_k H_k P_{k|k-1} - P_{k|k-1} {H_k}^T {K_k}^T + K_k S_k {K_k}^T$$

where we use the fact that

$$S_k = H_k P_{k|k-1} {H_k}^T + R_k$$

Remember that S_k is the variance of $\tilde{y}_k$, i.e.,

$$S_k = \langle {\tilde{y}_k}^2 \rangle$$

Further, notice the similarity between S_k and $P_{k|k-1}$, where we have switched H_k for F_k and R_k for Q_k - this results from identically analogous input drivers.

19. Minimization of the A Posteriori Error Vector: Minimization of $x_k - \hat{x}_{k|k}$ is the same as minimization of the trace of $P_{k|k}$.

20. Optimal Kalman Gain Formulation: Trace Minimization of $P_{k|k}$ =>

$$\frac{\partial Trace(P_{k|k})}{\partial K_k} = -2(H_k P_{k|k-1})^T + 2K_k S_k$$

$$\frac{\partial Trace(P_{k|k})}{\partial K_k} = 0$$

implies

$$K_k S_k = (H_k P_{k|k-1})^T = P_{k|k-1} {H_k}^T$$

where we've used the fact

$$P_{k|k-1} = {P_{k|k-1}}^T$$

Thus, the optimal Kalman Gain is

$$K_k = P_{k|k-1} {H_k}^T {S_k}^{-1}$$

$$\frac{\partial^2 Trace(P_{k|k})}{\partial {K_k}^2} = S_k > 0$$

Thus, the optimal $K_k$corresponds to the desired minimum. The corresponding optimal a posteriori Covariance becomes

$$P_{k|k} = (I - K_k H_k) P_{k|k-1}$$

The Optimal Kalman Gain is Inversely Proportional to the Observation Noise Variance. Thus, more weight (i.e., less gain) is applied to the predicted estimate in cases where the measurement noise variance is high. High Gain => Closer to measurements, therefore the filter becomes more responsive, but also jumpy (non-smooth).

21. Estimation of the Noise Covariances: Estimation of R_k and Q_k is often difficult. A recent promising method is the method of Autocovariance Least Squares, as laid out in Rajamani (2007), Rajamani and Rawlings (2009), and the reference on Autocovariance Least Squares.

- Remember that auto-regression is done on the innovation residual, as that is where the "update" rubber meets the "measurement" road. To eliminate/reduce bias, the expectation is that the innovation residual should be composed of evenly spread white noise.
- Further, the sensitivity Jacobian of the Filter to the noise covariances indicate how robust the estimator is to the potentially mis-specified noise and other statistical parameters (Anderson and Moore (1979)).

22. Filter Performance Estimation: Optimal performance of Kalman Filter can be cross-checked with the quality of white noise generated by the innovation sequence (see e.g., Matisko and Havlena (2012)).
23. Numerical Stability of the Filtering Algorithm Sequence: If the process noise Q_k is small, round-off problems result in $P_{k|k-1}$ becoming non-positive-definite. Algorithms to improve stability include:
    - Cholesky Factorization => Decompose P as SS^T - this ensures that P stays positive definite.
    - Upper Triangular Decomposition => Decompose P as UDU^T. This uses less storage and computation than SS^T. Algorithms using UDU^T for Kalman filtering are widely available (Thornton (1976), Bierman (1977)).
    - LDL^T/LU Decomposition => Possibly the most efficient and robust of the lot, with specific pivoting and conditioning passes applied (Thornton (1976), Golub and Van Loan (1996), Bar-Shalom, Li, and Kirubarajan (2001), Higham (2002)).
24. Bayes' Factor Analysis of Predict/Update: Bayes' Factor based analysis of the Kalman predict/update and eventual parameter inference occurs at the measurement stage, thereby introducing/associating the minimal error Optimal Kalman Gain at that stage (see e.g., Masraliez and Martin (1977)).
    - Bayesian Estimation vs. Filtering Estimation => Bayesian treatment deals with the hidden parametric uncertainty (given a set of uncertain observations), whereas filtering treatments deal with de facto observation/measurement uncertainty (in order to be able to imply the set of hidden states and their uncertainties).
    - Bayesian A Priori Estimation => This is a simple, straight-up MLE pass-through, as there is neither an explicit dependence on the hidden state, nor is there any measurement involved. Joint a priori predict inference is simply a result of MLE'izing the normalized

convolution of the model-transformed a-posteriori updated state for the prior step and the model uncertainty.

- A Posteriori Probabilistic Estimate => This is a convolution of the measurement uncertainty and the a priori state predict, given the observations. This therefore lends itself perfectly to standard Bayesian analysis (in fact, the whole suite, including ABC via sufficient statistics), and also eventually to the optimal gain extraction.

25. Information Vector Approach to Kalman Filtering:

- The main steps in this approach are the following. The main advantage of this approach is that N successive measurements may be trivially summed up in the information space.
    - Transform the State Space variables onto the Information Space.
    - Perform the update/predict in the Information Space.
    - Revert back to the State Space.
- Information Vector Definitions: The a priori information matrix is given from

$$Y_{k|k-1} = {P_{k|k-1}}^{-1}$$

From this, the a priori information state vector is extracted as

$$\hat{y}_{k|k-1} = {P_{k|k-1}}^{-1} x_{k|k-1}$$

The a-posteriori information matrix is computed as

$$Y_{k|k} = {P_{k|k}}^{-1}$$

and the corresponding a posteriori information state vector is

$$\hat{y}_{k|k} = {P_{k|k}}^{-1} x_{k|k}$$

The measurement information matrix is given from

$$\hat{y}_{k|k-1} = I_k = {H_k}^T {R_k}^{-1} H_k$$

and the corresponding measurement state information vector is

$$i_k = {H_k}^T {R_k}^{-1} z_k$$

- Predict/Update Phase:
  - Information Matrix Update =>

$$Y_{k|k} = Y_{k|k-1} + I_k$$

implies that

$$Y_{k|k} = Y_{k|k-1} + \sum_{j=1}^{n} I_{k,j}$$

Information State Vector Update =>

$$\hat{y}_{k|k} = \hat{y}_{k|k-1} + i_k$$

implies that

$$\hat{y}_{k|k} = \hat{y}_{k|k-1} + \sum_{j=1}^{n} i_{k,j}$$

- State Space Extraction from the Information Matrix/Vector:

$$M_k = {M_k}^{-1} Y_{k-1|k-1} F_k$$

$$C_k = M_k \left[M_k + {Q_k}^{-1} \right]^{-1}$$

$$L_k = I - C_k$$

$$Y_{k|k-1} = L_k M_k {L_k}^T + C_k {Q_k}^{-1} C_k$$

$$\hat{y}_{k|k-1} = L_k \left[{F_k}^{-1}\right]^T \hat{y}_{k-1|k-1}$$

Continuous Time Kalman Filtering

1. Discrete/Continuous Linear Filter Variants:
   - Kalman => Discretized Model Prediction, and Discretized Measure Updates.
   - Bucy => Continuous Model Prediction, and Continuous/Simultaneous Measure Updates.
   - Hybrid => Continuous Model Prediction, and Discretized Measure Updates.
2. Kalman-Bucy Filter:
   - The continuous time Kalman Filter is expressed using the state equations (Jazwinski (1970), Bucy and Joseph (2005))

     $$\frac{\partial}{\partial t} x(t) = F(t)x(t) + B(t)u(t) + w(t)$$

     and

     $$z(t) = H(t)x(t) + v(t)$$

     and $Q(t)$ and $R(t)$ are the intensities of the white noises $w(t)$ and $v(t)$.
   - In the continuous time Kalman Filtering equations, since there is no explicit distinction between the predict step and the update step, the covariance of the noise process $R(t)$ is the same as the covariance of the innovation residual

$$\tilde{y}(t) = z(t) - H(t)\hat{x}(t)$$

(Kailath (1968)).

$$\frac{\partial}{\partial t}\hat{x}(t) = F(t)\hat{x}(t) + B(t)u(t) + K(t)[z(t) - H(t)\hat{x}(t)]$$

$$\frac{\partial}{\partial t}P(t) = F(t)P(t) + P(t)F^T(t) + Q(t) + K(t)H(t)K^T(t)$$

where the optimal gain is

$$K(t) = H^T(t)R^{-1}(t)$$

This equation is called the Riccati equation.

3. Hybrid Kalman Filter: Used for e.g., in physical systems, which typically are continuous-time models, while discrete-time measurements are taken for estimation.

$$\frac{\partial}{\partial t}x(t) = F(t)x(t) + B(t)u(t) + w(t)$$

and

$$w(t) \sim N\big(0, Q(t)\big)$$

$$z_k = H_k x_k + v_k$$

and

$$v(t) \sim N\big(0, R(t)\big)$$

- Predict Phase: Using

$$\frac{\partial}{\partial t}\hat{x}(t) = F(t)\hat{x}(t) + B(t)u(t)$$

start with

$$\hat{x}(t_{k-1}) = \hat{x}_{k-1|k-1}$$

and compute

$$\hat{x}_{k|k-1} = \hat{x}(t_k)$$

Likewise, using

$$\frac{\partial}{\partial t}P(t) = F(t)P(t) + P(t)F^T(t) + Q(t)$$

start with

$$P(t_{k-1}) = P_{k-1|k-1}$$

and compute

$$P_{k|k-1} = P(t_k)$$

- Update Phase: Same as continuous-time Kalman filter. The optimal Kalman Gain results are re-capped below:

$$K_k = P_{k|k-1}{H_k}^T\left[H_k P_{k|k-1}{H_k}^T + R_k\right]^{-1}$$

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k\left[z_k - H_k\hat{x}_{k|k-1}\right]$$

$$P_{k|k} = (I - K_k H_k) P_{k|k-1}$$

Non-linear Kalman Filtering

1. Definition: In non-linear Kalman filters, the state transition and the observation models are not linear, but differentiable, i.e.,

$$\boldsymbol{K_k = f(x_{k-1}, u_k) + w_k}$$

$$\boldsymbol{z_k = h(x_k) + v_k}$$

2. Extended Kalman Filter: The non-linearity may be handled by computing the evolution Jacobian at each step. Using these Jacobian tensors essentially linearizes the non-linear functions $\boldsymbol{f}$ and $\boldsymbol{h}$ around their current estimates.
   - Disadvantages of extended Kalman Filter approach:
     - Highly non-linear functions f and h give poor linearization performance/accuracy.
     - Jacobian calculations can become time consuming, although it may be significantly improved (in some cases) using the automatic differentiation techniques.
3. Unscented Kalman Filter (UKF): The challenges above are overcome by picking a set of points (called sigma points) around the mean. These points are propagated through non-linear functions, from which the sample mean and covariance are recovered more accurately (Julier and Uhlmann (1997)).
   - Predict Phase => The state and the covariance estimators are augmented with the process noise:

$$\hat{x}_{AUG,k-1|k-1} = \left[{\hat{x}_{k-1|k-1}}^T \quad \langle {w_k}^T \rangle \right]^T$$

$$P_{AUG,k-1|k-1} = \begin{bmatrix} P_{k-1|k-1} & 0 \\ 0 & Q_k \end{bmatrix}^T$$

- UKF Sigma Points Generation for Predict => The $2L + 1$ predict sigma points are generated as:

$$\chi_{0,k-1|k-1} = x_{AUG,k-1|k-1}$$

$$\chi_{i,k-1|k-1} = x_{AUG,k-1|k-1} + \left[\sqrt{(L+\lambda)P_{AUG,k-1|k-1}}\right]_i$$

for

$$i = 1, \cdots, L$$

and

$$\chi_{i,k-1|k-1} = x_{AUG,k-1|k-1} - \left[\sqrt{(L+\lambda)P_{AUG,k-1|k-1}}\right]_{i-L}$$

for

$$i = L+1, \cdots, 2L$$

Here the subscript $i, i-L$ refers to the column $i, i-L$ of the Cholesky decomposition of $(L+\lambda)P_{AUG,k-1|k-1}$, i.e.,

$$(L+\lambda)P_{AUG,k-1|k-1} = \left[\sqrt{(L+\lambda)P_{AUG,k-1|k-1}}\right]\left[\sqrt{(L+\lambda)P_{AUG,k-1|k-1}}\right]^T$$

- UKF Sigma Points Propagation and Re-combination:

- Compute the a priori sigma points from their a-posteriori counterparts computed in the previous step:

$$\chi_{i,k|k-1} = f\left(\chi_{i,k-1|k-1}\right)$$

for

$$i = 1, \cdots, L$$

- Re-combine the a priori sigma points to estimate the predicted state and covariance.

$$\hat{x}_{k|k-1} = \sum_{i=0}^{2L} W_{i,s}\chi_{i,k|k-1}$$

$$P_{k|k-1} = \sum_{i=0}^{2L} W_{c,s}\left[\chi_{i,k|k-1} - \hat{x}_{k|k-1}\right]\left[\chi_{i,k|k-1} - \hat{x}_{k|k-1}\right]^T$$

- The weights $W_{i,s}$ and $W_{c,s}$ are chosen to (see Wan and van der Merwe (2000)):
  - Control the sigma point spread.
  - Adapt to the x_k distribution.

UKF Update Phase – A Priori State Augmentation => Again, the state and the covariance a priori estimates are augmented with the measurement noise:

$$\hat{x}_{AUG,k|k-1} = \left[\hat{x}_{k|k-1}{}^T \quad \langle v_k{}^T \rangle\right]^T$$

$$P_{AUG,k|k-1} = \begin{bmatrix} P_{k|k-1} & 0 \\ 0 & R_k \end{bmatrix}^T$$

UKF Update - Sigma Points Generation for Predict => The $2L+1$ predict sigma points are generated as:

$$\chi_{0,k|k-1} = x_{AUG,k|k-1}$$

$$\chi_{i,k|k-1} = x_{AUG,k|k-1} + \left[\sqrt{(L+\lambda)P_{AUG,k|k-1}}\right]_i$$

for

$$i = 1, \cdots, L$$

and

$$\chi_{i,k|k-1} = x_{AUG,k|k-1} - \left[\sqrt{(L+\lambda)P_{AUG,k|k-1}}\right]_{i-L}$$

for

$$i = L+1, \cdots, 2L$$

Alternately, the UKF a priori sigma points themselves may be augmented as

$$\hat{x}_{AUG,k|k-1} = \left[\hat{x}_{k|k-1}{}^T \quad \langle v_k{}^T \rangle\right]^T \pm \left[\sqrt{(L+\lambda)R_{AUG,k}}\right]_i$$

- UKF Update – Final Phase => Project the sigma points through the observation function

$$\gamma_{i,k} = h\left(\chi_{i,k|k-1}\right) \forall\, i = 0, \cdots, 2L$$

and generate the following metrics. The predicted observation is

$$\hat{z}_k = \sum_{i=0}^{2L} W_{i,s} Y_{i,k}$$

with the predicted measure covariance being

$$P_{\Xi_k \Xi_k} = \sum_{i=0}^{2L} W_{i,c} [\gamma_{i,k} - \hat{z}_k][\gamma_{i,k} - \hat{z}_k]^T$$

The corresponding predicted measure cross covariance is computed as

$$P_{\mathrm{x}_k \Xi_k} = \sum_{i=0}^{2L} W_{i,c} [\chi_{i,k|k-1} - \hat{x}_{k|k-1}][\gamma_{i,k} - \hat{z}_k]^T$$

The optimal UKF Kalman gain is computed from the alternate optimizing formulation as

$$K_k = P_{\mathrm{x}_k \Xi_k} {P_{\Xi_k \Xi_k}}^{-1}$$

and the corresponding state updates and covariance updates are computed from

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k (z_k - \hat{z}_k)$$

and

$$P_{k|k} = P_{k|k-1} - K_k P_{\Xi_k \Xi_k} {K_k}^T$$

4. Splined Kalman Filter: By using spline representations for both state variates and the covariance, we can almost certainly do a better estimation job than either EKF or UKF.

Kalman Smoothing

1. Optimal Fixed Lag Smoother: This provides an optimal estimate of $\hat{x}_{k-N|k}$ for a fixed lag N, using the measurements of z_1 to z_k. It does it one-pass-at-a-time for each time step, but does multiple passes in all.
2. Fixed Lag Smoother – State Equation:

$$\begin{bmatrix} \hat{x}_{t|t} \\ \hat{x}_{t-1|t} \\ \vdots \\ \hat{x}_{t-N+1|t} \end{bmatrix} = \begin{bmatrix} I \\ 0 \\ \vdots \\ 0 \end{bmatrix} \hat{x}_{t|t-1} + \begin{bmatrix} 0 & I & 0 & \cdots & 0 \\ I & 0 & I & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & I \end{bmatrix} \begin{bmatrix} \hat{x}_{t-1|t-1} \\ \hat{x}_{t-1|t-1} \\ \vdots \\ \hat{x}_{t-N+1|t-1} \end{bmatrix} + \begin{bmatrix} K_0 \\ K_1 \\ \vdots \\ K_{N-1} \end{bmatrix} y_{t|t-1}$$

   Here $\hat{x}_{t|t-1}$ is estimated using the Standard Kalman Filter.

$$y_{t|t-1} = z(t) - H\hat{x}_{t|t-1}$$

   represents the Standard Innovation Residual, and $\hat{x}_{t-i|t}$ is the estimate of the hidden/latent state at time $t-i$ given the observation at a later instant t.
3. Fixed Lag Smoother – Gain Estimation: The optimal gains are given by the following equations for P_i and K_i:

$$P_i = P\{[F - KH]^T\}^i$$

   and

$$K_i = P_i H^T \{HPH^T + R\}^{-1}$$

   Here P and K are the prediction error covariance and the gain, respectively, of the standard Kalman Filter (i.e., they correspond to $P_{t|t-1}$).
4. Fixed Lag Smoother – Error Covariance: Define

$$P_i = Covariance\left[\left(\hat{x}_{t-i} - \hat{x}_{t-i|t}|z_1, \cdots, z_t\right)\right]$$

The estimation $\hat{x}_{t-i}$ improves by the amount

$$P_i = P_{i|i-1} - \sum_{j=0}^{i}\left[P_j H^T\{HPH^T + R\}^{-1} H\left(P_i{}^T\right)\right]$$

5. Fixed Interval Smoother: The fixed interval smoother provides the estimate of $\hat{x}_{k|n}$ for $k < n$ using measurements from a fixed interval z_1 to z_n. This is also referred to as "Kalman smoothing".
6. Rauch-Tung-Striebel (RTS) Fixed Interval Smoother: This smoother uses 2 passes (Rauch, Tung, and Striebel (1965)).
   - The first pass is the regular forward Kalman pass. The filtered state estimates $\hat{x}_{k|k}$ and the covariance estimates $P_{k|k}$ are saved for the backward pass.
     - The backward pass computes the smoothed state estimate $\hat{x}_{k|n}$ and the covariance estimate $P_{k|n}$:

$$\hat{x}_{k|n} = \hat{x}_{k|k} + C_k\left[\hat{x}_{k+1|k} - \hat{x}_{k+1|k}\right]$$

$$P_{k|n} = P_{k|k} + C_k\left[P_{k+1|k} - P_{k+1|k}\right]C_k{}^T$$

$$C_k = P_{k|k}F_k{}^T P_{k-1|k}{}^{-1}$$

7. Fixed Interval Smoother: Modified - Bryson-Frazier Smoother: This technique also uses the regular Kalman filter for the forward pass, followed by a recursive backward pass that avoids finding the inverse of the covariance matrix.
8. Fixed Interval Smoother: Minimum-Variance Smoother: This is also a two-pass smoother that achieves the theoretically best possible error performance using the variance error minimizer (Einicke (2006)).
   - The forward pass is the one-step-ahead regular Kalman filter:

$$\hat{x}_{k+1|k} = F_k \hat{x}_{k|k-1} + K_k\left[z_k - H_k \hat{x}_{k|k-1}\right]$$

- Backward Pass => Set

$$\alpha_k = S_k^{-\frac{1}{2}}\left[z_k - H_k \hat{x}_{k|k-1}\right]$$

then time-reverse α_k to compute β_k and the estimate

$$y_{k|N} = z_k - R_k \beta_k$$

- Cross-check – Recovery of the Kalman Filter Formulation => Taking the causal part, i.e., setting

$$N = k$$

you get

$$y_{k|k} = z_k - R_k S_k^{-\frac{1}{2}} \alpha_k$$

which is the same as the original minimum variance Kalman filter.

- Advantages of the minimum variance Smoother:
  - Underlying distribution need not be Gaussian (unlike RTS).
  - Relatively easily extended to continuous time (Einicke (2007), Einicke (2009)). Continuous time uncertainty can be accommodated by adding a positive definite term to the Riccati equation (see Kalman-Bucy filter above) (Einicke, Ralston, Hargrave, Reid, and Hainsworth (2008)).
  - The structure of α_k and β_k makes it relatively easily integratable with MLE-based state-space parameters.
  - Relatively easy to incorporate step-wise linearization.

References

- Anderson, B. D. O., and J. B. Moore (1979): *Optimal Filtering* **Prentice Hall** New York.
- Andreasen, M. M. (2008): Non-linear DSGE Models, The Central Difference Kalman Filter, and The Mean Shifted Particle Filter.
- Bar-Shalom, Y., X. R. Li., and T. Kirubarajan (2001): *Estimation with Applications to Tracking and Navigation* **John Wiley & Sons** New York.
- Bierman, G. J. (1977): *Factorization Methods for Discrete Sequential Estimation* **Academic Press**.
- Bucy, R. S., and P. D. Joseph (2005): *Filtering for Stochastic Processes with Applications to Guidance (2nd edition)* **AMS Chelsea Publ**.
- Carmi, A., P. Gurfil, and D. Kanevsky (2010): Methods for Sparse Signal Recovery using Kalman Filtering with embedded pseudo-measurement norms and quasi-norms *IEEE Transactions on Signal Processing* **58 (4)** 2405-2409.
- Einicke, G. A. (2006): Optimal and Robust Non-causal Filter Formulations *IEEE Trans. Signal Processing* **54 (3)** 1069-1077.
- Einicke, G. A. (2007): Asymptotic Optimality of the Minimum Variance Fixed Interval Smoother *IEEE Trans. Signal Processing* **55 (4)** 1543-1547.
- Einicke, G. A., J. C. Ralston, C. O. Hargrave, D. C. Reid, and D. W. Hainsworth (2008): Longwall Mining Automation: An Application of Minimum Variance Smoothing *IEEE Control Systems Magazine* **28 (6)** 1543-1547.
- Einicke, G. A. (2009): Asymptotic Optimality of the Minimum Variance Fixed Interval Smoother *IEEE Trans. Signal Processing* **54 (12)** 2904-2908.
- Fruhwirth, R. (1987): Reversible-jump Markov Chain Monte-Carlo Computation and Bayesian Model Determination *Nucl. Instrum. Methods* **A262** 444-450.
- Golub, G. H., and C. F. Van Loan (1996): *Matrix Computations* **Johns Hopkins Studies in Mathematical Sciences** Baltimore.
- Hamilton, J. (1994): *Time Series Analysis* **Princeton University Press**.

- Higham, N. J. (2002): *Accuracy and Stability of Numerical Algorithms (2nd Ed)* **Society for Industrial and Applied Mathematics** Philadelphia.
- Jazwinski, A. H. (1970): *Stochastic Processes and Filtering Theory* **Academic Press** New York.
- Julier, S. J., and J. K. Uhlmann (1997): A new Extension of the Kalman Filter to Nonlinear Systems *Int. Symp. Aerospace/Defense Sensing, Simulation, and Controls* **3**.
- Kailath, T. (1968): An Innovation Approach to Least-Squares Estimation Part I: Linear Filtering in Additive White Noise *IEEE Transactions on Automatic Control* **13 (6)** 646-655.
- Kalman Filter (Wiki): Wikipedia Entry for Kalman Filter.
- Lauritzen, S. L. (1981): Time Series Analysis in 1880 – A Discussion of the Contributions made by T. N. Thiele *International Statistical Review* **49** 319-333.
- Lauritzen, S. L. (2002): *Thiele: Pioneer in Statistics* **Oxford University Press**.
- Masraliez, C. J., and R. D. Martin (1977): Robust Bayesian Estimation for the Linear Model and Robustifying the Kalman Filter *IEE Transactions Automatic Control*.
- Matisko, P., and V. Havlena (2012): Optimality Tests and Adaptive Kalman Filter *Proceedings of the 16th IFAC System Identification Symposium* **Brussels**.
- Rajamani, M. R. (2007): *Data-based Techniques to improve State Estimation in Model Predictive Control* PhD Thesis **University of Wisconsin-Madison**.
- Rajamani, M. R., and J. B. Rawlings (2009): Estimation of the Disturbance Structure from Data using Semi-definite Programming and Optimal Weighting **45** 142-148.
- Rauch, H. E., F. Tung, and C. T. Striebel (1965): Maximum Likelihood Estimates of Linear Dynamic Systems *Automatica AIAA* **3 (8)** 1445-1450.
- Roweis, S., and Z. Ghahramani (1999): A unified Review of Linear Gaussian Models *Neural Comput.* **11 (2)** 305-345.
- Stratonovich, R. L. (1959a): Optimum non-linear Systems which bring about a separation of a signal with constant Parameters from Noise *Radiofizika* **2 (6)** 892-901.
- Stratonovich, R. L. (1959b): On the theory of Optimal non-linear Filtering of Random Functions *Theory of Probability and its Applications* **4** 223-225.
- Stratonovich, R. L. (1960a): Application of the Markov Processes to Optimal Filtering *Radio Engineering and Electronic Physics* **5 (11)** 1-19.

- Stratonovich, R. L. (1960b): Conditional Markov Processes *Theory of Probability and its Applications* **5** 156-178.
- Strid, I., and K. Walentin (2009): Block Kalman Filtering for large-scale DSGE Models *Computational Economics* **33 (3)** 277-304.
- Thornton, C. L. (1976): Triangular Covariance Factorizations for Kalman Filtering *NASA Technical Memorandum* **33-798**.
- Vaswani, N. (2008): Kalman Filtered Compressed Sensing *15th International Conference on Image Processing*.
- Wan, E. A., and R. van der Merwe (2000): The Unscented Kalman Filter for Nonlinear Estimation.
- Wolpert, D. M. (1996): Forward Models for Physiological Motor Control *Neural Networks* **9 (8)** 1265-1279.

Particle Filtering

1. Definition: Simulation based model estimation technique – also called sequential Monte Carlo Method (Doucet, De Freitas, and Gordon (2001), Particle Filter (Wiki)).
2. State Space Evolution as Kolmogorov Density Estimation: Posterior probability may be modeled as a series histogram or a series of a set of weighted particles that evolve through time.
3. Advantages of using Particle Filtering:
   - It is faster than MCMC if the sampling/sample rejections are done well. The reason for this is that, while particle filtering computes $p(x_{k|k-1}|y_k)$, MCMC models attempt to compute $p(x_{k|k-1}|y_0, \cdots, y_k)$.
   - It is easily extended to a wider variety of tractable priors/posteriors, as well as diverse types of state space functions.
   - With sufficient number of samples, particle filters can be made much more accurate than EKF/UKF.
4. Kalman vs. EKF/UKF vs. Particle Filtering vs. MCMC:
   - Kalman => HMM + Linear Model/Measurement Processes + Gaussian Error Terms
   - EKF/UKF => HMM + non-linear Model/Measurement Processes + Gaussian Error Terms
   - Particle => HMM + non-linear Model/Measurement Processes + non-Gaussian Error Terms
   - MCMC => Hidden, non-Markov + non-linear Model/Measurement Processes + non-Gaussian Error Terms
5. Sufficient Statistics for Target Distributions: This is one relatively straightforward way to incorporate splining. Splines may also be used to represent the sampled, particle filtered Kolmogorov distributions.

6. Sequential Importance Re-sampling (SIR): Here the posterior filtering distribution $p\left(x_{k|k-1}|y_0,\cdots,y_k\right)$ may be represented as a weighted set of P particles $w_k: \{w_{k,0},\cdots,w_{k,L},\cdots,w_{k,P}\}$ where

$$L \in \{1,\cdots,P\}$$

(Gordon, Salmond, and Smith (1993)). SIR is also referred to as ***Sampling Importance Re-sampling***.

7. Practical SIR: In practice, however, given that $p\left(x_{k|k-1}|y_0,\cdots,y_k\right)$ is not known, a proposal distribution $\pi\left(x_k|x_{L,0;k},y_{0,k}\right)$ is chosen instead. A sequential re-sampling algorithm will be needed to generate the appropriate weights.
   - Transition Density as the Proposal SIR Distribution => The discretized version of $\pi\left(x_k|x_{L,0;k},y_{0,k}\right)$ provides a set of weights, and is referred to as the importance function. The transition prior probability density distribution

$$\pi\left(x_k|x_{L,0;k},y_{0,k}\right) = p\left(x_{k|k-1}\right)$$

     forms a convenient importance function, as it is easy to draw particles (samples) from it. It also has a few other advantages to be seen later.
   - SIR + Transition Density as Importance Function is also called bootstrap filtering or condensation (conditional density) algorithm.
8. SIR Re-sampling: Re-sampling is used to avoid degeneracy in the algorithm, i.e., avoiding the situation where all but one of the importance weights are almost zero. A variation of the algorithm, called stratified sampling (Kitagawa (1996)), is optimal in terms of variance.
9. SIR Algorithm:
   - #1 => Draw

$$L \in \{1,\cdots,P\}$$

samples from the proposal distribution $\pi\left(x_k|x_{L,0;k}, y_{0,k}\right)$ and update the importance weights up to a normalizing constant:

$$\widehat{w}_{L,k} = \widehat{w}_{L,k-1}\frac{p\left(y_k|x_{L,k}\right)p\left(x_{L,k}|x_{L,k-1}\right)}{\pi\left(x_k|x_{L,0;k}, y_{0,k}\right)}$$

Note that if we use the transition probability distribution $p\left(x_{L,k}|x_{L,k-1}\right)$ as the importance function $\pi\left(x_k|x_{L,0;k}, y_{0,k}\right)$, then the above reduces to

$$\widehat{w}_{L,k} = \widehat{w}_{L,k-1}p\left(y_k|x_{L,k}\right)$$

- #2 => Compute the normalized importance weights as

$$w_{L,k} = \frac{\widehat{w}_{L,k}}{\sum_{j=1}^{p}\widehat{w}_{j,k}}$$

and the effective number of particles as

$$\widehat{N}_{Eff} = \frac{1}{\sum_{j=1}^{p}\widehat{w}_{j,k}{}^{2}}$$

- #3 => If

$$\widehat{N}_{Eff} < \widehat{N}_{Threshold}$$

re-sample as follows:

  - Draw *P* particles from the current set of probabilities proportional to their weights.
  - Replace the current set with the new one.
  - Set

$$w_{L,k} = \frac{1}{P}$$

In effect, this prunes out the low contributors.

10. Sequential Importance Sampling (SIS): This is the same as SIR, but without the re-sampling stage. It is also called the "direct version" – slight modification is needed in the algorithm to avoid re-sampling.
11. SIS Steps:
    - #1: Set

$$n = 0$$

the number of particles counted so far. Uniformly choose an index L from $(1, \cdots, P)$.

    - #2: Generate a test $\hat{x}$ from the distribution $p\left(x_{L,k} | x_{L,k-1}\right)$, and generate the corresponding probability of a given $\hat{y}$ from

$$p(\hat{y}|\hat{x}) = p(y|x{:}\, y_k|\hat{x})$$

where y_k is the k[th] measured value.

    - #3: Generate another uniform p in $[0, 1]$. If

$$u > p(\hat{y})$$

then re-generate another L and continue from #1. Otherwise, save $\hat{x}$ as $x_p(k|k)$ and increment n; continue till

$$n = P$$

then quit. This technique is based off of rejection-sampling based optimal filter (Blanco, Gonzalez, and Fernandez-Madrigal (2008), Blanco, Gonzalez, and Fernandez-Madrigal (2010))

12. Particle Filter Extensions:
    - Auxiliary Particle Filter => Pitt and Shepard (1999)
    - Regularized Auxiliary Particle Filter => Liu, Wang, and Ma (2011)
    - Hierarchical/Scalable Particle Filter => Canton-Ferrer, Casas, and Pardas (2011)
    - Rao-Blackwellized Particle Filter => Doucet, De Freitas, Murphy, and Russell (2000)
    - Econometric Particle Filter => Flury and Shephard (2008)

References

- Blanco, J. L., J. Gonzalez, and J. A. Fernandez-Madrigal (2008): An Optimal Filtering Algorithm for Non-parametric Observation Models in Robot Localization, *IEEE International Conference on Robotics and Automation (ICRA '08)* 461-466.
- Blanco, J. L., J. Gonzalez, and J. A. Fernandez-Madrigal (2010): An Optimal Filtering Algorithm for Non-parametric Observation Models: Applications to Localization and SLAM *International Journal of Robotics Research (IJRR)* **29 (14)** 1726-1742.
- Canton-Ferrer, C., J. R. Casas, and M. Pardas (2011): Human Motion Capture using Scalable Body Models *Computer Vision and Image Understanding* **115 (10)** 1363-1374.
- Doucet, A., N. De Freitas, K. Murphy, and S. Russell (2000): Rao-Blackwellised particle filter for dynamic Bayesian Networks *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence* 176-183.
- Doucet, A., N. De Freitas, and N. J. Gordon (2001): *Sequential Monte-Carlo Methods in Practice* **Springer**.
- Flury, R., and N. Shepard (2008): Bayesian Inference based only on Simulated Likelihood: Particle Filter Analysis of Dynamic Economic Models *OFRC Working Papers Series 2008fe32* **Oxford Financial Research Centre**.

- Gordon, N. J., D. J. Salmond, and A. F. M. Smith (1993): Novel Approach to Non-linear/Non-Gaussian State Space Estimation *IEEE Proceeding F on Radar and Signal Processing* **140 (2)** 107-113.
- Liu, J., W. Wang, and F. Ma (2011): A Regularized Auxiliary Particle Filtering Approach for System State Estimation and Battery Life Prediction *Smart Materials and Structures* **20 (7)** 1-9.
- Particle Filter (Wiki): Wikipedia Entry for Particle Filter.
- Pitt, O., and N. Shepard (1999): Filtering via Simulation: Auxiliary Particle Filters *Journal of American Statistical Association* **94 (446)** 590-591.

Regression Analysis

Linear Regression

1. Calibrator Estimator Problem Set: Techniques presented here are useful for the following set of problems:
   - Regression Analysis
   - Density Estimation
   - Multi-variate (Polychotomous) Logistic Regression
   - Survival Analysis
   - Spectral Density Estimation
2. Regression Terminology:

$$\vec{Y} = [\vec{\beta}]\vec{X} + \vec{\varepsilon}$$

   - Names for $\vec{Y}$ => Regressand, endogenous variables, response variables, measured variables, dependent variables (note that none of these terms explicitly invoke causality).
   - Names for $\vec{X}$ => Regressor, exogenous variable, predictor variable, covariates, explanatory variables, input variables, independent variables, design matrix.
   - Names for $[\vec{\beta}]$ => Parameter matrix, effects matrix, matrix of regression coefficients.
   - Names for $\vec{\varepsilon}$ => Error term, disturbance, noise.
3. Generalized Linear Regression (GLR): GLR optimizes the log likelihood function in place of least squares used in linear regression.
   - The fit equation can be re-cast in a very generalized form, thus the objective function can be of the form

$$\eta_i = g(\mu_i) = \sum_{j=1}^{n} a_j B_j(x_i)$$

not just

$$y_i = \sum_{j=1}^{n} a_j B_j(x_i)$$

4. Some GLR re-casting examples:
   - Poisson Regression =>

$$g(\mu_i) = \log \mu_i$$

   - Logistic Regression =>

$$g(\mu_i) = \log \frac{\mu_i}{1 - \mu_i}$$

5. Generalized Additive Models: Here the representative space functionals are a set/mixture of the additive state functionals.
   - Further, the smoothing formulation can be re-cast as a confidence band estimation/optimization problem with the 2 extraneously specified credible band limits C_1 and C_2, where

$$\sum_{i=1}^{n} \left[\hat{\mu}(x_i) - \sum_{j=1}^{n} a_j B_j(x_i) \right]^2 \leq C_1$$

and

$$B^T D^T D B \leq C_2$$

6. Regression Hat Matrix: The Regression Hat Matrix can be decomposed or represented in terms of the well-known Demmler-Reinsch basis functionals, and therefore derive from all its spectral properties and insights.
7. Non-Gaussian Priors: In this case, the smoothing estimation process is called the Generalized Linear Model.

Assumptions underlying Basic Linear Regression

1. Weak Exogeneity of Predictor Variables: This assumption indicates that $\vec{X}$ is fixed, thereby the sub-set being limited to measurement, not observation. This assumption may be lightened by assuming an errors-in-variables approach to linear regression formulation.
2. Linearity of the Predictor-Response Relationship: Given that there is a choice in the basis-functions, this is not too restrictive. However, over-fitting should be regularized out (e.g., by using ridge/lasso/Bayesian regression techniques).
3. Constant Response Variable Variance (Homoscedasticity): This means that different response variables have the same variance in their errors, regardless of the predictors. Sometimes, this impact may be reduced by using a transformation of the response variable (e.g., using a log normal link function).
4. Independence of Errors: It is also assumed that the errors of the response variables are uncorrelated with each other.
5. Lack of multi-collinearity: This assumption states that the design matrix $\vec{X}$ must be of full rank. Under special circumstances, multi-collinearity may be handled using an adjustment to the formulation (e.g., if $\vec{X}$ has effect sparsity). In general, more computationally intensive iterated algorithms for parameter estimation (such as those used in generalized linear models) do not suffer from this, as it is typical when handling categorical predictors to introduce a separate indicator variable predictor for each category (which inevitably introduces multi-collinearity).

Multivariate Regression Analysis

1. Multi-variate Regression Analysis vs. Errors-in-Variables - Setup: Say $\{x_0, \cdots, x_j, \cdots, x_{m-1}, y_0, \cdots, y_i, \cdots, y_{n-1}\}$ contains the measurement set, where $\{x_j\}_{j=0}^{m-1}$ is the set of predictor variables, and $\{y_i\}_{i=0}^{n-1}$ is the set of response variables. We propose

$$\hat{y}_i = \sum_{j=0}^{m-1} \beta_{i,j} x_j$$

as the sequence set of regressors, and attempt to minimize $\langle S_i \rangle$, where

$$S_i = [\hat{y}_i - y_i]^2$$

2. Multi-variate Regression Analysis vs. Errors-in-Variables - Formulation:

$$S_i = [\hat{y}_i - y_i]^2$$

$$= \left[\sum_{j=0}^{m-1} \beta_{i,j} x_j - y_i\right]^2$$

$$= \left[\beta_{i,k} x_k + \sum_{\substack{j=0; \\ j \neq k}}^{m-1} \beta_{i,j} x_j - y_i\right]^2$$

$$= {\beta_{i,k}}^2 {x_k}^2 + \left[\sum_{\substack{j=0; \\ j \neq k}}^{m-1} \beta_{i,j} x_j - y_i\right]^2 + 2\beta_{i,k} x_k \left[\sum_{\substack{j=0; \\ j \neq k}}^{m-1} \beta_{i,j} x_j - y_i\right]$$

$$\langle S_i \rangle = {\beta_{i,k}}^2 \langle {x_k}^2 \rangle + \left\langle \left[\sum_{\substack{j=0; \\ j \neq k}}^{m-1} \beta_{i,j} x_j - y_i\right]^2 \right\rangle + 2\beta_{i,k} \left[\sum_{\substack{j=0; \\ j \neq k}}^{m-1} \beta_{i,j} \langle x_j x_k \rangle - \langle x_k y_i \rangle\right]$$

$$\frac{\partial\langle S_i\rangle}{\partial\beta_{i,k}} = 2\beta_{i,k}\langle {x_k}^2\rangle + 2\left[\sum_{\substack{j=0;\\ j\neq k}}^{m-1}\beta_{i,j}\langle x_j x_k\rangle - \langle x_k y_i\rangle\right]$$

$$\frac{\partial\langle S_i\rangle}{\partial\beta_{i,k}} = 0$$

implies

$$\sum_{j=0}^{m-1}\beta_{i,j}\langle x_j x_k\rangle = \langle x_k y_i\rangle$$

- Further

$$\frac{\partial^2\langle S_i\rangle}{\partial{\beta_{i,k}}^2} = 2\langle {x_k}^2\rangle > 0$$

thereby indicating that the optimization is a minimum.

- Compact Formulation:

$$\beta X X^T = X^T Y$$

implies

$$\beta = \frac{X^T Y}{X X^T}$$

3. <u>Multi-variate Regression Analysis – Explicit Measurement Error plus Errors-in-Variables</u>:
   - We now set

$$\hat{y}_i = \sum_{j=0}^{m-1} \beta_{i,j} x_j + \varepsilon_i$$

implies that

$$\beta = \frac{X^T Y - X^T \varepsilon}{X X^T}$$

where ε_i is the corresponding component measurement error. If

$$X^T \varepsilon = 0$$

i.e., if ε is uncorrelated with X, we recover the compact formulation as before.

Multivariate Predictor/Response Regression

1. Nomenclature:
   - $$i = 0, \cdots, n-1$$

     Response Variables:

     $$\{y_i\}_{i=0}^{n-1}$$

   - $$j = 0, \cdots, m-1$$

     Response Variables:

$$\{x_j\}_{j=0}^{m-1}$$

- Observations:

$$p = 0, \cdots, q-1$$

2. Linear Regression – Set up:

$$\hat{y}_i = \sum_{j=0}^{m-1} \beta_{ij} x_j$$

$$\varepsilon_i = y_i - \hat{y}_i = y_i - \sum_{j=0}^{m-1} \beta_{ij} x_j$$

$$S_i = {\varepsilon_i}^2 = \left[y_i - \sum_{j=0}^{m-1} \beta_{ij} x_j \right]^2$$

3. k-Term Separation:

$$S_I = {\beta_{ik}}^2 \sum_{p=0}^{q-1} {x_{kp}}^2$$

$$S_{II} = 2\beta_{ik} \sum_{p=0}^{q-1} x_{kp} \left[\sum_{j=0; j\neq k}^{m-1} \beta_{ij} x_{jp} - y_{ip} \right]$$

$$S_{III} = \sum_{p=0}^{q-1} \left[\sum_{j=0; j\neq k}^{m-1} \beta_{ij} x_{jp} - y_{ip} \right]$$

$$S = S_I + S_{II} + S_{III}$$

4. Minimizing Optimizer Set up:

$$\frac{\partial S}{\partial \beta_{ik}} = \frac{\partial S_I}{\partial \beta_{ik}} + \frac{\partial S_{II}}{\partial \beta_{ik}} + \frac{\partial S_{III}}{\partial \beta_{ik}} = \sum_{p=0}^{q-1} \left[\left(\sum_{j=0}^{m-1} \beta_{ij} x_{kp} x_{jp} \right) - x_{kp} y_{ip} \right] = 0$$

Thus

$$\sum_{j=0}^{m-1} \beta_{ij} \left[\left(\sum_{p=0}^{q-1} x_{kp} x_{jp} \right) - x_{kp} y_{ip} \right] = \sum_{p=0}^{q-1} x_{kp} x_{ip}$$

Setting

$$\sum_{p=0}^{q-1} x_{kp} x_{jp} = \alpha_{jk}$$

and

$$\sum_{p=0}^{q-1} x_{kp} y_{ip} = \gamma_{ik}$$

the above becomes

$$\sum_{j=0}^{m-1} \beta_{ij} \alpha_{jk} = \gamma_{ik}$$

5. Second derivative – minimizing optimizer verification:

$$\frac{\partial^2 S}{\partial {\beta_{ik}}^2} = \sum_{p=0}^{q-1} {x_{kp}}^2$$

Given that the predictor variance is greater than zero, the β_{ik} that results from

$$\frac{\partial S}{\partial \beta_{ik}} = 0$$

corresponds to a minima.

6. Expanded Formulation: Considering just the regression co-efficient set $\{\beta_{ij}\}_{j=0}^{m-1}$, we get

$$\begin{pmatrix} \alpha_{0,0} & \cdots & \alpha_{0,m-1} \\ \vdots & \ddots & \vdots \\ \alpha_{m-1,0} & \cdots & \alpha_{m-1,m-1} \end{pmatrix} \begin{pmatrix} \beta_{i,0} \\ \vdots \\ \beta_{i,m-1} \end{pmatrix} = \begin{pmatrix} \gamma_{i,0} \\ \vdots \\ \gamma_{i,m-1} \end{pmatrix}$$

7. The α Matrix: The α matrix does not depend on i, i.e., it is dependent only on j and k, therefore it can be pre-computed across the observation set $\{x_j\}_{j=0}^{m-1}$ and re-used across all β's and γ's.
   - Given that

$$\sum_{p=0}^{q-1} x_{kp} x_{jp} = \alpha_{jk}$$

   the α matrix can be re-written as $X^T X$. This way the observations are directly accommodated, without the need to extract variances and co-variances.
8. The β Matrix: Recall that the β matrix is solved a single y set at a time, i.e., β_{ik} across all k is specified in a single linear system (corresponding to the γ matrix).
9. The γ Matrix: The γ matrix does not depend on j, i.e., it is dependent only on i and k. Thus, the pre-computation of γ is not as automatic as for the α matrix.
   - Given that

$$\sum_{p=0}^{q-1} x_{kp} y_{ip} = \gamma_{ik}$$

the γ matrix can be re-written as $X^T Y$. This way the observations are directly accommodated, again without the need to parametrically extract variances and co-variances.

10. Bringing it all together: Finally

$$\sum_{j=0}^{m-1} \beta_{ij} \alpha_{jk} = \gamma_{ik}$$

may be re-written as

$$X^T X \beta = X^T Y$$

Thus

$$\beta = (X^T X)^{-1} X^T Y$$

11. Variance/Covariance Form in OLS Formulation: The coefficient of β in the least-squares expansion term is twice the predictor-response covariance, i.e.

$$\beta_{Coeff} = 2\langle xy \rangle$$

The coefficient of β^2 in the least squares expansion term is the response-response covariance, i.e.

$$(\beta^2)_{Coeff} = 2\langle x_i x_j \rangle$$

Therefore

$$\beta_{OLS} = \frac{\langle xy \rangle}{\langle x_i x_j \rangle}$$

Also remember that the usage of the squared term in OLS causes the formulation occur with the second moment terms at most – thereby resulting in a nice workable formulation with Gaussian distribution.

OLS on Basis Spline Representation

1. Base Formulation: We start with the point squared-error term:

$$S_p = \left[y_p - \sum_{i=0}^{n-1} \beta_{ij} f_i(x_p) \right]^2$$

After expanding and k-separating, this results in:

$$S = \sum_{p=0}^{q-1} S_p = \sum_{p=0}^{q-1} {y_p}^2 + {\beta_k}^2 \sum_{p=0}^{q-1} {f_k}^2(x_p) + 2\beta_k \left[\sum_{\substack{i=0 \\ i \neq k}}^{n-1} \beta_i \left\{ \sum_{p=0}^{q-1} f_i(x_p) f_k(x_p) \right\} - \sum_{p=0}^{q-1} f_k(x_p) y_p \right]$$
$$- 2 \sum_{\substack{i=0 \\ i \neq k}}^{n-1} \left[\beta_i \left\{ \sum_{p=0}^{q-1} f_i(x_p) y_p \right\} \right] + \sum_{\substack{j=0 \\ j \neq k}}^{n-1} \left(\sum_{\substack{i=0 \\ i \neq k}}^{n-1} \left[\beta_i \beta_j \left\{ \sum_{p=0}^{q-1} f_i(x_p) f_j(x_p) \right\} \right] \right)$$

Turning the point summands into expectations

$$\langle S\rangle = \langle y^2\rangle + {\beta_k}^2\langle {f_k}^2(x)\rangle + 2\beta_k\left[\sum_{\substack{i=0\\ i\neq k}}^{n-1}\beta_i\langle f_i(x)f_k(x)\rangle - \langle f_k(x)y\rangle\right] - 2\sum_{\substack{i=0\\ i\neq k}}^{n-1}[\beta_i\langle f_i(x)y\rangle]$$
$$+\sum_{\substack{j=0\\ j\neq k}}^{n-1}\left(\sum_{\substack{i=0\\ i\neq k}}^{n-1}[\beta_i\beta_j\langle f_i(x)f_j(x)\rangle]\right)$$

2. Basis Spline OLS Optimizer:

$$\frac{\partial\langle S\rangle}{\partial\beta_k} = 2\beta_k\langle {f_k}^2(x)\rangle + 2\left[\sum_{\substack{i=0\\ i\neq k}}^{n-1}\beta_i\langle f_i(x)f_k(x)\rangle - \langle f_k(x)y\rangle\right]$$

Further

$$\frac{\partial\langle S\rangle}{\partial\beta_k} = 2\langle {f_k}^2(x)\rangle > 0$$

so there is always a minimum.

$$\frac{\partial\langle S\rangle}{\partial\beta_k} = 0$$

implies that

$$\sum_{i=0}^{n-1}\beta_i\langle f_i(x)f_k(x)\rangle - \langle f_k(x)y\rangle = 0$$

resulting in

$$\beta = (F^TF)^{-1}F^TY$$

This is the basis spline equivalent of linear regression off of linear basis function, i.e.

$$\beta = (X^T X)^{-1} X^T Y$$

OLS on Basis Spline Representation with Roughness Penalty

1. Base Formulation: We start with the penalized point squared-error term

$$S_p = \left[y_p - \sum_{i=0}^{n-1} \beta_{ij} f_i(x_p) \right]^2 + \lambda \int_{x_{Left}}^{x_{Right}} \left[\frac{\partial^m \hat{\mu}(x)}{\partial x^m} \right]^2 dx$$

   Note that the roughness penalizer term on the right is independent of x_p, so should have no dependence on the point expectations.

2. Penalizing Optimizer Formulation:

$$\int_{x_{Left}}^{x_{Right}} \left[\frac{\partial^m \hat{\mu}(x)}{\partial x^m} \right]^2 dx$$

$$= {\beta_k}^2 \int_{x_{Left}}^{x_{Right}} \left[\frac{\partial^m f_k}{\partial x^m} \right]^2 dx + 2\beta_k \sum_{\substack{i=0 \\ i \neq k}}^{n-1} \beta_i \left(\int_{x_{Left}}^{x_{Right}} \left[\frac{\partial^m f_i}{\partial x^m} \right] \left[\frac{\partial^m f_k}{\partial x^m} \right] dx \right)$$

$$+ \sum_{\substack{i=0 \\ i \neq k}}^{n-1} \left\{ \sum_{\substack{j=0 \\ j \neq k}}^{n-1} \beta_i \beta_j \left(\int_{x_{Left}}^{x_{Right}} \left[\frac{\partial^m f_i}{\partial x^m} \right] \left[\frac{\partial^m f_j}{\partial x^m} \right] dx \right) \right\}$$

3. Full Basis Spline Penalizing Optimizer:

$$\langle S \rangle = {\beta_k}^2 \left(\langle {f_k}^2 \rangle + \lambda \int_{x_{Left}}^{x_{Right}} \left[\frac{\partial^m f_k}{\partial x^m} \right]^2 dx \right)$$
$$+ 2\beta_k \left\{ \sum_{\substack{i=0 \\ i \neq k}}^{n-1} \beta_i \left(\langle f_i f_k \rangle + \int_{x_{Left}}^{x_{Right}} \left[\frac{\partial^m f_i}{\partial x^m} \right] \left[\frac{\partial^m f_k}{\partial x^m} \right] dx \right) - \langle f_k y \rangle \right\} - 2 \sum_{\substack{i=0 \\ i \neq k}}^{n-1} \beta_i \langle f_i y \rangle$$
$$+ \sum_{\substack{i=0 \\ i \neq k}}^{n-1} \left\{ \sum_{\substack{j=0 \\ j \neq k}}^{n-1} \beta_i \beta_j \left(\langle f_i f_j \rangle + \int_{x_{Left}}^{x_{Right}} \left[\frac{\partial^m f_i}{\partial x^m} \right] \left[\frac{\partial^m f_j}{\partial x^m} \right] dx \right) \right\}$$

Here we substitute f_i for $f_i(x)$ for ease of parsing.

4. Full Basis Spline Penalizing Optimizer – Second Derivative:

$$\frac{\partial^2 \langle S \rangle}{\partial {\beta_k}^2} = 2 \left(\langle {f_k}^2 \rangle + \lambda \int_{x_{Left}}^{x_{Right}} \left[\frac{\partial^m f_k}{\partial x^m} \right]^2 dx \right) > 0$$

for

$$\lambda > 0$$

and

$$x_{Left} > 0$$

Thus, under these situations, a minimum is possible in S.

5. Full Basis Spline Penalizing Optimizer – Solution:

$$\beta = Q^{-1} F^T Y$$

where

$$Q_{ij} = \langle f_i(x) f_k(x) \rangle + \int_{x_{Left}}^{x_{Right}} \left[\frac{\partial^m f_i(x)}{\partial x^m}\right]\left[\frac{\partial^m f_k(x)}{\partial x^m}\right] dx$$

Extensions to Linear Regression Methodology

1. Generalized Linear Models (GLM): Here the response variables are bounded and/or discrete, thereby necessitating the usage of a link function g that relates the mean of the response variables to the predictors, i.e.

$$\langle Y \rangle = g(\beta' X)$$

   e.g., transformation between $(-\infty, +\infty)$ - the range of the linear predictor and the range of the linear response variables.
2. Common Examples of Usage under a GLM Setting:
   - Poisson Regression for Count Data
   - Logistic/Probit Regression for Binary Data
   - Multinomial logistic/probit regression for ordered data
   - Ordered probit regression for ordinal data
3. Hierarchical Linear Models: Also referred to as multi-level regression, this technique organizes the data into a hierarchy of regressions. The final response is hierarchical on the intermediate/bottom-most predictors.
4. Error-in-variables: This allows the predictor variables to modeled under error. This may cause β to be biased, but that is typically in the form of attenuation (meaning that the effects tend to go to zero).

Linear Regression Estimator Extensions

1. Objective of the Estimation Methods:
   - Computational Simplicity
   - Preferably result in a closed form solution
   - Robust with regards to heavy tailed distributions
   - Produce desirable statistical properties such as consistency and asymptotic efficiency with minimal theoretical assumptions/restrictions.
2. Ordinary Least Squares (OLS): As seen earlier

$$\beta = (X^T X)^{-1} X^T Y$$

   OLS estimator is unbiased and consistent if

$$\langle x_i \varepsilon_i \rangle = 0$$

   and if $\langle {\varepsilon_i}^2 \rangle$ is independent of i (the homoscedastic assumption).
3. Generalized Least Squares (GLS): GLS handles heteroscedasticity using a covariance matrix of the errors Ω. GLS minimizes a weighted sum of squared residuals $\{w_i\}$ of OLS, where

$$w_i \propto \frac{1}{Variance(\varepsilon_i)}$$

   GLS can be viewed as applying a linear transform to the data so that the OLS assumptions are met. GLS analysis finally produces

$$\beta = (X^T \Omega^{-1} X)^{-1} X^T \Omega^{-1} Y$$

4. Percentage Least Squares: Here OLS is performed on the percentage transforms of the predictors. Thus, as opposed to typical OLS where the errors are additive, here the errors become multiplicative.

5. Iteratively Re-weighted Least Squares: This technique is used where both heteroscedasticity and correlated errors exist, but their structures are unknown. The following are the typical steps:
   - First, a GLS/OLS is performed on a provisional covariance structure, and the residuals are obtained from a fit.
   - Using these residuals, an improved covariance structure is estimated.
   - A subsequent GLS is then performed to estimate the weights until convergence.
   - 1-2 iterations of the above should be enough to obtain β estimates usually.
6. Instrumental Variables (IV): If the regressors and the error variables are correlated, then regression is performed on a set of instrumental variables $\{z_i\}$ that have the property

$$\langle z_i \varepsilon_i \rangle = 0$$

   In this case β is given as

$$\beta = (X^T Z (Z^T Z)^{-1} Z^T X)^{-1} X^T Z (Z^T Z)^{-1} Z^T Y$$

7. Total Least Squares: This approach handles the predictor variables and the response variables more evenly, thereby automatically accommodating errors-in-variables.
8. MLE: This method is used when the errors belong to the distribution family $f(\theta)$ that is not Gaussian – if $f(\theta)$ is Gaussian, the method becomes OLS.
   - Least Absolute Deviation => Here $f(\theta)$ corresponds to the Laplace error distribution. Least absolute deviation is more robust and less sensitive to errors than OLS, but is less efficient.
9. Ridge/Lasso Regression: These are penalizing estimators that deliberately introduce bias into the estimation procedure. They generally produce lower mean-squared error than OLS, as well as handle multi-collinearity at times.
10. Adaptive Estimation: The adaptive estimation is used to estimate the ε distribution. Assuming that ε is independent of X, it estimates ε in the first step, and then uses MLE in the second step to estimate β.

11. Quantile Regression: Quantile regression techniques focus on the conditional quantiles of $\langle Y|X \rangle$ rather than the mean of $\langle Y|X \rangle$ as a linear function $\beta' X$ of the quantiles.
12. Mixed Model Estimators: Mixed model estimators handle the case where the dependencies between the different y_i's have a known structure, so the mixed models regress across them.
13. Principal Component Regression (PCR): PCR is used when the number of predictors is huge, or when there are strong correlations across the predictors. However, there is no a priori cause for the principal components to serve as the target predictors. To address this, the partial least squares regression can be used.
14. Least Angle Regression: This has been developed to handle high dimensional covariate vectors, or for situations where there are more covariates than observations.
15. Theil-Sen Estimator: In this estimation, the regression line is chosen to have the slope that is the mean of the slopes of the lines through the sample point, thus it is less sensitive to outliers. It is also a simple and robust technique.

Bayesian Approach to Regression Analysis

1. Penalizing Smootheners: Penalizing smootheners are the consequence of Bayes' estimation applied on the Quadratic Penalties with Gaussian Priors (also referred to with maxim "The Penalty is the Prior").
2. Inference Based Curve Fitting: Here, the target function is treated as a realization of a stochastic process. Hence the model hypothesis estimation, credible interval estimate, etc. are all automatically available as part of the Bayesian Inference Framework.
3. Bayesian Optimizing Inference Schemes: Are all optimizing inference schemes castable as Bayesian? Clearly, given that regression schemes are part of a calibration framework, they can be reduced to Bayesian formulation, with specific priors having been extracted for smoothing/optimizing regressors. How about other inference schemes?

Component Analysis

Independent Component Analysis (ICA) - Specification

1. Definition: Give me the independent source signals that cause the sequence of observations that I observe. This is a plain orthogonalization challenge.
2. Nomenclature:

$$x_i : i = 0, \cdots, n-1$$

are the n random variables and the vector X_k is the k[th] snapshot of all x_i's. After many observations

$$m \gg n$$

say that you have measured $\langle x_p x_q \rangle$ and $\langle {x_p}^2 \rangle$ for all

$$p, q \in (0, \cdots, n-1)$$

3. Problem Statement: Assume, without loss of generality, that x_p and x_q are mean-centered. We look for the orthogonal factor set $\{s_j\}_{j=0}^{n-1}$ such that

$$\langle s_k s_l \rangle = 0$$

for

$$k \neq l$$

and

$$\langle {s_k}^2 \rangle \neq 0$$

otherwise, for all

$$k, l \in (0, \cdots, n-1)$$

4. Formulation: Let

$$x_p = \sum_{j=0}^{n-1} a_{pj} s_j$$

and

$$x_q = \sum_{j=0}^{n-1} a_{qj} s_j$$

Thus

$$\langle x_p x_q \rangle = \sum_{j=0}^{n-1} a_{pj} a_{qj} \langle {s_j}^2 \rangle$$

and

$$\langle {x_p}^2 \rangle = \sum_{j=0}^{n-1} {a_{pj}}^2 \langle {s_j}^2 \rangle$$

- If you work in the correlation space of x_p and x_q instead of the covariance space, we work on the corresponding covariates ε_p and ε_q with

$$\langle {\varepsilon_p}^2 \rangle = 1$$

and

$$\langle \varepsilon_p \varepsilon_q \rangle = \rho_{pq}$$

recalling

$$\varepsilon_p = \frac{x_p}{\sum_{j=0}^{n-1} {x_j}^2}$$

In this case, the corresponding orthogonal factor equations become

$$\sum_{j=0}^{n-1} a_{pj} a_{qj} \langle {s_j}^2 \rangle = \rho_{pq}$$

and

$$\sum_{j=0}^{n-1} {a_{pj}}^2 \langle {s_j}^2 \rangle = 1$$

5. Under-determined Uniqueness of Orthogonality: From

$$\langle \varepsilon_p \varepsilon_q \rangle = \rho_{pq}$$

you have n^2 RHS values, and therefore that many equations. But there are $n^2 + n$ unknowns - n^2 from a_{pq}, and n from $\langle {s_j}^2 \rangle$.

- Implicit $\langle s_j \rangle = 0$ assumption => Bear in mind that, in addition to $\langle x_p x_q \rangle$ and $\langle {x_p}^2 \rangle$, we also know $\langle x_p \rangle$ (which, by our axioms, is zero). Thus, we have n equations for each

$$x_p = \sum_{j=0}^{n-1} a_{pj} s_j$$

and n unknowns $\langle s_j \rangle$, which we, arbitrarily for now, satisfy using

$$\langle s_j \rangle = 0$$

- One way of right determining the orthogonal equation unknowns => If we impose

$$\langle {s_j}^2 \rangle = 1$$

for all j, then you have n^2 equations and n^2 unknowns, thereby it is right-determined.

- n-D Basis Coefficients Calibration => Remember

$$\langle x_p x_q \rangle = \sum_{j=0}^{n-1} a_{pj} a_{qj}$$

and

$$\langle x_q x_p \rangle = \sum_{j=0}^{n-1} a_{qj} a_{pj}$$

Thus, these 2 are essentially the same set – this equivalence reduces/takes off $\frac{(n-1)(n-2)}{2}$ equations from the system, thereby providing that much degrees of freedom for orthogonalization.

- However, given the non-linearity inherent in the product terms, solving this system is still not trivial.

6. Alternate Formulation: Instead of specifying x_p and x_q in terms of s_j, we do the reverse.

$$s_p = \sum_{j=0}^{n-1} a_{pj} x_j$$

and

$$s_q = \sum_{j=0}^{n-1} a_{qj} x_j$$

$$x_p x_q = \left[\sum_{j=0}^{n-1} a_{pj} x_j\right]\left[\sum_{j=0}^{n-1} a_{qj} x_j\right] = \sum_{i=0}^{n-1}\sum_{j=0}^{n-1} a_{pi} a_{qj} x_i x_j$$

$$\langle s_p s_q \rangle = 0$$

for

$$\sum_{i=0}^{n-1}\sum_{j=0}^{n-1} a_{pi} a_{qj} \langle x_i x_j \rangle = 0$$

for

$$p \neq q$$

$$\langle s_p s_p \rangle = \langle {s_p}^2 \rangle$$

for

$$\sum_{i=0}^{n-1}\sum_{j=0}^{n-1} a_{pi}a_{qj}\langle x_i x_j\rangle = \langle {s_p}^2\rangle$$

for

$$p = q$$

Thus, this is no easier – the complexity is only more than with the original formulation!

7. Non-linearity Reduction in an Optimizing Framework: The above ICA is non-linear in the coefficients, but if you find a way to introduce optimization into this, you may be able to reduce it to a sequence of linear equations (given that it is quadratic non-linear).

Independent Component Analysis (ICA) - Formulation

1. Nomenclature:
   - Signal Components:

$$i = 0, \cdots, n-1 \Rightarrow \{S_i\}_{i=0}^{n-1}$$

   - Data Points:

$$p = 0, \cdots, q-1 \Rightarrow \{x_{jp}\}_{p=0}^{q-1}$$

   - The correspondingly observed x_j is

$$x_j = \sum_{i=0}^{n-1} a_{ij}S_i$$

where a_{ij} is the mixing matrix.

- Goal: For the m observations given by $\{x_{jp}\}_{p=0}^{q-1}$ estimate a_{ij} as well as S_i.

2. Estimation Technique #1 - Minimization of Mutual Information (MMI): MMI family of algorithms use metrics such as Kullback-Leibler divergence, maximum entropy etc. and minimize them.
3. Estimation Technique #2 - Maximization of non-Gaussianity: This family uses kurtosis/negentropy as the metric, and maximizes them using the calculated signal term S_i.
4. Common Estimation Steps across all Algorithms:
   - Mean-centering
   - Whitening (using Eigenization)
   - Dimensionality reduction
5. ICA Formulation Extensions:
   - Linear Noisy ICA:

$$x_j = \sum_{i=0}^{n-1} a_{ij} S_i + n_j$$

where

$$n_j \sim \mathcal{N}(0, \rho_{ij})$$

   - Non-linear ICA:

$$\vec{x} = f(\vec{S}|\theta) + \vec{n}$$

where $f(\vec{S}|\theta)$ is a non-linear mixer.

6. Binary ICA: In binary ICA, both the source and the monitors are both in the binary form

$$x_i = \vee_{j=0}^{n-1} (g_{ij} \wedge S_j)$$

where ∧ is the Boolean AND, ∨ is the Boolean OR, and

$$g_{ij} = 1$$

indicates that the i[th] source is observed by the j[th] monitor. Noise is not explicitly modeled – it may be treated as another source.

7. Binary ICA Solution – Continuous Heuristics: The simple, heuristic solution is to assume that the predictor and the response variables are continuous, extract a real-valued g_{ij}, then round it to imply the individual g_{ij}. This has, of course, been shown to be inaccurate in many cases.
8. Binary ICA Solution - Dynamic Programming: Here the observation matrix X is broken down into sub-matrices, and targeted inferences are run individually on them. This has been shown to be accurate under moderate levels of noise.

Principal Component Analysis

1. Definition: The orthogonal transformation that transforms the data to a new co-ordinate system such that the greatest variance by any projection of the data comes to lie along the first co-ordinate (the first principal component), the second greatest variance along the second co-ordinate, and so on.
2. Alternate Phenomenon Nomenclature: The following are the names PCA is known as under the individual sub-fields:
   - Quality Control => Karhunen-Loeve Transform (KLT)
   - Linear Algebra => Proper Orthogonal Decomposition (POD), Singular Value Decomposition (SVD) of the design matrix $\vec{X}$, Eigen-value Decomposition (EVD) of $X^T X$, Factor Analysis
   - Psychometrics => Eckert-Young theorem

- Meteorological Science => Schmidt-Mirsky theorem, Empirical Orthogonal Functions (EOF)
- Noise and Vibration => Empirical Eigen-function Decomposition, Empirical Component Analysis, Quasi-harmonic Modes, Spectral Decomposition
- Structural Dynamics => Empirical Modal Analysis

3. PCA Computation Results Representation: The result of the PCA computation typical contain the following:
   - Component Scores/Factor Scores => The transformed variable values corresponding to a particular point.
   - Loadings => The weight by which each standardized original variable should be multiplied to get the component score.
4. Canonical Correlation Analysis (CCA): CCA identifies the transformed co-ordinate systems that optimally describe the cross-variance between two data-sets.
5. PCA Nomenclature and Set up:
   - Observation set:

$$p = 0, \cdots, q - 1$$

   - Dimensions:

$$i = 0, \cdots, n - 1$$

   - For a target point $\vec{P}$, the co-ordinates are $\{x_{iP}\}_{i=0}^{n-1}$. Thus, point $\vec{P}$ has a unit vector

$$\hat{P} = \frac{x_{0,P}\hat{x}_0 + x_{1,P}\hat{x}_1 + \cdots + x_{i,P}\hat{x}_i + \cdots + x_{n-1,P}\hat{x}_{n-1}}{\sqrt{{x_{0,P}}^2 + {x_{1,P}}^2 + \cdots + {x_{i,P}}^2 + \cdots + {x_{n-1,P}}^2}}$$

   - The Target PCA vector is

$$\widehat{W} = w_0\hat{x}_0 + w_0\hat{x}_0 + \cdots + w_0\hat{x}_0 + \cdots + w_0\hat{x}_0 \Longrightarrow \{w_i\}_{i=0}^{n-1}$$

under the constraint

$$\sum_{i=0}^{n-1} {w_i}^2 = 1$$

6. Projection of $\vec{P}$ on $\widehat{W}$:

$$\vec{P} \cdot \widehat{W} = \sum_{i=0}^{n-1} w_i x_{iP}$$

Note

$$\langle \vec{P} \cdot \widehat{W} \rangle = \sum_{i=0}^{n-1} w_i \langle x_{iP} \rangle = 0$$

since

$$\langle x_{iP} \rangle = 0$$

Thus, the projection is also mean-centered.

7. Square Error of the Projection of $\vec{P}$ on $\widehat{W}$:

$$S_p = \left[\sum_{i=0}^{n-1} w_i x_{i,p}\right]^2 = \left[\sum_{i=0}^{n-1} w_i x_{i,p}\right]\left[\sum_{j=0}^{n-1} w_j x_{j,p}\right]$$

- k-separating this projection yields

$$S_p = \left[w_k x_{k,p} + \sum_{\substack{i=0 \\ i \neq k}}^{n-1} w_i x_{i,p} \right] \left[w_k x_{k,p} + \sum_{\substack{i=0 \\ i \neq k}}^{n-1} w_i x_{i,p} \right]$$

- Thus

$$S_p = {w_k}^2 {x_{k,p}}^2 + 2 w_k x_{k,p} \sum_{\substack{i=0 \\ i \neq k}}^{n-1} w_i x_{i,p} + \left[\sum_{\substack{i=0 \\ i \neq k}}^{n-1} w_i x_{i,p} \right]^2$$

- The True Variance (referred from now on as the unconstrained variance U) is the cumulated k-separated point errors across all the sample points

$$U = \sum_{p=0}^{q-1} S_p = {w_k}^2 \sum_{p=0}^{q-1} {x_{k,p}}^2 + 2 w_k \sum_{\substack{i=0 \\ i \neq k}}^{n-1} w_i \sum_{p=0}^{q-1} x_{i,p} x_{k,p} + \sum_{p=0}^{q-1} \left[\sum_{\substack{i=0 \\ i \neq k}}^{n-1} w_i x_{i,p} \right]^2$$

- Re-cast in another way:

$$U = \sum_{p=0}^{q-1} S_p = {w_k}^2 \sum_{p=0}^{q-1} {x_{k,p}}^2 + 2 w_k \sum_{\substack{i=0 \\ i \neq k}}^{n-1} w_i \sum_{p=0}^{q-1} x_{i,p} x_{k,p} + \left[\sum_{\substack{i=0 \\ i \neq k}}^{n-1} \sum_{\substack{j=0 \\ j \neq k}}^{n-1} \left(w_i w_j \sum_{p=0}^{q-1} x_{i,p} x_{j,p} \right) \right]$$

8. <u>Recast the Unconstrained Variance into Variance/Covariance Grouping</u>: By applying the corresponding expectation as opposed to sample accumulation, we get

$$U = {w_k}^2 \langle {x_k}^2 \rangle + 2 w_k \sum_{\substack{i=0 \\ i \neq k}}^{n-1} w_i \langle x_i x_k \rangle + \left[\sum_{\substack{i=0 \\ i \neq k}}^{n-1} \sum_{\substack{j=0 \\ j \neq k}}^{n-1} \left(w_i w_j \langle x_i x_j \rangle \right) \right]$$

9. Two methods to work out the details of PCA Solution:
    - The first (and the best) is to use the formulated unconstrained projected variance, and to incorporate the constraint using a Lagrange multiplier.
    - The second is to use the constraint explicitly right during the formulation, thereby doing away with the external constraint.

Principal Component Analysis – Constrained Formulation

1. The Constraint N: The U above is subject to the constraint

$$N = \sum_{i=0}^{n-1} {w_i}^2 = 1$$

   This forces the eventual solution to lie on the nD sphere with ordinates $\{w_i\}_{i=0}^{n-1}$.
2. Reconstitute the Constrained Optimizer: We optimize U subject to the constraint

$$N = \sum_{i=0}^{n-1} {w_i}^2 = 1$$

   To put it differently, we optimize V, where

$$V = \frac{U}{N}$$

   thereby the scaling with N automatically ensures that the constraint is applied.
3. Formulate $\frac{\partial V}{\partial w_k}$:

$$\frac{\partial V}{\partial w_k} = \frac{1}{N^2}\left[N\frac{\partial U}{\partial w_k} - U\frac{\partial N}{\partial w_k}\right]$$

The w_k corresponding to

$$\frac{\partial V}{\partial w_k} = 0$$

and

$$\frac{\partial^2 V}{\partial {w_k}^2} < 0$$

results in the maximization of the constrained variance.

4. Extract $\frac{\partial U}{\partial w_k}$:

$$\frac{\partial U}{\partial w_k} = 2w_k\langle {x_k}^2\rangle + 2\sum_{\substack{i=0 \\ i\neq k}}^{n-1} w_i\langle x_i x_k\rangle$$

Note that this reduces to

$$\sum_{i=0}^{n-1} w_i\langle x_i x_k\rangle = 0$$

5. k-separation of N and the extraction of $\frac{\partial N}{\partial w_k}$:

$$N = {w_k}^2 + \sum_{\substack{i=0 \\ i\neq k}}^{n-1} {w_i}^2$$

and so

$$\frac{\partial N}{\partial w_k} = 2w_k$$

6. The Optimizer Simplification: Given that

$$N \neq 0$$

we simply seek to solve

$$N^2 \frac{\partial V}{\partial w_k} - N \frac{\partial U}{\partial w_k} + U \frac{\partial N}{\partial w_k} = 0$$

7. Problems with Explicit Constraint Incorporation:
   - $\frac{\partial V}{\partial w_k}$ is out of control, non-intuitive, irreducible, and devoid of formulaic meaning
   - $\frac{\partial^2 V}{\partial {w_k}^2}$ even more so
   - These terms are so derivationally challenging even in 2D PCA, as will soon be seen.

2D Principal Component Analysis – Constrained Formulation

1. Base Setup:
   - Target Vector:

$$\widehat{W} = \frac{w_x \hat{x} + w_y \hat{y}}{\sqrt{{w_x}^2 + {w_y}^2}} = \frac{w_x \hat{x} + w_y \hat{y}}{\sqrt{N}}$$

where

$$N = {w_x}^2 + {w_y}^2$$

consistent with earlier terminology.

- Point Vector Under Consideration:

$$\hat{P} = \frac{x_p \hat{x} + y_p \hat{y}}{\sqrt{{x_p}^2 + {y_p}^2}}$$

2. Point-Target Projection:

$$\hat{P} \cdot \widehat{W} = \frac{x_p w_x + y_p w_y}{\sqrt{{w_x}^2 + {w_y}^2}\sqrt{{x_p}^2 + {y_p}^2}}$$

The projection of $\hat{P}$ on $\widehat{W}$ is

$$\hat{P} \cdot \widehat{W} \times \left|\hat{P}\right| = \frac{x_p w_x + y_p w_y}{\sqrt{{w_x}^2 + {w_y}^2}}$$

3. Point Projection Squared Error and Sample Variance: The point squared error is

$$V_p = \frac{U_p}{N} = \frac{\left[x_p w_x + y_p w_y\right]^2}{N}$$

Thus

$$U = \sum_{p=0}^{q-1} U_p = {w_x}^2 \sum_{p=0}^{q-1} {x_p}^2 + {w_y}^2 \sum_{p=0}^{q-1} {y_p}^2 + 2 w_x w_y \sum_{p=0}^{q-1} x_p y_p$$

Therefore

$$V = \frac{U}{N} = \frac{{w_x}^2 \langle x^2 \rangle + {w_y}^2 \langle y^2 \rangle + 2 w_x w_y \langle xy \rangle}{N}$$

after migrating to the expectations-framework.

4. Unconstrained Variance and Constraint Derivatives:

$$\frac{\partial U}{\partial w_x} = 2w_x\langle x^2\rangle + 2w_y\langle xy\rangle$$

$$\frac{\partial^2 U}{\partial {w_x}^2} = 2\langle x^2\rangle$$

$$\frac{\partial N}{\partial w_x} = 2w_x$$

$$\frac{\partial^2 N}{\partial {w_x}^2} = 2$$

5. Constrained Variance Maximizer Set up: From before

$$\frac{\partial V}{\partial w_k} = \frac{1}{N^2}\left[N\frac{\partial U}{\partial w_k} - U\frac{\partial N}{\partial w_k}\right]$$

Thus

$$\frac{\partial V}{\partial w_k} = \frac{1}{N^2}\left[-2{w_x}^2 w_y\langle xy\rangle + 2w_x{w_y}^2(\langle x^2\rangle - \langle y^2\rangle) + 2{w_x}^3\langle xy\rangle\right] = 0$$

6. Note on PCA for uncorrelated variates: If x and y are perfectly uncorrelated, that implies that the data should be all over, with no preferential principal component. This means that the data lies with uniform density over the nD spherical shell.
   - The fact that no solution is possible in such a case is clear from the fact that the coefficients of ${w_x}^2$ in $\frac{\partial V}{\partial w_k}$ is $\langle xy\rangle$, which for mean-centered data is zero, as

$$\langle xy\rangle = \langle x\rangle\langle y\rangle = 0$$

(i.e., the solution requires both w_x and w_y to be zero, which is impossible). It is obvious from intuition that this should be true for orthogonal higher dimensions as well – we will see a proof of this soon.

7. Variance/Covariance Coefficient Representation: Setting

$$\langle x^2 \rangle = {\sigma_X}^2$$

$$\langle y^2 \rangle = {\sigma_Y}^2$$

and

$$\langle xy \rangle = \rho_{XY}\sigma_X\sigma_Y$$

we get

$$\frac{\partial V}{\partial w_k} = \frac{1}{N^2}\left[-2{w_x}^2 w_y \rho_{XY}\sigma_X\sigma_Y + 2w_x{w_y}^2({\sigma_X}^2 - {\sigma_Y}^2) + 2{w_x}^3\rho_{XY}\sigma_X\sigma_Y\right] = 0$$

Eliminating

$$w_X = 0$$

from consideration as a possible solution, and setting

$$\alpha = \frac{w_y}{w_X}$$

the above equation becomes

$$\alpha^2({\sigma_X}^2 - {\sigma_Y}^2) - \alpha\rho_{XY}\sigma_X\sigma_Y + \rho_{XY}\sigma_X\sigma_Y = 0$$

or, more succinctly

$$\alpha^2(\langle x^2\rangle - \langle y^2\rangle) - \alpha\rho_{XY}\sqrt{\langle x^2\rangle\langle y^2\rangle} + \rho_{XY}\sqrt{\langle x^2\rangle\langle y^2\rangle} = 0$$

8. Solution to α:

$$\alpha = \frac{{\sigma_Y}^2 - {\sigma_X}^2 \pm \sqrt{{\sigma_X}^4 + {\sigma_Y}^4 + 2(2{\rho_{XY}}^2 - 1){\sigma_X}^2{\sigma_Y}^2}}{2\rho_{XY}\sigma_X\sigma_Y}$$

- Solution for

$$\rho_{XY} = 1$$

the perfectly correlated case => In this case, the solution should intuitively correspond to the diagonals

$$\sigma_X = \pm\sigma_Y$$

- the corresponding variance axis, depending on which corresponds to the major/minor component. This may be seen by plugging

$$\rho_{XY} = 1$$

in the solution for α:

$$\alpha = \frac{{\sigma_Y}^2 - {\sigma_X}^2 \pm \sqrt{{\sigma_X}^4 + {\sigma_Y}^4 - 2{\sigma_X}^2{\sigma_Y}^2}}{2\rho_{XY}\sigma_X\sigma_Y} = \frac{\sigma_X}{\sigma_Y}, \frac{\sigma_Y}{\sigma_X}$$

- Solution for $\rho_{XY} = 0$, the perfectly uncorrelated case => Here

$$\alpha = \frac{0}{0}$$

or undefined. Thus, there can be no solution for the

$$\rho_{XY} = 0$$

case.

2D Principal Component Analysis – Lagrange Multiplier Based Constrained Optimization

1. Base Setup: We consider the projected variance as a constrained optimization exercise:

$$\Lambda = {w_x}^2\langle x^2\rangle + {w_y}^2\langle y^2\rangle + 2w_x w_y\langle xy\rangle - \lambda\left({w_x}^2 + {w_y}^2 - 1\right)$$

$$\Lambda = {w_x}^2(\langle x^2\rangle - \lambda) + {w_y}^2(\langle y^2\rangle - \lambda) + 2w_x w_y\langle xy\rangle + \lambda$$

2. Optimization of the above with respect to w_x and w_y:

$$\frac{\partial\Lambda}{\partial w_x} = 2w_x(\langle x^2\rangle - \lambda) + 2w_y\langle xy\rangle = 0$$

$$\frac{\partial\Lambda}{\partial w_y} = 2w_y(\langle y^2\rangle - \lambda) + 2w_x\langle xy\rangle = 0$$

To ensure consistency of the linear system above, you need the determinant across the equation set to vanish, i.e.

$$(\langle x^2\rangle - \lambda)(\langle y^2\rangle - \lambda) - \langle xy\rangle = 0$$

This should be true independent of the constraint.

3. Constraint Optimizer Formulation: Second Derivative:

$$\frac{\partial^2 \Lambda}{\partial {w_x}^2} = 2w_x(\langle x^2 \rangle - \lambda) < 0$$

ONLY if

$$\langle x^2 \rangle < \lambda$$

and

$$\frac{\partial^2 \Lambda}{\partial {w_y}^2} = 2w_y(\langle y^2 \rangle - \lambda) < 0$$

ONLY if

$$\langle y^2 \rangle < \lambda$$

Of course, if

$$\lambda > \langle x^2 \rangle$$

and

$$\lambda > \langle y^2 \rangle$$

that satisfies both of the above.

$$\frac{\partial \Lambda}{\partial \lambda} = 0$$

clearly results in

$$w_x{}^2 + w_y{}^2 = 1$$

which recovers the constraint as expected.

4. Solution to λ: From the linear system collinearity, we get

$$\lambda^2 - (\langle x^2 \rangle - \langle y^2 \rangle)\lambda - \langle xy \rangle^2 = 0$$

Thus

$$\lambda_{1,2} = \frac{\langle x^2 \rangle - \langle y^2 \rangle \pm \sqrt{(\langle x^2 \rangle - \langle y^2 \rangle)^2 + 4\langle xy \rangle^2}}{2}$$

5. Residuals Analysis: Set

$$\mathcal{R} = \frac{\sqrt{(\langle x^2 \rangle - \langle y^2 \rangle)^2 + 4\langle xy \rangle^2} - (\langle x^2 \rangle - \langle y^2 \rangle)}{2}$$

Thus

$$\mathcal{R} > 0$$

Further

$$\lambda_1 = \langle x^2 \rangle + \mathcal{R}$$

and

$$\lambda_2 = \langle y^2 \rangle - \mathcal{R}$$

Thus

$$\lambda_1 > \langle x^2 \rangle$$

and

$$\lambda_2 < \langle y^2 \rangle$$

Notice that

$$\langle y^2 \rangle - \lambda_1 = \langle y^2 \rangle - \langle x^2 \rangle - \mathcal{R}$$

which may or may not satisfy the original criterion.

$$\langle y^2 \rangle - \lambda_2 = \mathcal{R}$$

which eliminates this from the first PCA. Thus, the best candidate for the first PCA is λ_1.

6. w_x/w_y Solution: Remember

$$w_y = -\frac{\lambda - \langle x^2 \rangle}{\langle xy \rangle} w_x$$

and that

$${w_x}^2 + {w_y}^2 = 1$$

Thus

$$w_x = \frac{\pm \langle xy \rangle}{\sqrt{(\lambda - \langle x^2 \rangle)^2 + \langle xy \rangle^2}}$$

and

$$w_y = \frac{\mp(\lambda - \langle x^2 \rangle)}{\sqrt{(\lambda - \langle x^2 \rangle)^2 + \langle xy \rangle^2}}$$

7. Principal Components:

$$w_{1x} = \frac{\langle xy \rangle}{\sqrt{(\lambda_1 - \langle x^2 \rangle)^2 + \langle xy \rangle^2}}$$

$$w_{1y} = \frac{(\lambda_1 - \langle x^2 \rangle)}{\sqrt{(\lambda_1 - \langle x^2 \rangle)^2 + \langle xy \rangle^2}}$$

$$w_{2x} = \frac{\langle xy \rangle}{\sqrt{(\lambda_2 - \langle x^2 \rangle)^2 + \langle xy \rangle^2}}$$

$$w_{2y} = \frac{(\lambda_2 - \langle x^2 \rangle)}{\sqrt{(\lambda_2 - \langle x^2 \rangle)^2 + \langle xy \rangle^2}}$$

8. Orthogonality of the Principal Components:

$$\vec{W}_1 = w_{1x}\hat{x} + w_{1y}\hat{y}$$

$$\vec{W}_2 = w_{2x}\hat{x} + w_{2y}\hat{y}$$

$$\vec{W}_1 \cdot \vec{W}_2 = w_{1x}w_{2x} + w_{1y}w_{2y} = \frac{\langle xy \rangle^2 + (\lambda_1 - \langle x^2 \rangle)(\lambda_2 - \langle x^2 \rangle)}{\sqrt{(\lambda_1 - \langle x^2 \rangle)^2 + \langle xy \rangle^2}\sqrt{(\lambda_2 - \langle x^2 \rangle)^2 + \langle xy \rangle^2}}$$

$$\lambda_1 - \langle x^2 \rangle = \frac{-\langle x^2 \rangle + \langle y^2 \rangle + \sqrt{(\langle x^2 \rangle - \langle y^2 \rangle)^2 + 4\langle xy \rangle^2}}{2}$$

$$\lambda_2 - \langle x^2 \rangle = \frac{-\langle x^2 \rangle + \langle y^2 \rangle - \sqrt{(\langle x^2 \rangle - \langle y^2 \rangle)^2 + 4\langle xy \rangle^2}}{2}$$

Thus

$$(\lambda_1 - \langle x^2 \rangle)(\lambda_2 - \langle x^2 \rangle) = -\langle xy \rangle^2$$

results in

$$\vec{W}_1 \cdot \vec{W}_2 = 0$$

9. Sample Cross-Variance and Joint Expectation under the new Principal Components: As always

$$\hat{P} = \frac{x_p \hat{x} + y_p \hat{y}}{\sqrt{{x_p}^2 + {y_p}^2}}$$

The projection of $\hat{P}$ on $\vec{W}_1$ is

$$P_1 = w_{1x} x_p + w_{1y} y_p$$

and the projection of $\hat{P}$ on $\vec{W}_2$ is

$$P_2 = w_{2x} x_p + w_{2y} y_p$$

Thus

$$\begin{aligned} P_1 P_2 &= \left(w_{1x} x_p + w_{1y} y_p\right)\left(w_{2x} x_p + w_{2y} y_p\right) \\ &= w_{1x} w_{1y} {x_p}^2 + w_{2x} w_{2y} {y_p}^2 + \left(w_{1x} w_{2y} + w_{1y} w_{2x}\right) x_p y_p \end{aligned}$$

resulting in

$$\langle P_1 P_2 \rangle = w_{1x} w_{1y} \langle x^2 \rangle + w_{2x} w_{2y} \langle y^2 \rangle + \left(w_{1x} w_{2y} + w_{1y} w_{2x} \right) \langle xy \rangle$$

From

$$w_{1x} w_{2x} \langle x^2 \rangle = \frac{\langle xy \rangle^2 \langle x^2 \rangle}{\sqrt{(\lambda_1 - \langle x^2 \rangle)^2 + \langle xy \rangle^2} \sqrt{(\lambda_2 - \langle x^2 \rangle)^2 + \langle xy \rangle^2}}$$

and

$$w_{1y} w_{2y} \langle y^2 \rangle = \frac{(\lambda_1 - \langle x^2 \rangle)(\lambda_2 - \langle x^2 \rangle) \langle y^2 \rangle}{\sqrt{(\lambda_1 - \langle x^2 \rangle)^2 + \langle xy \rangle^2} \sqrt{(\lambda_2 - \langle x^2 \rangle)^2 + \langle xy \rangle^2}}$$

we get

$$w_{1x} w_{2x} \langle x^2 \rangle + w_{1y} w_{2y} \langle y^2 \rangle = \frac{\langle xy \rangle^2 (\langle x^2 \rangle - \langle y^2 \rangle)}{\sqrt{(\lambda_1 - \langle x^2 \rangle)^2 + \langle xy \rangle^2} \sqrt{(\lambda_2 - \langle x^2 \rangle)^2 + \langle xy \rangle^2}}$$

since

$$(\lambda_1 - \langle x^2 \rangle)(\lambda_2 - \langle x^2 \rangle) = -\langle xy \rangle^2$$

Using

$$\left(w_{1x} w_{2y} + w_{1y} w_{2x} \right) \langle xy \rangle = \frac{\langle xy \rangle^2 (\langle x^2 \rangle - \lambda_1) + \langle xy \rangle^2 (\langle x^2 \rangle - \lambda_2)}{\sqrt{(\lambda_1 - \langle x^2 \rangle)^2 + \langle xy \rangle^2} \sqrt{(\lambda_2 - \langle x^2 \rangle)^2 + \langle xy \rangle^2}}$$

and given that

$$\langle x^2 \rangle - \lambda_1 + \langle x^2 \rangle - \lambda_2 = \langle y^2 \rangle - \langle x^2 \rangle$$

$$\left(w_{1x}w_{2y} + w_{1y}w_{2x}\right)\langle xy \rangle = \frac{\langle xy \rangle^2 (\langle y^2 \rangle - \langle x^2 \rangle)}{\sqrt{(\lambda_1 - \langle x^2 \rangle)^2 + \langle xy \rangle^2}\sqrt{(\lambda_2 - \langle x^2 \rangle)^2 + \langle xy \rangle^2}}$$

by combining all the above, we see that

$$\langle P_1 P_2 \rangle = 0$$

showing that extracting principal components is identically the same as diagonalizing the data set covariance.

10. PCA as an Eigenization Operation: The diagonal entries of the 2D Matrix get converted to $\lambda_1 - \langle x^2 \rangle$ and $\lambda_2 - \langle x^2 \rangle$ by virtue of the constraint incorporation. Thus, eigenization corresponds to norm-preserving constraint variance maximization.
11. PCA Variance Range: If

$$\langle x^2 \rangle > \langle y^2 \rangle$$

the range of variance resides between $\langle x^2 \rangle + \varepsilon_1$ and $\langle y^2 \rangle - \varepsilon_2$. If the data set is already orthogonally represented in its native basis, then

$$\varepsilon_1 = \varepsilon_2 = 0$$

n-D Principal Component Analysis – Lagrange Multiplier Based Constrained Optimization

1. The Lagrangian Set up: The main step is to separate the U term into variance and the co-variance sub-components.
2. Lagrangian – Point Projected Variance Partitioning:

$$U_p = \left[\sum_{i=0}^{n-1} w_i x_{i,p}\right]\left[\sum_{j=0}^{n-1} w_j x_{j,p}\right] = \sum_{i=0}^{n-1}\sum_{j=0}^{n-1} w_i w_j x_{i,p} x_{j,p} = \sum_{i=0}^{n-1} {w_i}^2 {x_{i,p}}^2 + \sum_{i=0}^{n-1}\sum_{\substack{j=0 \\ j\neq i}}^{n-1} w_i w_j x_{i,p} x_{j,p}$$

3. Lagrangian – Partitioning Variance:

$$U = \sum_{i=0}^{n-1} {w_i}^2 \langle {x_i}^2 \rangle + \sum_{i=0}^{n-1}\sum_{\substack{j=0 \\ j\neq i}}^{n-1} w_i w_j \langle x_i x_j \rangle$$

4. Formulation of the Lagrangian: Setting the constraint as

$$N = \sum_{i=0}^{n-1} {w_i}^2 - 1$$

one gets

$$\Lambda = U - \lambda N = \sum_{i=0}^{n-1} {w_i}^2 \langle {x_i}^2 \rangle + \sum_{i=0}^{n-1}\sum_{\substack{j=0 \\ j\neq i}}^{n-1} w_i w_j \langle x_i x_j \rangle - \lambda\left(\sum_{i=0}^{n-1} {w_i}^2 - 1\right)$$

Thus

$$\Lambda = \sum_{i=0}^{n-1} {w_i}^2 (\langle {x_i}^2 \rangle - \lambda) + \sum_{i=0}^{n-1}\sum_{\substack{j=0 \\ j\neq i}}^{n-1} w_i w_j \langle x_i x_j \rangle$$

5. Decompose the Lagrangian into variance/covariance sections:

$$\Lambda = \upsilon + \xi$$

where

$$\upsilon = \sum_{i=0}^{n-1} {w_i}^2(\langle {x_i}^2 \rangle - \lambda)$$

and

$$\xi = \sum_{i=0}^{n-1} \sum_{\substack{j=0 \\ j \neq i}}^{n-1} w_i w_j \langle x_i x_j \rangle$$

6. k Separate υ and ξ:

$$\upsilon = {w_k}^2(\langle {x_k}^2 \rangle - \lambda) + \sum_{\substack{i=0 \\ i \neq k}}^{n-1} {w_i}^2(\langle {x_i}^2 \rangle - \lambda)$$

and

$$\xi = 2w_k \sum_{\substack{i=0 \\ i \neq k}}^{n-1} w_i \langle x_i x_k \rangle + \sum_{\substack{i=0 \\ i \neq k}}^{n-1} \sum_{\substack{j=0 \\ j \neq i,k}}^{n-1} w_i w_j \langle x_i x_j \rangle$$

7. Derivatives of Λ:

$$\frac{\partial \Lambda}{\partial w_k} = 2w_k(\langle {x_k}^2 \rangle - \lambda) + 2 \sum_{\substack{i=0 \\ i \neq k}}^{n-1} w_i \langle x_i x_j \rangle = 0$$

Also

$$\frac{\partial^2 \Lambda}{\partial {w_k}^2} = 2(\langle {x_k}^2 \rangle - \lambda)$$

Thus, all the earlier observations made with regards to 2D PCA apply here too.

8. Constraint on the Linear System from $\frac{\partial \Lambda}{\partial w_k} = 0$: Again, the above linear system should have zero determinant

$$L = 0$$

This will, in turn, produce n eigen-values and eigen-components.

$$L = \begin{bmatrix} \langle {x_n}^2 \rangle - \lambda & \cdots & \langle x_0 x_{n-1} \rangle \\ \vdots & \ddots & \vdots \\ \langle x_{n-1} x_0 \rangle & \cdots & \langle {x_{n-1}}^2 \rangle - \lambda \end{bmatrix}$$

9. Diagonalized L: If L is already diagonal, then are 3 implications.
   - λ_i corresponds to the i[th] variance.
   - $\max \lambda_i$ is the maximum variance, i.e., the principal component.
   - All the components are orthogonal, i.e., the order of $abs\ (\lambda_i)$ determines the order of the principal component.
10. PCA Determination Steps:
    - Diagonalize the basis to create the new ones in which the data set is orthogonal i.e.

$$\langle x_i x_j \rangle = 0$$

    - Each diagonal entry corresponds to the eigen-value, and to the corresponding principal component.
11. PCA Variance Range: The 2D variance range analysis is broadly true for this case as well. Consider the partitioned variance

$$U = \sum_{i=0}^{n-1} {w_i}^2 \langle {x_i}^2 \rangle + \sum_{i=0}^{n-1} \sum_{\substack{j=0 \\ j \neq i}}^{n-1} w_i w_j \langle x_i x_j \rangle$$

Given that (without loss of generality)

$$w_i w_j \langle x_i x_j \rangle \geq 0$$

is always true - because of the following: If

$$\langle x_i x_j \rangle < 0$$

then

$$w_i w_j < 0$$

and if

$$\langle x_i x_j \rangle > 0$$

then

$$w_i w_j > 0$$

Thus, the diagonalized basis corresponds to the maximal/minimal variance range, with maximum corresponding to the most significant PC, and the minimum corresponding to the least significant PC.

Information Theoretic Analysis of PCA

1. Information Preservation Property of PCA: If

$$\vec{X} = \vec{S} + \vec{n}$$

where $\vec{X}$ is the data set, $\vec{S}$ is the signal, and $\vec{n}$ is the noise, then it can be shown that if both $\vec{S}$ and $\vec{n}$ are Gaussian, and $\vec{n}$ is an orthogonal matrix (i.e., the noise components are orthogonal to each other), then PCA maximizes the mutual information $I(\vec{y} \mid \vec{S})$ between $\vec{S}$ and $\vec{y}$, where

$$\vec{y} = {W_L}^T \vec{X}$$

the L PCA components.

- If the noise is still Gaussian and orthogonal i.i.d., but $\vec{S}$ is not, PCA ends up minimizing the net information loss $I(\vec{X} \mid \vec{S}) - I(\vec{y} \mid \vec{S})$.
- If the noise is orthogonal and i.i.d., and is more Gaussian than $\vec{S}$ in the Kullback-Leibler sense, then PCA is still optimal.
- If the noise is co-dependent, then PCA is not optimal anymore (in terms of information loss, information being measured by Shannon entropy) irrespective of where the signal is Gaussian or not.

Empirical PCA Estimation from the Data Set

1. Calculate the central measures: Calculate the sample mean, the deviations from the mean, and the covariance matrix C.
2. Eigenize the covariance Matrix: Find the eigen-values and eigen-vectors of C from

$$V^{-1}CV = D$$

where V is the matrix of eigen-vectors, and D is a diagonal matrix.

3. Eigen-vector Ordering: Re-arrange the eigen-vectors and eigen-values in the order of decreasing eigen-values.
4. Cumulative Eigen-Energy: Compute the cumulative energy content for each eigen-vector from

$$g[m] = \sum_{q=1}^{m} D(q,q)$$

where m is the total number of eigen-vectors, and the summation runs over the ordered eigen-vector set.

- Select the sub-set of eigen-vectors as the PCA proxy until the desired energy level has been achieved, e.g.

$$\frac{g(L)}{g(M)} \geq 0.9$$

5. Conversion to Z-Scores:
   - Convert the source data to Z-scores ($\vec{Z}$) by scaling it element-by-element by the variance.
   - Project the Z-scores of the data onto the new basis as

$$Y = W^* \cdot \vec{Z}$$

where W^* is the matrix of the eigen-vectors.

Kriging

1. Definition: Kriging is an estimation technique that infers the value of a random field at an unobserved location from the nearby samples. Simply put, kriging is a stochastic spline interpolation technique – the nodal samples themselves are only stochastically observed, i.e.

$$x_0 = \alpha x_1 + (1 - \alpha)x_2$$

implies

$$\langle x_0 \rangle = \alpha \langle x_1 \rangle + (1 - \alpha)\langle x_2 \rangle$$

and

$$\langle {x_0}^2 \rangle = \alpha \langle {x_1}^2 \rangle + (1 - \alpha)\langle {x_2}^2 \rangle$$

2. Moment Stationarity:
   - Stationarity in the first moment => This means that

$$\langle x_i \rangle = \langle x_j \rangle$$

for all

$$i \neq j$$

The second moment may vary, though.
   - Stationarity in the second moment => Here, the correlation between x_1 and x_2 is posited to depend solely on

$$h = |x_1 - x_2|$$

(or, more generally, $\vec{h}$).

- Variance

$$\gamma(x_1, x_2) = \gamma(x_1 - x_2) = \gamma(h)$$

- Co-variance

$$\varsigma(x_1, x_2) = \varsigma(x_1 - x_2) = \varsigma(h)$$

- From these, the variance and the covariance may also be computed – based on N samples:

$$\gamma(h) = \frac{1}{2N(\vec{h})} \sum_{p=1}^{N(\vec{h})} \left[x_{i,p} - x_{i+h,p}\right]^2$$

$$\varsigma(h) = \frac{1}{2N(\vec{h})} \sum_{p=1}^{N(\vec{h})} \left[x_{i,p} x_{i+h,p} - m(x_i) m(x_i + \vec{h})\right]$$

$m(x_i)$ is the sample mean , i.e

$$m(x_i) = \frac{1}{2N(\vec{h})} \sum_{p=1}^{N(\vec{h})} x_{i,p}$$

3. Linear Estimation for a Location:

$$\hat{x}_0 = [w_0 \quad \cdots \quad w_{n-1}] = \begin{bmatrix} x_0 \\ \vdots \\ x_{n-1} \end{bmatrix} = W^T X = \sum_{i=0}^{n-1} w_0(x_0) * x_1$$

The objectives when calculating the weights are:

- Unbias: this implies that the estimation mean is the same as the sample mean.
- Minimal variance of estimation: Somehow $\hat{x}_0 - x_0$ should be minimal.

4. Ordinary Kriging: Error is $\hat{x}_0 - x_0$. Since the random field is assumed to be stationary, i.e.

$$\langle x_i \rangle = \langle x_j \rangle$$

$$\sum_{i=0}^{n-1} w_i(x_0) = 1$$

$$[W^T \quad -1] \begin{bmatrix} \langle {x_i}^2 \rangle & \langle x_0 x_i \rangle \\ \langle x_0 x_i \rangle & \langle {x_i}^2 \rangle \end{bmatrix} \begin{bmatrix} W^T \\ -1 \end{bmatrix} = \langle {x_0}^2 \rangle + \sum_{i,j} w_i w_j \langle x_i x_j \rangle - 2 \sum_i w_i \langle x_i x_0 \rangle$$

This shows the dependence of the error variance on the individual γ and ς operators. A generalized Kriging System is created by the minimization of

$$Variance[\varepsilon(x_0)] = [W^T \quad -1] \begin{bmatrix} \langle {x_i}^2 \rangle & \langle x_0 x_i \rangle \\ \langle x_0 x_i \rangle & \langle {x_i}^2 \rangle \end{bmatrix} \begin{bmatrix} W^T \\ -1 \end{bmatrix}$$

subject to the constraint

$$\sum_{i=0}^{n-1} w_i(x_0) = 1$$

which is handled by using the Lagrange multipliers λ.

5. Simple Kriging: Both simple and ordinary kriging assume stationarity on the first moment, but in simple kriging, it is further assumed that the first moment (i.e., the mean) is known.

6. Universal Kriging: Universal kriging assumes a polynomial trend model for the location estimator, i.e.

$$\hat{x}_0 = \sum_{i=0}^{n-1} \beta_i f_i(\vec{x})$$

7. IRFk Kriging: Same as universal kriging, but assumes an unknown polynomial in $\vec{x}$.
8. Disjunctive kriging: Disjunctive kriging is a non-linear generalization of kriging. Log-normal kriging is an example.
9. Kriging in Computer Simulation Outputs: Kriging is used in the interpolation of the data coming out as response variables from deterministic computer simulations. Typical steps involved are:
    - Generate several location specific parameters and responses.
    - Use kriging to interpolate between them.

Hidden Markov Models

HMM State Transition/Emission Parameter Estimation

1. State Estimation Definition: State estimation is the full parameterization of the functional relationship between one/more elastic constitutive predictor(s) and one/more elastic response variables. Typically, the constitutive predictor(s) are not stochastic; the response variables may/may not be stochastic.
    - Representation vs. Transformation => Representation makes sense when dealing with state estimation, as it captures an inherent relationship (i.e., a relationship that already exists) between a hidden state quantification metric and the predictor. Transformation happens when a quantification metric manifests itself as an observation manifest metric. Representation and transformation may share similar function forms, but the conceptual notions underlying them are distinct.
    - Incorporating Dynamics vs. History-based Evolution => In HMMs, dynamics still refers to modeling dynamic equilibrium – otherwise there no real "stasis" oriented state to calibrate/uncover/characterize.
2. Undirected Graphical Relationship of the HMM States: Given the nature of the transition probability matrix (potentially non-zero, i.e., dense), HMM essentially models essentially an undirected graphical relationship. However, directionality may be imposed using non-zero unidirectional state transition matrix. Of course, a single HMM state inference run may use the full suite of the observation set – while the still maintaining the state updates Markovian (for e.g., this property makes HMM filtering/smoothing use the entire observation set).
3. Categorical Observable HMM Emission Parameters Analysis: If the observed categorical variable takes any of the M discrete values, and if there are N HMM parameters, then the number of emission parameters is $N(M-1)$ (M-1 instead of M since the emission probabilities from each state sum up to 1).

4. Continuous Real-Valued Observable HMM Emission Parameters Analysis: If there are M continuous states for each of the N hidden states' emission set, you then have one mean and $\frac{M(M+1)}{2}$ covariance entries across all the states' emission sets. Thus, the total is

$$N\left[M + \frac{M(M+1)}{2}\right] = N\frac{M(M+3)}{2} \cong \mathcal{O}(NM^2)$$

   - One way to reduce the number of parameters in the previous case is to assume that the emission states are independent, in which case the total number of emission parameters comes to NM. The State Transition matrix will still be a $N \times N$ matrix with

$$N^2 - N = N(N-1)$$

     entries (the $-N$ term is due to the sum of the probabilities being 1).
5. Continuous State Space/Continuous Observation HMM: Kalman filtering is a simple example where the state estimation from observation is a simple linear inference, and therefore tractable. If either the state evolution or the observation transform is non-linear, then the UKF/EKF/Particle Filter techniques are to be used.
6. Auto-regressive Nature of Markovian States: Markov states are, by definition, auto-regressive. Thus, the auto-regression covariance matrix corresponds to the state transition matrix.
7. A Note on Inferring the Past vs. Predicting the Future: You may infer the past quantification metric, as well as predict the future quantification metric/manifest measure. Therefore, in that sense, inference/prediction is relative only to the current time (and using earlier/later information).
8. i.i.d. vs. HMM: HMM's are not quiet i.i.d.'s, since the state jumps depend on the current state. In a history transition sense, the order of history dependence goes as i.i.d -> HMM -> k-level HMM -> Volterra series …

HMM Based Inference - Applications

1. Formulation Treatment: Details in Stratonovich (1960a, 1960b), Baum and Petrie (1966), Baum and Eagon (1967), Baum and Sell (1968), Baum, Petrie, Soules, and Weiss (1970), Baum (1972).
2. Inferring the Probability of an Observed Sequence: Given the model parameters, the probability of observing the sequence

$$\vec{Y} = \{y(0), \cdots, y(L-1)\}$$

   is given as

$$P(\vec{Y}) = \sum_{\vec{X}} P(\vec{Y}|\vec{X})P(\vec{X})$$

   where $\vec{X}$ captures all the possible hidden sequences. Dynamic Programming and Forward Algorithm solves this.
3. Filtering: Given the model parameters and a sequence of observations, the task is to compute the distribution $P\left(\vec{X}|y(0), \cdots, y(t)\right)$ over the hidden states of the last latent variable at the end of the sequence. Again, the Forward Algorithm solves this.
4. Smoothing: This is the same as filtering, with the difference being that the task is now to compute the distribution $P\left(\vec{X}(k) \mid y(0), \cdots, y(t)\right)$ where

$$k < t$$

   Again Forward-Backward Algorithm solves this.
5. Most Likely State Sequence: This addresses the joint probability of the entire sequence of hidden states that generated the specified sequence of observations. This is solved effectively using the Viterbi algorithm, and requires finding a maximum over all possible state sequences.

6. Statistical Significance: Re-casting the MLE statement above as: What is the probability that a sequence drawn from some NULL distribution will have a HMM emission probability at least as large as that of a particular sequence (using forward algorithm, see Newberg (2009))? What is the probability that a sequence drawn from some NULL distribution will have a maximum state sequence probability at least as large as that of a particular sequence (using Viterbi algorithm)?
7. Parameter Learning: Given an output sequence (or a set of sequences), what is the optimal set of state transition and emission/output probabilities (using MLE)?
    - Both Baum-Welch and Baldi-Chauvin algorithms are special case locally optimal MLE's. Exact solution to parameter learning is not available through any known tractable algorithm.

Non-Bayesian HMM Model Setup

1. Stage #1: The prior state distribution is assumed to uniform over all possible states (i.e., the starting state is not specified). Further, in typical HMM cases, the categorical states are computed from the real valued states.
2. Stage #2:
    - N => The Number of States (the number of discrete state realizations possible).
    - T => The Number of Observations – one measurement per each time step.
    - $\theta_{i=1,\cdots,N}$ => Emission Parameter for all observations associated with state i. Each $\theta_{i=1,\cdots,N}$ could be an array in itself, if the observation space is discrete (which is the case for the typical HMM).
    - $\phi_{i=1,\cdots,N;j=1,\cdots,N}$ => Probability of transition from state i from state j.
    - $\Phi_{i=1,\cdots,N}$ => N dimensional vector for each of $\phi_{i,\forall j}$. Sums to Unity.
    - $x_{t=1,\cdots,T}$ => State at each of the time instants t.
    - $y_{t=1,\cdots,T}$ => Observation at each of the time instants t.
    - $F(y \mid \theta)$ => Probability distribution of an Observation, parameterized on θ (Gaussian or Categorical).

- $x_{t=2,\cdots,T}$ => Categorical value computed from $\phi(x_{t-1})$.
- $y_{t=1,\cdots,T}$ => Observation computed from $F[\theta(x_t)]$

3. Setup with Real-Valued/Gaussian Observations:
   - N => The Number of States (the number of discrete state realizations possible).
   - T => The Number of Observations – one measurement per each time step.
   - $\phi_{i=1,\cdots,N;j=1,\cdots,N}$ => Probability of transition from state i from state j.
   - $\Phi_{i=1,\cdots,N}$ => N dimensional vector for each of $\phi_{i,\forall j}$. Sums to Unity.
   - $\mu_{i=1,\cdots,N}$ => Mean of Observations associated with state i.
   - ${\sigma^2}_{i=1,\cdots,N}$ => Variance of Observations associated with state i.
   - $x_{t=1,\cdots,T}$ => State at each of the time instants t.
   - $x_{t=2,\cdots,T}$ => Categorical value computed from $\phi(x_{t-1})$.
   - $y_{t=1,\cdots,T}$ => Observation at each of the time instants t.
   - $y_{t=1,\cdots,T}$ => $\mathcal{N}\left(\mu(x_t), \sigma^2(x_t)\right)$

   Notice that there is no θ here, since the observations are real-valued.

4. Setup with Categorical Observations:
   - N => The Number of States (the number of discrete state realizations possible).
   - T => The Number of Observations – one measurement per each time step.
   - $\phi_{i=1,\cdots,N;j=1,\cdots,N}$ => Probability of transition from state i from state j.
   - $\Phi_{i=1,\cdots,N}$ => N dimensional vector for each of $\phi_{i,\forall j}$. Sums to Unity.
   - V => Dimension of the categorical variables – e.g., size of the word vocabulary, number of financial regimes etc:
   - $\theta_{i=1,\cdots,N;j=1,\cdots,V}$ => Probability of state i observing observation item j.
   - $\theta_{i=1,\cdots,N}$ => V dimension vector, composed of $\theta_{j=1,\cdots,V}$ - must sum to Unity.
   - $x_{t=1,\cdots,T}$ => State at each of the time instants t.
   - $x_{t=2,\cdots,T}$ => Categorical value computed from $\phi(x_{t-1})$.
   - $y_{t=1,\cdots,T}$ => Observation at each of the time instants t.
   - $y_{t=1,\cdots,T}$ => Observation computed from $\theta(x_t)$

Bayesian Extension to the HMM Model Setup

1. The Framework: All are essentially same as those for the non-Bayesian setup, except for the new formulations below, based on the specified hyper-parameters:
   - α => Shared Hyper-parameters for the Emission Parameters.
   - β => Shared Hyper-parameters for the Transition Parameters.
   - $H(\theta \mid \alpha)$ => Prior Probability Distribution of Emission Parameters parameterized on α.
   - $\theta_{i=1,\cdots,N} \rightarrow H(\alpha)$ => H is the conjugate of F.
   - $\Phi_{i=1,\cdots,N}$ => Made from $Symmetric\_Dirichlet(\beta)$.
2. Setup with Real-Valued/Gaussian Observations:
   - N => The Number of States (the number of discrete state realizations possible).
   - T => The Number of Observations – one measurement per each time step.
   - $\phi_{i=1,\cdots,N;j=1,\cdots,N}$ => Probability of transition from state i from state j.
   - $\Phi_{i=1,\cdots,N}$ => N dimensional vector for each of $\phi_{i,\forall j}$. Sums to Unity.
   - $\mu_{i=1,\cdots,N}$ => Mean of Observations associated with state i.
   - ${\sigma^2}_{i=1,\cdots,N}$ => Variance of Observations associated with state i.
   - $x_{t=1,\cdots,T}$ => State at each of the time instants t.
   - $y_{t=1,\cdots,T}$ => Observation at each of the time instants t.
   - β => Concentration Hyper-parameter controlling the Density of the Transition Matrix.
   - μ_0, λ => Shared Hyper-parameters of the Means for each State.
   - ${\sigma_0}^2, \nu$ => Shared Hyper-parameters of the Means for each State.
   - $\Phi_{i=1,\cdots,N}$ => $Symmetric\_Dirichlet(\beta)$.
   - $x_{t=2,\cdots,T}$ => Categorical value computed from $\phi(x_{t-1})$.
   - $\mu_{i=1,\cdots,N}$ => $\mathcal{N}(\mu_0, \lambda{\sigma_i}^2)$
   - ${\sigma^2}_{i=1,\cdots,N}$ => $Inverse\_Gamma(\nu, {\sigma_0}^2)$
   - $y_{t=1,\cdots,T}$ => $\mathcal{N}\left(\mu(x_t), \sigma^2(x_t)\right)$
3. β as a Concentration Parameter: β controls the density of the transition matrix. With high β

$$\beta \gg 1$$

the probabilities of transitioning to the other states are higher, making the HMM process more random. Low β

$$\beta \ll 1$$

causes greater state localization, and thinner outbound transition, thereby making the HMM less random.

4. Priors for Categorical Distribution: Dirichlet distribution is a natural choice, as it automatically serves as a conjugate pair for any categorical distribution. Symmetric Dirichlet used across multiple categorical variates is analogous to uniform distributions used in real-valued priors.
5. Setup with Categorical Observations:
   - N => The Number of States (the number of discrete state realizations possible).
   - T => The Number of Observations – one measurement per each time step.
   - $\phi_{i=1,\cdots,N;j=1,\cdots,N}$ => Probability of transition from state i from state j.
   - $\Phi_{i=1,\cdots,N}$ => N dimensional vector for each of $\phi_{i,\forall j}$. Sums to Unity.
   - V => Dimension of the categorical variables – e.g., size of the word vocabulary, number of financial regimes etc:
   - $\theta_{i=1,\cdots,N;j=1,\cdots,V}$ => Probability of state i observing observation item j.
   - $\theta_{i=1,\cdots,N}$ => V dimension vector, composed of $\theta_{j=1,\cdots,V}$ - must sum to Unity.
   - $x_{t=1,\cdots,T}$ => State at each of the time instants t.
   - $y_{t=1,\cdots,T}$ => Observation at each of the time instants t.
   - α => Shared Concentration Hyper-parameter θ for each State.
   - β => Concentration Hyper-parameter controlling the Density of the Transition Matrix.
   - $\phi_{i=1,\cdots,N}$ => $Symmetric\_Dirichlet(\beta)$.
   - $\Theta_{i=1,\cdots,N}$ => $Symmetric\_Dirichlet(\alpha)$.
   - $x_{t=2,\cdots,T}$ => Categorical value computed from $\phi(x_{t-1})$.
   - $y_{t=1,\cdots,T}$ => Observation computed from $\theta(x_t)$

6. Two-Level Bayesian HMM: The purposes are to a) independently control the overall density of the transition matrix, and b) independently control the target densities of states to which the transitions are likely (i.e., the density of the prior distribution of states in any particular hidden variable – this serves as the initial estimate for the next stage). In both cases, this is accomplished by maintaining ignorance over which specific states are more likely to be transitioned into starting from a given state.
    - The following set of 2-level parameter models with non-uniform priors can be learned with Gibbs' sampling or enhanced Expectation Maximization algorithms.
    - Formulation => Here, we enhance the fields β and $\Phi_{i=1,\cdots,N}$ with:
        - γ => Concentration Hyper-parameter controlling how many states are intrinsically linked.
        - β => Concentration Hyper-parameter controlling the Density of the Transition Matrix.
        - η => N Dimensional Vector Probabilities, specifying the intrinsic probability of a given State.
        - $\mathcal{N}$ => $Symmetric\_Dirichlet_N(\gamma)$.
        - $\Phi_{i=1,\cdots,N}$ => $Dirichlet_N(\beta, \mathcal{N}, \gamma)$.
    - Given that the Dirichlet process is the conjugate of unknown infinite state of categorical variables, multi-level Dirichlet models are also called hierarchical Dirichlet HMM (HDP-HMM) or Infinite Markov Models.
    - Two-level priors with both concentration parameters set to produce sparse distributions are used in unsupervised part-of-speech tagging, for e.g., where some parts occur more frequently than others.

HMM in Practical World

1. Role for HMM based Algorithmization: Full-scale HMM appear to deployed much more after the process is a) either matured, or b) is in the final stages of maturity (e.g., speech processing) where the characteristic regime transitions/state switches are all well tested and studied – in other words, it lends itself well to post-supervised processes.

2. Applications Developed as a Practice:
    - Biological Sequences (Bishop and Thompson (1986)) and Bio-informatics (Durbin, Eddy, Krogh, and Mitchison (1999)).
    - Gesture Recognition (Starner and Pentland (1995)).
    - Metamorphic Virus Detection (Wong and Stamp (2006)).
    - Musical Score Following (Pardo and Birmingham (2005)).
    - Partial Discharges (Satish and Gururaj (2003)).
    - Protein Folding (Stigler, Ziegler, Gieseke, Gebhardt, and Rief (2011)).
    - Speech Recognition (Baker (1975), Jelinek, Bahl, and Mercer (1975), Huang, Jack, and Ariki (1990), Huang, Acero, and Hon (2001)).
    - Speech Tagging (Rabiner (1989)).

References

- Baker, J. (1975): The DRAGON System – An Overview, *IEEE Transactions on Acoustics, Speech, and Signal Processing* **23** 24-29.
- Baum, L. E., and T. Petrie (1966): Statistical Inference for Probabilistic Functions of Finite State Markov Chains *Annals of Mathematical Statistics* **37 (6)** 1554-1563.
- Baum, L. E., and J. A. Eagon (1967): An Inequality with Applications to Statistical Estimation for Probabilistic Functions of Markov Processes and to a Model for Ecology *Bulletin of the American Mathematical Society* **73 (3)** 360.
- Baum, L. E., and G. R. Sell (1968): Growth Transformations for Functions on Manifolds *Pacific Journal of Mathematics* **27 (2)** 211-227.
- Baum, L. E., T. Petrie, G. Soules, and N. Weiss (1970): A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains *The Annals of Mathematical Statistics* **41** 164.
- Baum, L. E. (1972): An Inequality and Associated Maximization Technique in Statistical Estimation of Probabilistic Functions of a Markov Process *Inequalities* **3** 1-8.

- Bishop, M. and E. Thompson (2001): Random Projection in Dimensionality Reduction: Applications to Image and Text Data *Proceedings of the 26th Annual International Conference on Knowledge Discovery and Data Mining, ACM 2001*.
- Durbin, R., S. R. Eddy, A. Krogh, and G. Mitchison (1999): *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids* **Cambridge University Press**.
- Huang, X., M. Jack, and Y. Ariki (1990): *Hidden Markov Models for Speech Recognition*, **Edinburgh University Press**.
- Huang, X., A. Acero, and H. W. Hon (2001): *Spoken Language Processing* **Prentice Hall**.
- Jelinek, F., L. Bahl, and R. Mercer (1975): Design of a Linguistic Statistical Decoder for the Recognition of Continuous Speech, *IEEE Transactions on Information Theory* **21 (3)** 250.
- Newberg, L. (2009): Error Statistics of Hidden Markov Model and Hidden Boltzmann Model Results *BMC Bioinformatics* **10** 212.
- Pardo, B. and W. Birmingham (2005): Modeling Form for Online following of Musical Performances **AIAA-05 Proceedings**.
- Rabiner, L. R. (1989): A Tutorial on Hidden Markov Models and selected Applications in Speech Recognition, *Proceedings of the IEEE* **77 (2)** 257-286.
- Satish, L. and B. I. Gururaj (2003): Use of Hidden Markov Models for Partial Discharge Pattern Classification *IEEE Transactions on Dielectrics and Electrical Insulation*.
- Starner, T. and A. Pentland (1995): Real Time American Sign Language Visual Recognition from Video using Hidden Markov Models *Master's Thesis* **MIT**.
- Stigler, J., F. Ziegler, A. Gieseke, J. C. M. Gebhardt, and M. Rief (2011): The Complex Folding Network of Single Calmodulin Molecules, *Science* **334 (6055)** 512-516.
- Stratonovich, R. L. (1960a): Application of the Markov Processes to Optimal Filtering *Radio Engineering and Electronic Physics* **5 (11)** 1-19.
- Stratonovich, R. L. (1960b): Conditional Markov Processes *Theory of Probability and its Applications* **5** 156-178.
- Wong, W. and M. Stamp (2006): Hunting for Metamorphic Engines, *Journal in Computer Virology* **2 (3)** 211-229.

Markov Chain Models

Markov Property

1. Definition: The state of the system at time t_i only depends on the state at t_{i-1} (of course, in addition it may also depend on additional external shocks during this instant).
2. Semi-Markovian Smoothener: In this case, the state at t_i actually depends on the states at $\{t_j\}_{j=i-N}^{i}$, i.e., smoothening needs to occur across the last N observations.
    - In lagging Markovian system, the state at t_i depends only on the lagging state t_{i-N}.

Markov Chains

1. Definition: Markov definition indicates strict immediate prior dependence. The sequence of Markov realizations constitutes the Markov chain.
2. Example of Markov Chain: Language parsing using tagging is a great example (Sequence Labeling (Wiki)). Here the whole context ends up being parsimoniously restricted to the neighborhood tags per each parsed fragment. The tags can, of course, become a sequence, and therefore a sizeable list/chain – a Markov chain.
3. Bayesian Estimation/Updates as Markov Chain Update: The single posterior Bayesian estimate depends only on the prior. Thus, the Markov property is maintained. Further, the sequence of Bayesian posteriors automatically constitutes a Markov chain.
4. Entities in the Markov Chain: The entities that constitute the Markov chain are all inferred/predicted, i.e., they form a sequence of entities that sequentially form the basis for the next stage of prediction.
5. Markov Random Field: These are also called Markov network. It is just a multi-dimensional Markov chain. The inference techniques used result in multi-dimensional joint distributions.

Classification of the Markov Models

1. Markov Model Classification Table: This classification comes from Wikipedia (Markov Model (Wiki)). Markov chains as described earlier correspond to models for fully observable and autonomous systems.

System Observability and Type	Fully Observable	Partially Observable
Autonomous	Markov Chain	Hidden Markov Model
Controlled	Markov Decision Process	Partially Observable Markov Decision Process

2. Partially Observable, Autonomous System: These systems are modeled using Hidden Markov Models, which employ state inference/estimation techniques from a sequence of observations. The following algorithms fall into this category:
   - The Viterbi algorithm will infer the most likely corresponding sequence of states.
   - The forward algorithm will infer the probability of a given sequence of observations.
   - The Baum-Welch will infer the starting probabilities, the transition functions, and the observation functions.
3. Controlled Markov System: Control is always specified through a controlled action vector, which implies that the optimal control vector parameters need to be extracted. In general, this extraction occurs by using non-linear iterative fixed-point finder approaches.
4. Fully Observable, Controlled System: In these systems, the state transitions are observed from the current state to which an action vector is applied. These systems are modeled using the Markov Decision Process, which employ a policy of actions that maximize a chosen utility function with respect to some targeted rewards.

- These are closely related to reinforcement learning and Kalman filtering/extraction formulation, and the action policy parameters are computed using iteration and other related methods.

5. Partially Observable, Controlled System: These systems are also modeled using the Markov Decision Process. These are known to be NP-complete, but recent approximations complete (Kaelbling, Littman, and Cassandra (1998)) have improved their tractability to make them useful for a variety of applications such as controlling robots, etc.

Monte Carlo Markov Chains (MCMC)

1. Definition: MCMC methods are a class of algorithms for sampling a probability distribution based on constructing a Markov chain that has the desired distribution that, after a large number of steps, the generated distribution becomes the desired distribution (Markov Chain Monte-Carlo (Wiki)).
2. MCMC Entity Distribution vs. Target Distribution: Given that the steps correspond to a Markov chain in the theoretical sense, the chain of steps can be constructed to form the TRUE target distribution.
    - MCMC Entity Generation Rule => As long as the proposed rule is guaranteed to be able to sample the target distribution (either as a consequence of direct algorithms, or using sufficient statistics proxies), the MCMC entity generation walks will correspond to the equilibrium distribution.
3. MCMC Mixing Time: This is the time needed to converge to the stationary distribution within acceptable error (it is measured in the number of steps, or more accurately, in cops or cost-of-operations). Lesser number of steps indicates that the corresponding algorithm has a rapid mixing time.

MCMC for Multi-dimensional Integrals

1. Base Algorithm: Here, an ensemble of MCMC invocations (called walkers) moves around the variate space semi-randomly, looking for a place with a "high-enough" contribution (Robert and Casella (2004), Gill (2008)).
2. Conventional Random-Walker Integral vs. MCMC Integral: Random samples in a conventional random-walker integrand are generated statistically independently, whereas those generated in MCMC random walks are auto-correlated (i.e., due to the Markov nature, the current walker position determines where it is walk to next). The only constraint is that the generated walk-coordinates maintain the target distribution, which, as we've seen, is easy to do.
3. Random Walker Sampling Algorithms:
    - Metropolis-Hastings Algorithm => This algorithm generates a random walk using a proposal density, and a method for rejecting the proposed moves.
        - The multiple-try Metropolis algorithm is a variant on the above that allows multiple trials at each variate point. It allows the algorithm to take larger steps in each direction, and is particularly useful when dealing with large dimensions.
    - Gibbs Sampling => In this algorithm, all the conditional samples of the target distribution are generated precisely, thereby without need for algorithmic tuning.
    - Slice Sampling => This algorithm samples the distribution by sampling uniformly the region under its density plots. In practice, it alternates between the "uniform vertical" and the "uniform horizontal" slice defined by the current vertical position.
4. MCMC Semi Random-Walker Integration Algorithms: MCMC Semi Random-Walker Integration Algorithms prevent the walker from doubling back. These are harder to implement, but result in faster convergence.
5. MCMC Semi Random-Walker Sampling Algorithms:
    - Successive Over-relaxation => The Monte-Carlo version of successive over-relaxation sometimes avoids the random walks, thereby improving upon the Gibbs sampling.
    - Hybrid Monte-Carlo => Hybrid Monte-Carlo algorithms avoid random walks by introducing an auxiliary momentum vector and Hamilton dynamics (where the potential energy is a function of the target density).
        - Hybrid Monte-Carlo vs. Simulated Annealing => Both of these are techniques borrowed from statistical physics and applied in optimal algorithmic walk/search.

- Targeted Slice Sampling => This uses a modification in the slice sampling technique to avoid random walks.
- Self-targeting Candidates => Langevin MCMC and other methods (Stramer and Tweedie (1999)) that rely on gradient/Hessian (and other higher order Jacobian descents) of the log posterior avoid random walks by making proposals that are likely to be in the directions of higher probability.
- Changing Dimensions => Reversible-jump algorithm is a variant on the Metropolis-Hastings algorithm that allows proposals that alter the dimensionality of the variate space. This comes of particular use when performing Gibbs sampling over non-parametric Bayesian models, where the number of mixing components/clusters has to be automatically inferred from the data.
    - Sliced reversible jumps => Once the dimensions are altered, reversing along a variate may be needed. Using this technique in conjunction with sliced/sampling and other methods may quicken the convergence.

References

- Gill, J. (2008): *Bayesian Methods: A Social and Behavioral Sciences Approach (2nd edition)* **Chapman and Hall/CRC** London.
- Kaelbling, L. P., M. Littman, and A. Cassandra (1998): Planning and Acting in Partially Observable Stochastic Domains, *Artificial Intelligence* **101** 99-134.
- Markov Chain Monte-Carlo (Wiki): Wikipedia Entry for Markov Chain Monte-Carlo.
- Markov Model (Wiki): Wikipedia Entry for Markov Model.
- Robert, C. P. and G. Casella (2004): *Monte-Carlo Statistical Methods (2nd edition)* **Springer** New York.
- Stramer, O., and R. Tweedie (1999): Langevin-Type Models II: Self-targeting Candidates for MCMC Algorithms, *Methodology and Computing in Applied Probability* **1 (3)** 307-328.

Markov Random and Condition Random Fields

Introduction and Background

1. Multi-dimensional Graph/Lattice View of MRF/CRF: In MRF, the state response variables are laid out in an n-D graph/lattice. In addition, in the case of CRF, the realizations at the response lattices are conditional on the corresponding observations.
2. Random Fields as an Enhancement to the Old Regression Relations: In traditional regression frameworks, you have $y_i = f(x_i) + \varepsilon$. In Conditional Random Fields extension to Markov Random Fields formulation, these change to $y_i = f(y_{i-1}, x_i) + \varepsilon$ where y_{i-1} in the feature function is used to accommodate the Markov nature of the regression.
    - Discriminant models as State-Space based multivariate regression approaches => That is another way of looking at its, since after all they generate conditional distributions (both y_i and y_{i-1} can be vectors).
    - Non-trivial observation/feature functions => Further, the enhanced regression may be formulated as $y_i = f_{y_{i-1}}(x_i) + \varepsilon$, indicating that the feature function can in itself be dependent on the (Makovian) history.
3. Markov Random Field Motivation: This is undirected discriminant approach, and thus can represent cyclic relationships unlike directed graphs. However, it cannot represent induced/directed dependencies.
4. Conditional Random Field Motivation: CRF is a variant of MRF where the random variable at each graph node is conditioned upon the set of global observations $\{O_i\}$. The feature function for MRF maps $\{O_i\}$ to the given feature clique (defined below).
5. n-Way Property of MRF: Undirected lattice background (e.g., Ising models, see Kindermann and Snell (1980)) are MRF's prototypical settings. Thus, it is commonly used in image processing (image restoration, image retrieval, image registration, image segmentation, and image completion), and computer vision (texture synthesis, resolution, matching, and retrieval) (He, Zemel, and Carreira-Perpinnan (2004), Rue and Held (2005), Li (2009)).

- Labeling is a very common application scenario, particularly for CRF. Here, the predictor is a word/speech, and the response could the corresponding word/speech type.
- Other customized uses for CRF include Shallow Parsing (Sha and Pereira (2003)) and Named Entity Recognition (Settles (2004)).

6. Markovian on State only: Just like HMM, states represented by the random fields are Markovian. Thus, a single state inference run may use the full suite of the observation set – while the still maintaining the state updates Markovian.
    - Difference with HMM => In typical HMM setup, the observation/feature functions and/or the state transition matrix are independent of the observation set. In MRF/CRF, however, this is not the case (as seen before).

MRF/CRF Axiomatic Definition/Properties

1. MRF Definition: Given an undirected graph

$$G = (V, E)$$

the set of random variables

$$X = (X_v)_{v \in V}$$

indexed by v form a MRF with respect to graph G if they satisfy the following three Local Markov properties (Markov Random Field (Wiki)):

- Pair-wise Markov Property => Any two non-adjacent variables are conditionally independent given all the other variables, i.e.

$$X_u \perp X_v \mid X_{v \setminus \{u,v\}}$$

if

$$\{u, v\} \notin E$$

($\perp$ is the symbol for independence).

- Local Markov Property => A variable is conditionally independent of all other variables given its neighbors, i.e.

$$X_u \perp X_{ClosedNeigh(v)} \mid X_{Neigh(v)}$$

where $Neigh(v)$ is the set of neighbors of v and $ClosedNeigh(v)$ is the closed neighborhood of v.

- Global Markov Property => Any two subsets of variables are conditionally independent given a separating subset, i.e.

$$X_A \perp X_B \mid X_S$$

where every path from any node in set A to a node in set B passes through a node in set S.

2. Markov Property Strength: The above three Markov properties are NOT equivalent to each other at all (i.e., none of them telescope any other). In other of strength precedence, the Local Markov Property is stronger than the pair-wise one, but is weaker than the Global Property.
3. Gaussian MRF: A multivariate normal distribution forms an MRF with respect to the graph

$$G = (V, E)$$

if the missing edges correspond to zeros on the precision matrix (the inverse covariance matrix)

$$X = (X_v)_{v \in V} \cong \mathcal{N}(\mu, \Sigma)$$

such that

$$(\Sigma^{-1})_{uv} = 0$$

if

$$\{u, v\} \notin E$$

4. CRF Definition: For observations $\vec{O}$ and the Random State Variables $\vec{X}$, CRF is defined as (Lafferty, McCallum, and Pereira (2001)): Consider an undirected graph

$$G = (V, E)$$

such that

$$\vec{X} = \left(\vec{X}_v\right)_{v \in V}$$

such that $\vec{X}$ is indexed by the vertices of G. Then $\left(\vec{X}, \vec{O}\right)$ is a conditional random field when the random variables X_v, conditioned on $\vec{O}$, obey the following Markov property with respect to the graph:

$$P\left(X_v \mid \vec{O}, X_w : w \neq v\right) = P\left(X_v \mid \vec{O}, X_w : w \sim v\right)$$

where

$$w \sim v$$

indicates that w and v are neighbors (Conditional Random Field (Wiki)).

- In other words, CRF is an undirected graphical discriminant model whose nodes can be divided into exactly 2 disjoint sets - $\vec{X}$ and $\vec{O}$ (these two graphs are unconnected). Then conditional distribution $P\left(\vec{X} \mid \vec{O}\right)$ is then modeled.

- Another view – CRF is essentially an extension of logistic regression applied to sequential data – this form makes is particularly amenable for use in NLP.

5. Higher Order CRF: Higher-order CRF relaxes the single look-back MRF requirement, thereby using the fixed observation set with look-back K. K is typically kept ≤ 5 to reduce computational costs. Large margin models such as SVM are alternatives to CRF.
   - Semi-Markov CRF employs variable length look-back segmentations (Sarawagi and Cohen (2005))), thereby retaining the power of higher-order CRF to model deep-range dependencies at a reasonable computational cost.
   - Linear-chain CRF can be used effectively in conjunction with parallelization (Lavergne, Cappe, and Yvon (2010)).

Clique Factorization

1. Definition: In certain cases (Moussouris (1974) identifies an exception), it may be possible to express the joint distribution probability into a distribution over the "feature base", i.e., the joint probability density may be expressed as

$$\prod_i P(x_i) = \prod_C \phi_C(x_C)$$

   where ϕ_C is the probability of a particular configuration in the feature space. ϕ_C is called the factor potential or clique potential.
2. Logistic Formulation using the Clique Factorized Representation:
   - Nomenclature:
     - k => Element index into the clique configuration space (i.e., the clique cardinality).
     - N_k => The number of State Entities in Configuration k.
     - $w_{i,k}$ => Weight of the State Entity in Configuration k, defined as

$$w_{i,k} = \log \phi\left(C_{i,k}\right)$$

where $C_{i,k}$ is the i[th] configuration in clique k.

- $f_{i,k}$ => The Feature Function Indicator in the clique configuration, defined as

$$f_{i,k}(x_{[k]}) = 1$$

if state i is part of the clique configuration C_k, and zero otherwise.

- $\mathfrak{N}$ => The universe of the relevant states across all the cliques under consideration.

- The normalized joint distribution now is

$$P(X = x) = \frac{1}{Z} e^{\Sigma_k {w_k}^T f_k(x_{[k]})}$$

where

$${w_k}^T f_k(x_{[k]}) = \sum_{i=1}^{N_k} {w_{i,k}}^T f_{i,k}(x_{[k]})$$

and

$$Z = \sum_{x \in \mathfrak{N}} e^{\Sigma_k {w_k}^T f_k(x_{[k]})}$$

- The probability expression above is also called Gibbs' measure, with the only restriction being that, in the partition expression for $\mathfrak{N}$, there can be no ZERO contributions.

3. Value behind the Partition-Function based Logistic Representation: These formulations derive direct intuitions from statistical mechanics, and therefore ease the computations of expectations for various metrics. For instance, by adding a driving force term J_v for each vertex v in the graph, we may differentiate it to get the expectation as follows:

$$Z[\vec{J}] = \sum_{x \in \Re} e^{\sum_k w_k{}^T f_k(x_{[k]}) + \sum_v J_v X_v}$$

results in

$$\langle X_v \rangle = \frac{1}{Z[\vec{J}]} \left| \frac{\partial Z[\vec{J}]}{\partial J_v} \right|_{J_v = 0}$$

provides the expectation of X_v. Further

$$\langle X_u, X_v \rangle = \frac{1}{Z[\vec{J}]} \left| \frac{\partial^2 Z[\vec{J}]}{\partial J_u \partial J_v} \right|_{J_u, J_v = 0}$$

provides the expectation of the pair-wise correlation function.

Inference in MRF/CRF

1. Alternate Terminology in Use: Most of this involves the observation set, so applicable more to CRF.
   - Model Training => Learn the Conditional distribution $(\vec{X}, \vec{O})$ between $\vec{X}$ and $\vec{O}$ using the feature/observation functions from the input corpus of data.
   - "Inference" => Determine the probability of a given label sequence $\vec{X}$ from $\vec{O}$; this implicitly requires learning/calibration to have been already done.
   - Decoding => Determine the most likely label sequence $\vec{X}$ from $\vec{O}$ (honestly, this is one of the prediction operations).
2. Exact Inference in MRF: If we use exact inference as is done in a Bayesian network, it is easy to calculate the conditional distribution of a set of nodes

$$V' = \{v_1, \cdots, v_i, \cdots, v_n\}$$

given another set

$$W' = \{w_1, \cdots, w_i, \cdots, w_n\}$$

by summing over all

$$u \in V', W'$$

However, this is NP-complete, and is computationally intractable.

- Here V' and W' can be current and the Markov prior states, respectively. Given that this is an MRF framework, observations do not come in here explicitly.
- CRF Inference/Learning => Using MLE, if all state nodes have exponential family distributions and they are all observable/observed, the optimization is convex; therefore, gradient descent, Quasi-Newton etc: maybe used (Sutton and McCallum (2010)).
- CRF Unobserved Inference => If some of the state variables are unobservable, exact inference becomes intractable, and the approximate methods outlined earlier become useful.

3. Approximate Inference: For pure MRF, MCMC, loopy belief propagation, MRF subclasses (using trees – for e.g., Chow-Liu tree) have more feasible polynomial time inference.
    - In case of CRF, approximate inference techniques also include mean field inference, linear programming relaxations, etc.

References

- Conditional Random Field (Wiki): Wikipedia Entry for Conditional Random Field.
- He, X., R. S. Zemel, and M. A. Carreira-Perpinnan (2004): Multi-scale Condition Random Fields for Image Labeling *IEEE Computer Society*.
- Kindermann, R., and J. L. Snell (1980): *Markov Random Fields and Their Applications* **American Mathematical Society**.

- Lafferty, J., A. McCallum, and F. Pereira (2001): Conditional Random Fields: Probabilistic Models for segmenting and labeling Sequence Data *Proc. 18th International Conf. On Machine Learning* **Morgan Kaufmann** 282-289.
- Lavergne, T., O. Cappe, and F. Yvon (2010): Practical Very Large Scale CRFs *Proceedings of the 48th Annual Meeting of the ACL* 504-513.
- Li, S. Z. (2009): *Markov Random Field Modeling in Image Analysis* **Springer**.Linear Discriminant Analysis (Wiki): Wikipedia Entry for Linear Discriminant Analysis.
- Markov Random Field (Wiki): Wikipedia Entry for Markov Random Field.
- Moussouris, J. (1974): Gibbs' and Markov Random Systems with Constraints *Journal of Statistical Physics* **10 (1)** 11-33.
- Rue, H., and L. Held (2005): *Gaussian Markov Random Fields: Theory and Applications* **CRC Press**.
- Sarawagi, S. and W. W. Cohen (2005): Semi-Markov Conditional Random Fields for Information Extraction, in *Advances in Neural Information Processing Systems 17 (L. K. Saul, Y. Weiss, and L. Boutto (eds.))* **MIT Press: Cambridge, MA** 1185-1192.
- Settles, B. (2004): Bio-medical Named Entity Recognition using Conditional Random Fields and Rich Feature Sets *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications* 104-107.
- Sha, F. and F. Pereira (2003): Shallow Parsing with Conditional Random Fields.
- Sutton, C. and A. McCallum (2010): Introduction to Conditional Random Fields.

Maximum Entropy Markov Models (MEMM)

1. Definition: MEMM's are also referred to as Conditional Markov Model. They are discriminant graphical models for sequence labeling that combine the features of HMM and maximum entropy models (Maximum Entropy Markov Model (Wiki)).
2. Sequence Generation given Observations: The probability of generating a state sequence S given the observation set O is given as

$$P(S_1, \cdots, S_n \mid O_1, \cdots, O_n) = \prod_{t=1}^{n} P(S_t \mid S_{t-1}, O_t)$$

   This arises out of the Markov property, namely that the probability of transitioning to a particular label depends ONLY on a) the observation at that location, and b) the earlier label.
3. Clique based Reduction (Berger, Pietra, and Pietra (1996)):

$$P(S_t \mid S_{t-1}, O_t) = \frac{1}{Z(S_{t-1}, O_t)} e^{\sum_a \lambda_a f_a(S_t, O_t)}$$

   where $Z(S_{t-1}, O_t)$ is the normalizer, and $f_a(S_t, O_t)$ is the real-valued or categorical-valued feature function.
   - Inference of λ_a => Given that λ_a is extracted out of a calibration process, the generalized iterative scaling technique (Darroch and Ratcliff (1972)) is a popular one if the observations are available. Variant of the Baum-Welch algorithm (McCallum, Freitag, and Pereira (2000)) has been proposed if the training data has missing or incomplete, and has been applied for speech tagging (Toutanova and Manning (2000)) and information extraction (McCallum, Freitag, and Pereira (2000)).
4. Optimal Label Sequence Extraction: Given the observation set O, a variant of the Viterbi algorithm may be used to compute the forward probability

$$\sum_{s' \in S} \alpha_t(s') P(s' \mid S, O_{t+1})$$

5. Drawbacks of MEMM: Two key shortcomings: a) Label bias associated with completely ignoring low-entropy state transitions (Lafferty, McCallum, and Pereira (2001)), and b) Since the training/calibration occur only with regards to the last known label, MEMM does not work well if there is label uncertainty (Bottou (1991)).
6. Advantage over HMM: Given that it is an undirected discriminant model, MEMM freedom in the choice of features and their functions (same as CRF). HMM relies upon observation independence, while MEMM does not.
    - Advantage over CRF => Training in MEMM is significantly more efficient, since there is a straightforward one-to-one mapping between the transition probability and the MEMM probability distribution. In HMM/CRF, typically a variant of the forward/backward algorithm needs to be used for the training, which can be expensive.

References

- Berger, A. L., V. J. D. Pietra, and S. A. D. Pietra (1996): Maximum Entropy Approach in Natural Language Processing *Computational Linguistics* **22 (1)** 39-71.
- Darroch, J. N. and D. Ratcliff (1972): Generalized Iterative Scaling for Log-Linear Models *Annals Mathematical Studies* **43 (5)** 1470-1480.
- Maximum Entropy Markov Model (Wiki): Wikipedia Entry for Maximum Entropy Markov Model.
- McCallum, A., D. Freitag, and F. Pereira (2000): Maximum Entropy Markov Models for Information Extraction and Segmentation *Proc. ICML 2000* 591-598.
- Toutanova, K., and C. D. Manning (2000): Enriching the Knowledge Sources used in a Maximum-Entropy Part-of-Speech Trigger *Proc. J. DIGDAT Conf. On Empirical Methods in NLP and Very Large Corpora (EMNLP/VLC-2000)* 63-70.

Probabilistic Grammar and Parsing

Parsing

1. Definition: Parsing is also called syntactic analysis, and is the process of analyzing a string of symbols according to the rules of a formal grammar (Parsing (Wiki)).
2. Traditional Sentence Parsing: This emphasizes the importance of divisions in sentence construction (such as subject/predicate etc.), and is done with the help of sentence diagrams.
    - Traditional Method-Clause Analysis is essentially a grammatical exercise to break down the speech form and function and to identify the syntactic relation between the parts. This is, of course, very intricate for highly inflected languages (i.e., languages that employ frequent conjugations/declensions).
    - Problems with human languages => Structural ambiguity, i.e., "man bites dog" in language #1 is the same as "dog bites man" in language 2, i.e., the interpretation needs to rely on a bigger context established by the language grammar rules, which are often hard to clearly elicit.
    - Challenges with a single grammatical framework => Even if one grammar is chosen, the grammar parsing system must be established, with the typical choices being lexical functional grammar parser (which is NP-complete!), head-driven grammar parsing (which can get very complex), shallow parsing (which only identifies the boundaries between the major sentence constituents), and any of these augmented by the dependency grammar parsing, etc.
3. Computational Linguistic Parsing: This is done using a formal computer analysis on a word string and its constituents, resulting in a parse tree that displays the syntactic relation, semantic content, and other information between those words.
    - Modern statistical parsing => Modern statistical parsers are trained on a corpus of data, using a context-based frequency of occurrence metric, with enhancements to incorporate

probabilistic context free grammars, lexical statistics, and smoothing to avoid over-fitting.

4. Psycholinguistic Parsing: This refers to the way humans analyze a sentence/word in terms of its grammatical constituents, identifying the parts of speech and their syntactic relations etc: These techniques are used by psychologists to describe language comprehension and to provide cues to help speakers interpret/fix garden-path (i.e., wrongly structured) sentences.
    - Psycholinguistic parsing evaluates the meaning of the sentence according to the rules drawn by the inferences made from each word in the sentence.
    - Psycholinguistic parsing is also incremental in that the interpretation/structure is constructed right through the processing, and expressed in terms of the partial syntactic structure, with care to avoid garden-pathing.
5. Programming Language Parsing: This refers to the syntactic analysis of the input computer code into its component parts to facilitate the tasks of compilers/interpreters.
    - Steps:
        - Lexical analyzer breaks the input into tokens
        - Syntactic analyzer (true parser) verifies the validity of the sequence to check if they constitute a valid expression (done using the grammar rules)
        - Semantic analyzer converts the sequence into the appropriate operational/execution units (e.g., generates machine code, or the interpretation execution).
    - The typical output of syntactic parsing is the parse tree (either as an abstract syntax tree or as an equivalent hierarchical structure (Tree Structure (Wiki))).

Parser

1. Context-free Parsing Grammar: Context-free grammars are limited in the extent to which they can be expressive of all the language requirements; however, they are simpler than context-oriented grammars which need to choose in/switch out of the context (based upon an additionally established criterion).

2. Formal Definition: The task of a parser is to determine how the input can be derived from the start symbol of the grammar. It is conducted in essentially 2 ways – top-down parsing, and bottom-up parsing.
3. Terminology and Nomenclature:
    - LL => Left-to-right token processing, left-most derivative
    - LR => Left-to-right token processing, right-most derivative
    - LALR => Look-ahead LR parser
    - Production => Production is the result of conversion of a single token into its corresponding part-of-speech.
    - Derivation => A set of productions together constitute a derivation or a parse. Typically, notions of syntactic validity are enforced at this stage, as the "set" may correspond to a syntactic isolation unit (e.g. a "sentence" in a natural language, or a "statement" in programming language).
4. Top-down Parsing: Top-down parsing attempts to find the left-most derivations of an input stream by searching for the parse trees using a top-down expansion of the given formal rules. Tokens are consumed left to right (Aho, Sethi, and Ullman (1986)).
5. Bottom-up Parsing: Here, the parser typically attempts to locate the most basic elements, followed by the elements containing these, etc. This is also referred to a shift-reduce parsing. LR parsers are examples of bottom-up parsers.
    - Enhanced top-down parsing => A new family of sophisticated algorithms for top-down LL parsing of ambiguous, context-free grammars (Frost, Hafiz, and Callaghan (2007), Frost, Hafiz, and Callaghan (2008)) accommodate ambiguity and left-recursion in polynomial time, and polynomial-size representations of potentially exponential number of parse trees (again in polynomial time). These algorithms can produce both left-most and right-most derivations of a given input.
6. Look-ahead Parsing: Look-ahead parsing establishes the maximum number of incoming tokens that a parser can use to decide which rule it can invoke. It is represented as LALR (number of tokens).

Context-Free Grammar (CFG)

1. Probabilistic/Stochastic Context-Free Grammar (PCFG/SCFG) Definition: In this context-free grammar, each production is augmented with a production/translation probability, and thus the probability of the entire derivation becomes a product of the individual productions used in the derivation. Alternately, this probability is indicative of the probability that the derivative is consistent with the given grammar.
2. Estimation of the Probability of most likely Derivation: A variant of the CYK algorithm finds the Viterbi parse of the sequence for a given SCFG – the Viterbi parse that is most likely derivative/parse of the sequence given the SCFG.
3. Probability of all the Generatable Sets associated with the given Sequence: Use the inside-outside algorithm, a variant of the forward-backward algorithm. This may be used in conjunction with expectation-maximization algorithms to learn the maximum likelihood probabilities of a SCFG based off of the given set of training sequences (Aycinena (2005)).
4. CFG as a Natural Language Model: Natural Language grammars, conceived as sets of production rules (syntaxed, say, with the Backus-Naur form), are absolute (Chomsky (1957)), but inadequate (except in programming language type situations).
    - PCFG/SCFG as a way to handle Natural Language Complexities => CFG's are one way to ensure that more than one production rule may apply to a given sequence of words. CFG-based approach is better than approaches that use fixed-set production rules for the following reasons:
        - Having a deep-set production rule proliferate (to add more fixed rules as necessary) makes it difficult to manage (despite using rule precedence hierarchy).
        - Results in over-generation (i.e., generation of valid yet highly unlikely structures).
        - Prevents parsing without context, thereby cannot apply CFG's.
    - How CFG handles this => CFG's address the issues above by assigning a probability to the given derivation, and working out a winner-takes-all (i.e., the most likely) interpretation.
5. Learning Grammar for SCFG: Learning happens from large corpus of annotated texts. Typical sources include the Penn Tree-Bank that trained the Stanford Statistical Parser (Klein

and Manning (2003)), Speech Recognition Tuner (SCFG parsing implementation is available in Buetler, Kaufman, and Pfister (2005)).

6. RNA Sequence Mapping: Nucleotide secondary structure base-pairing may be represented with the grammar

$$S \rightarrow aSu \mid cSg \mid gSc \mid uSa$$

(this admits only wholly complementary regions of canonical pairs $a - u$ and $c - g$ (Durbin, Eddy, Krogh, and Mitchison (1999), Eddy and Durbin (1994))). Thus, given a genome sequence, SCFG may use this grammar to find the RNA genes (e.g., see Stochastic Context Free Grammar (Wiki)).

7. SCFG in Psycholinguistics: SCFG based models (Horning (1969) and Clark (2001)) have helped revise earlier nativist views that language is hard-wired in humans at birth (Gold (1967) and Chomsky (1980)).
8. Probabilistic Grammars in Cognitive Plausibility: Probabilistic versions of minimalist grammars (Hale (2006)) have helped evaluate psycholinguistic models and assess product difficulties associated with different syntactic structures (e.g., accessibility hierarchy for relative clauses).

References

- Aho, A. V., R. Sethi, and J. D. Ullman (1986): *Compilers: Principles, Techniques, and Tools* **Addison-Wesley Longman** Boston.
- Aycinena, M. A. (2005): *Probabilistic Geometric Grammars for Object Recognition* PhD Thesis **Massachusetts Institute of Technology**.
- Buetler, R., T. Kaufman, and B. Pfister (2005): Integrating a non-probabilistic Grammar into large Vocabulary Continuous Speech Recognition *IEEE Workshop on Automated Speech Recognition and Understanding* 104-109.
- Chomsky, N. (1957): *Syntactic Structures* **Mouton and Co. Publishers** Den Haag Netherlands.

- Chomsky, N. (1980): *Rules and Representations* **Oxford: Basil Blackwell.**
- Clark, A. (2001): *Unsupervised Language Acquisition: Theory and Practice* PhD Thesis **University of Sussex**.
- Durbin, R., S. R. Eddy, A. Krogh, and G. Mitchison (1999): *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids* **Cambridge University Press**.
- Eddy, S. R., and R. Durbin (1994): RNA Sequence Analysis using Covariance Models, *Nuclei Acids Res.* **22 (11)** 2079-2088.
- Frost, R., R. Hafiz, and P. Callaghan (2007): Modular and Efficient Top-down Parsing for Ambiguous Left-Recursive Grammars, *10th International Workshop on Parsing Technologies (IWPT), ACL-SIGPARSE* 109-120.
- Frost, R., R. Hafiz, and P. Callaghan (2008): Parsers Combinators for Ambiguous Left-Recursive Grammars *10th International Conference on Practical Aspects of Declarative Languages (PADL), ACL-SIGPLAN* **4092** 167-181.
- Hale, J. (2006): Uncertainty about the rest of the Sentence *Cognitive Science* **30 (4)** 643-672.
- Horning, J. J. (1969): *A Study of Grammatical Inference* PhD Thesis **Stanford University**.
- Klein, D., and C. Manning (2003): Accurate Unlexicalized Parsing *Proceedings of the 41st Meeting of the Association for Computational Linguistics* 423-430.
- Parsing (Wiki): Wikipedia Entry for Parsing.
- Stochastic Context Free Grammar (Wiki): Wikipedia Entry for Stochastic Context Free Grammar.
- Tree Structure (Wiki): Wikipedia Entry for Tree Structure.

Bayesian Analysis: Concepts, Formulation, Usage, and Application

Framework Symbology

1. General Framework Formulation:

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}$$

2. $P(A \mid B)$: This refers to a population sample restricted exclusively to those consisting of B – in this population what is the chance of A occurring? $P(B \mid A)$ refers to the opposite.
3. Relation between $P(A \mid B)$ and $P(B \mid A)$:

$$P(A \mid B) \propto P(B \mid A)$$

and

$$P(A \mid B) \propto P(A)$$

$$P(A \mid B)P(B) \propto P(B \mid A)P(A) = P(A \cap B)$$

4. Hierarchical Bayesian Network: Represented as $P(\theta, \varphi \mid x) = P(x \mid \theta)P(\theta \mid \varphi)P(\varphi)$ etc: There can be multiple such φ where each φ stands for a specific parameter of parameter.

Applicability

1. Evidentiary Belief Analysis: Purpose of Bayes' analysis both stochastic prior/posterior processes, as well as evidentiary belief analysis – not just purely stochastic prior/posteriors.
2. Usage quiet general: Simply stems from the realization that most modern systems rely on the concept "past determines future".
3. Historical Usage of Bayesians: Celestial mechanics observations used to be parameterized using Bayesian approaches, but that was discarded in favor of using deterministic parameters with experimental uncertainties.
4. Generalized Decision Rules: MLE, MAP etc. are referred to as decision rules – fundamentally use inference techniques for parameter calibration.
5. Likelihood Estimates: Remember that likelihood

$$\Lambda \equiv \Lambda(\vec{X} \mid \vec{\alpha})$$

where $\vec{X}$ refers to the set of observations, and $\vec{\alpha}$ refers to the set of models and/or parameters. Thus likelihood estimates are, in a sense, the analogue of calibration – a methodology for assigning the probabilities of transforming observations from models/parameters.

- Likelihood estimates estimate the likelihood of parameters' assuming a certain value given the observation set, not the other way.
- What constitutes a "parameter" is simple. Every numerical model input that goes into specifying/characterizing a distribution is a parameter (e.g., μ, σ^2 for normal distribution, and p/q for binomial distribution, etc.)
- Likelihood dependence on models => Likelihood formulation is the only place where physics of all the inherent dynamics sows up. This is the reason why all the model validation activities use likelihood ratios, Bayes' factors etc.

6. Likelihood Estimation vs. Eigenization: The Bayesian formulation

$$Posterior = Likelihood \times Prior$$

is analogous to the Eigenization

$$Y = \lambda X$$

In fact, if the posterior and the prior are in the same family (conjugates), then the analogy with eigenization is even closer.

- In eigenization, transformation of the specified vector set occurs using a mixing operator that is a linear combination of the inputs; in Bayesian transformation, the mixing occurs through the conditioning operator using a convex combination of the hyper-parameter distributions.

7. Product Probability Distribution Convolutions: This is what posteriors are composed of – convolutions of likelihood and prior – to produce convolved posteriors. This is applicable whenever multi-distribution products are convolved to produce a compound convolution. If the input distributions are conjugate, it makes the compounding convolution all the easier.
   - Distributions that arise as a result of common factor conditional independence make themselves very amenable to such convolutions. If they are conjugates, that aids their closed-form tractability as well.
   - Multi-feature classifier convolutions => Previous observations maybe applied for conditionally independent Bayesian classifier feature convolutions, i.e.

$$Posterior = P(C \mid F_1)P(C \mid F_2) \cdots P(C \mid F_i) \cdots P(C \mid F_n)$$

8. Posterior vs. Likelihood Simplified:
   - Posterior $P(\theta \mid X)$ => Probability that the parameters are θ given the evidence (or observations) X.
   - Likelihood $P(X \mid \theta)$ => Probability that the observation is X given that the parameter is θ
9. Bayesian Decomposition of Technical Signals: In general, the signal core drivers are limited (like systemic/idiosyncratic factors), but the product specific manifestations are more varied. Bayesian frameworks are well suited for these.
10. Bayesian Learning: Bayesian learning occurs because of "enhancing" the posterior – enhancement happens via time evolution – through the set of data sampled over time.
11. Bayesian Techniques in NLP: Application of Bayesian techniques (particularly as practiced in Spam filtering etc:) to natural language processing challenges appears to be a good area.

Use NLP in conjunction with Bayesian techniques to work out probabilistic knowledge models.

12. Frequentist Predictions and Bayesian Prediction: Bayesian techniques are slanted more towards inference, whereas frequentist techniques are slanted towards calibration/prediction. Where predictions are employed in a Bayesian setting, they are used primarily as an out-of-sample framework quality control check, and prediction in itself is not the goal.

Analysis of Bayesian Systems

1. Conditional Probability vs. Parameter Uncertainty: As a concept, conditional probability is more general applied than targeted handling of parameter uncertainty (which, in itself, is a variant of the belief process/system). However, both use "trimming of the input universe" coming out of the observation sub-set.
   - Example of conditional probability application => Bayesian spam filtering techniques, naïve Bayesian classification techniques.
   - Examples of handling parameter uncertainty => Parameter prior distributions, Approximate Bayesian Computation, Empirical Bayes' Method, Maximum a-posteriori estimate etc.
2. Usage of Maximum Expectation: If $\tilde{V}_{Obs}$ is the observed measure, then the calibration of the parameter P_a for model M_a is done from the solution to

$$\tilde{V}_{Obs} = \max \Phi_a(M_a, V, P_a)$$

where Φ_a is the distribution corresponding to M_a. Likewise, the calibration of the parameter P_b for model M_b is done from the solution to

$$\tilde{V}_{Obs} = \max \Phi_b(M_b, V, P_b)$$

Now, whichever probability is bigger represents the better model proxy.

3. Sufficient Statistics in Bayes' Classifier: Differently banded sufficient statistics may be used to indicate correspondence with different classes – the correspondence represented by a stochastic matrix is stochastic, i.e.

$$POSTERIOR(\varsigma_k) = \sum_j TM(j,k) \cdot PRIOR(\varsigma_j)$$

Here

- ς_k represents the posterior parameters for the sufficient statistics set corresponding to the parameter combination k
- ς_j represents the prior parameters for the sufficient statistics set corresponding to the parameter combination j
- $TM(j,k)$ represents the Bayesian Translation Matrix from the Prior to the Posterior. Such intra-family closed transition forms for $TM(j,k)$ may be possible only for conjugates.

Advantage of Bayesian over other systems

1. Parameter Uncertainty: Bayesian likelihood estimation and Bayes' factor calculation occur over the full parameter space distribution, thereby automatically accounting for parameter uncertainties.
2. Classical Maximum Likelihood Estimate: Occurs simply for a single parameter, so there is no parameter distribution and/or uncertainty estimate.
3. Frequentist approach: Even more horrible, as it is simply a confidence interval estimate based on sigma cutoffs.

Bayesian Networks

1. DAG: Bayesian DAG representations should be format able as sequentially dependent stochastic variate trees, each with their independent marginal distributions – joints maybe described using a copula.
2. SKU DAG Search Quality: When using Bayesian network to determine the SKU's DAG ancestry/causality, always be careful of the "weak" graph links. Use something like Bayes' K factor decibans to filter out weak DAGs.

Hypothesis Testing

1. Types of Model Validation: It is important to distinguish between validation across different types of models, versus parameter validation ranges within a single model. Of course, Bayesian techniques can be used for both.
2. Over-fitting avoidance in Bayesian Hypothesis Testing: Automatically happens through the incorporation of the model/parameter uncertainties.
3. NULL Hypothesis: Refers to the single CLASSICAL hypothesis invoked as the sole explainer of observed phenomenon.
4. Penalizing Loss Functions: Bayesian analysis accommodates this too.
5. Hypothesis Testing:
    - Type 1 Error => This is also called FALSE POSITIVE. NULL Hypothesis is TRUE, but the experimental result rejects the NULL hypothesis.
    - Type 2 Error => This is also called FALSE NEGATIVE. NULL Hypothesis is FALSE, but the experimental result accepts the NULL hypothesis.
6. Specificity: Specificity is the fraction that truly negates the NULL Hypothesis.
    - TN => Number of TRUE Negatives
    - TP => Number of TRUE Positives
    - FN => Number of FALSE Negatives
    - FP => Number of FALSE Positives

$$Specificity = \frac{TN}{TN + FP}$$

7. Sensitivity: Sensitivity is the fraction that is truly positive of all those tested.

$$Sensitivity = \frac{TP}{TP + FN}$$

8. Positive/Negative Predictive Value:
   - Positive Predictive Value (PPV) => If the test result is positive, how well does that predict an actual presence?

$$PPV = \frac{TP}{TP + FP}$$

   - Negative Predictive Value (NPV) => If the test result is negative, how well does that predict an actual absence?

$$NPV = \frac{TN}{TN + FN}$$

Bayesian Updating

1. Successive Conjugate Bayesian Updates: Successively updating a conjugate prior using conjugate likelihood's results in progressively narrowing of the variance for the posterior. This is so despite the mean in itself getting shifted.
2. Successive Convolutions: Above observations need not apply for conjugate convolutions alone – they apply for any convolution. Specifically, if the starting prior is $(\mu_0, {\sigma_0}^2)$ and you apply successive likelihood's

$$\mathcal{N}(\mu_i, {\sigma_i}^2) \; \forall \; i = 1, \cdots, n$$

   the final variance is given by

$$\frac{1}{{\sigma_{FINAL}}^2} = \sum_{i=0}^{n} \frac{1}{{\sigma_i}^2}$$

and the final mean by

$$\mu_{FINAL} = \frac{\sum_{i=0}^{n} \frac{\mu_i}{{\sigma_i}^2}}{\sum_{i=0}^{n} \frac{1}{{\sigma_i}^2}}$$

Maximum Entropy Techniques

1. Discrete Entropy Maximization Distribution: The maximum entropy prior on a discrete space, given that the probability is normalized to unity, is the prior distribution that assigns equal probability to each state.
2. Entropy Maximization Continuous Distribution: The maximum entropy prior given that the density is normalized with a given mean and a variance, is a normal distribution!
3. MAXENT to Update Priors: Maximization of the posterior information (or MINIMIZATION of the posterior entropy) given a prior is equivalent to minimizing the information of posterior relative to a given prior.

Priors

1. Uninformed Priors: These are TRICKY. Make sure ALL possible uninformed prior states are factored in before you do this. Input group invariants (e.g., rotational, transformation, or other group transformation invariants) can be used to determine uninformed priors.

- Prior proposition Order Unimportant => All that matters when constructing the prior distribution is how many of the given parametric propositions that are testable and successful exist, not their order.

2. Bayesian Principle of Indifference: If multiple mutually exclusive and collectively exhaustive variables $x_1, x_2, \cdots, x_n$ span the distribution variate space, then the incremental cumulative distribution is the same across any one, i.e.

$$\frac{\Phi_1(x_1)}{N_1}dx_1 = \frac{\Phi_2(x_2)}{N_2}dx_2 = \cdots = \frac{\Phi_i(x_i)}{N_i}dx_i = \cdots = \frac{\Phi_n(x_n)}{N_n}dx_n$$

   Here $\Phi_i(x_i)$ is the distribution of the i[th] variate, and N_i is the corresponding normalizer.

3. MLE-based Prior Estimator: As a starting prior, it may be worth using frequentist MLE to kick start the initial prior estimate – to some extent this is the same as "uninformed prior" above.
4. Prior in Finance: Both prior as well hyperprior (hyperhyperprior etc.) are useful in finance. For example, the hazard rate can be parametrically by other parameters, which will be the hyper-parameters.
   - Financial Stochastic Volatility => Given that "volatility" is really a parameter, stochastic volatility is just a "prior" on the volatility, i.e., all the Bayesian techniques for inferring distributions of the volatility should be used.

Predictive Posteriors and Priors

1. Posteriors are always Bayesian: Posterior – by definition – is the distribution in the parameter space given the observation set. Therefore, they are applicable only to Bayesian belief systems, not frequentist systems.
2. Predictive Posterior: Say the posterior is $p_{POST}\left(\theta \mid \vec{X}, \alpha\right)$. The predictive posterior is defined as the predictive distribution for the next value of the observation x, given the earlier $\vec{X}$ and α, i.e.

$$p_{POST}(x \mid \vec{X}, \alpha) = \int p(x \mid \theta) p_{POST}(\theta \mid \vec{X}, \alpha) d\theta$$

It is simply an expectation of $p(x \mid \theta)$ over the posterior distribution $p(\theta \mid \vec{X}, \alpha)$

3. Predictive Prior:

$$p_{PRIOR}(x \mid \alpha) = \int p(x \mid \theta) p(\theta \mid \alpha) d\theta$$

Almost identical to predictive posterior, except for the absence of $\vec{X}$ - owing to the fact that there have been no observations thus far.

Approximate Bayesian Computation

1. Philosophy: Full mathematical formulation of the mathematical basis of the model (and therefore the explicit likelihood) is hard – much easier simply to just formulate the stage-by-stage model algorithm.
    - Some of the reasons why estimation of likelihood is hard: multivariate inputs resulting in input variable curse of dimensionality, poorly formulated output to latent input variate map, etc.
2. Frequentist ABC (AFC): This is equally as valid as Bayesian – in this case, it becomes good old-fashioned Monte-Carlo. You can still algorithmize the Monte-Carlo evolution/computation, and run MLE-type parameter estimation and model evaluation/validation.
3. Applications of ABC/AFC:
    - Prior Parameter Estimation
    - Model Choice/Evaluation
    - Inferring starting point/pre-divergence times computation
    - All the above in one go
4. Components of ABC:

- Non-zero sufficient statistics Convergence ε
- Insufficient/inaccurate summary statistics
- Challenges with Model Selection before ABC
- Too much sensitivity to priors and the parameter ranges
- Curse of Dimensionality
- Model Ranking off of summary statistics => Bayes' factor of summary statistics goes out of synch with Bayes' factor of the Model.

5. Non-zero Tolerance ε: Non-zero tolerance to the outcome measure results in lack of precision as well as bias.
    - Solution => Theoretical/practical studies of the sensitivity of the posterior distribution to the tolerance. "Noisy ABC" methods are solutions.
6. Inaccurate/Insufficient Summary Statistics: Sufficient statistics effectively reduces the dimensionality of the observations by trimming the full UNIVERSE of observations to a set of parsimonious parameters. Insufficient/inaccurate summary statistics cause information loss due to inflated credible intervals.
    - Solution => Automatic selection and semi-automatic identification of sufficient summary statistics; better model validation checks.
    - Sufficient Statistics vs. Point Acceptance => If acceptance/rejection of the target distribution is based on a target point metric (say, target Euclidean distance to within a tolerance), then a single sequence of point parameter sampling is enough.
    - ABC using point value parameter input vs. ABC using the parameter prior => The simulation should run using the entire parameter prior suite as opposed to the point value parameter set, since only parameter prior enables wholesale acceptance/rejection based on sufficient statistics.
7. Insufficient or Mis-specified Models: This happens when the models chosen for investigation are not representative, or lack predictive power.
    - Solution => Careful selection of models, and evaluation of their predictive power.
8. Priors and Parameter Ranges: This happens when the conclusions are too sensitive to the choice of the priors, thereby rendering the model choices meaningless.

- Solution => Check the sensitivity of the Bayes' factor to the choice of prior. Use the theoretical results available regarding the choice of priors, as well as several varied alternative methods for model validation.

9. Curse of Dimensionality: This causes low parameter acceptance rates and/or results in over-fitting. This makes it hard to distinguish model error from insufficient exploration of the parameter space.
    - Solution => Use applicable and appropriate parsimonious methods for model fitting, as well as penalized information criteria (AIC, BIC etc.) to detect over-fitting. Investigate methods to speed up the process of parameter space exploration.
10. Model Ranking with Summary Statistics: The computation of Bayes' factor with summary statistics may be totally unrelated to the Bayes factors on the original data, which may render the results/conclusions meaningless.
    - Solution => Only use the summary statistics that fulfill the necessary and sufficient condition to produce a consistent Bayesian "Likelihood Model" choice.
11. Implementation: This is caused by low safe-guards/protection to the common assumptions in the simulation and inference processes.
    - Solution => Standard Software Methods, plus "model hedge ratios" and model predicted variances, as well as choice of a divergent set of models/model parameters/"secure" processes (e.g., model VaR, etc.).
12. Inverse ABC: Typically, forward ABC (as in prior -> likelihood -> posterior) is used in conjunction with sufficient statistics approach to determine the appropriate starting prior for the analysis. How about "inverting" the forward ABC to compute the starting point?
    - Reverse ABC used along with adjoint algorithmic/automatic differentiation techniques might enable computation of the path-wise sensitivity, as well as simulation start/appropriate prior calibration.
13. Application to Financial Monte Carlo: In finance, ABC may be used for MC path-wise Greeks/measures, thereby probabilistically inferring the simulation state variate start.
    - Typical financial MC evolution would rules/algorithm would indicate that the explicit computation of the likelihood is hard/infeasible, therefore ABC is appropriate.
14. Target Check Tolerance Bias: Given that most probability distributions (including exponential distribution, in particular normal distributions) are asymmetric around an

arbitrary variate, an estimation bias is introduced whenever the target match is done using a tolerance parameter.

15. Sufficient Statistics Tolerance Bias: Checks on target outcomes within tolerance introduces bias, so moment distributions on outcomes as proxy for the check on target outcomes – this is called sufficient statistics. However, bear in mind that finite number of moments may still not be enough to completely capture the sufficient statistics, e.g., normal distribution needs two moments (μ and σ^2), other exponential distributions need more than two (but finite), and non-exponential may need literally infinite set of moments to be fully characterized.
16. ABC + MAP/Bayes' Estimator: The ABC algorithm using parametric priors is:
    - Generate a distribution of θ based on the prior distribution.
    - Generate the posterior distribution using the model mnemonics, and the prior.
    - Choose the prior distribution that best matches the measured posterior distribution – by default, this automatically corresponds to using MAP.
17. Re-use of the Posterior Samplings: For a given set of input-to-output samplings, different posteriors may be generated from different priors that are assembled by assigning appropriate population weights to the input-to-output samplings.

Measurement and Parametric Calibration

1. Measurement vs. Calibration: Unbiased outcome of a set of measurements for the same set space of inputs – if it exactly the same set of inputs, the measurement outcome will just be the average (in the case of empirical risk minimization). Calibration, however, has inference built into it.
    - Cliché – measurement is "discovery", whereas calibration is "inference".
2. Sources of Output Stochasticity:
    - Measurement Stochasticity => Stochasticity introduced by random measurement error, despite deterministic inputs
        - Handled using the frequentist MLE framework
    - Stochasticity from Parameter Uncertainty => Deterministic input and stochastic output, where the output determinism is eliminated purely due to a diffuse belief system

    - Handled using Bayesian techniques such as MAP, etc.
- Model stochasticity => Deterministic input result in stochastic output owing to model introduced stochasticity
    - Handled using stochastic calibration inside a regression analyzer
- Stochastic Input => Stochastic output occurs purely due to the transformation of the input; further the input should be characterizable.
- Combination of any of the above still may result in output stochasticity. Thus, the corresponding analysis techniques are a combination of the above too.

3. Curve/Parameter Fitting with MLE/MAP: Curve/parameter can only be justified in a logical inference framework. The framework may use MLE (in a frequentist set-up), or MAP (along with Bayes' estimator inside of a cost function) in a Bayesian set-up.
4. Measurement Stochasticity vs. Model Stochasticity: Typical frequentist MLE cost-function based regression only deals with measurement stochasticity. However, distribution of the posterior may also result from model stochasticity. Therefore, de-convolution techniques that separate the measurement stochasticity from model stochasticity are needed.

Regression Analysis

1. Machine Learning as Regression: Learning is construed as a calibration process, where the output of calibration (viz., the parameter) is really produced as a consequence of regression.
2. Causality of Response on Predictor: Often, the predictor and the response variables result from one or more common factor moves. If the common factors are orthogonalizable into precisely that one factor that solely drives predictor and response (and none else, including no idiosyncratic drivers), the dependence shows symptoms of causality, i.e., 100% (spontaneous or delayed) correlation between predictor and response.
    - However, causality still maintains stochasticity. In addition to the driving factor, the measurement errors also contribute to the stochasticity of causal observations set.
3. Regression Analysis Focus: It focuses on the conditional realization of the measured outputs given the observed inputs (rather than the joint realization of the inputs and the outputs), although these two statements are completely equivalent.

4. Frequentist Regression Axiom Set:
   - All the measurements for y_i correspond only to the input vector set $\{x_j\}$.
   - The only source of error is the instrumentation/measurement error, which is random and characterizable.
   - The measurement error bias is zero, i.e.

$$\mu_{Error} = 0$$

but

$$\sigma_{Error}{}^2 \neq 0$$

5. Frequentist Regression Setup: Given the model

$$y = f(x)$$

implying

$$y_i = \beta x_i + \varepsilon$$

where

$$i = 1, \cdots, n$$

are the observations

$$\varepsilon_i \approx \mathcal{N}(0, \sigma^2)$$

i.e., no bias, the frequentist multi-observation likelihood is becomes

$$P(y_1 \mid x_1, \beta, \sigma^2) \cdots P(y_n \mid x_n, \beta, \sigma^2) = \prod_{i=1}^{n} e^{-\frac{(y_i - \beta x_i)^2}{2\sigma^2}}$$

6. Frequentist Measurement Log-likelihood:

$$\Lambda = -\frac{(y_i - \beta x_i)^2}{2\sigma^2}$$

Thus, maximum likelihood estimate corresponds to minimization of the log-likelihood, which in turn works out to least squares minimization (LSM) of the log-likelihood.

7. Log likelihood Least Squares Minimization:

$$\frac{\partial \Lambda}{\partial \beta} = 0$$

implies

$$-\frac{1}{2\sigma^2} \frac{\partial}{\partial \beta} \sum_{i=1}^{n} (y_i - \beta x_i)^2 = 0$$

which results in

$$\sum_{i=1}^{n} x_i y_i - \beta \sum_{i=1}^{n} {x_i}^2 = 0$$

thereby producing

$$\beta = \frac{\sum_{i=1}^{n} x_i y_i}{\sum_{i=1}^{n} {x_i}^2}$$

8. Matrix re-formulation of Likelihood:

$$P(\vec{Y} \mid \vec{X}, \beta, \sigma^2) = \prod_{i=1}^{n} P(y_1 \mid x_1, \beta, \sigma^2) \propto e^{-\frac{1}{2\sigma^2}[\vec{Y}-\beta\vec{X}][\vec{Y}-\beta\vec{X}]}$$

9. Matrix re-formulation of the Least Squares Regressor:

$$\beta = \left[\vec{X}^T\vec{X}\right]^{-1}\vec{X}^T\vec{Y}$$

This is also referred to as the Penrose-Moore pseudo-inverse methodology.

10. Multi-factor Frequentist EM Based Regression – Practice Steps:

- Step #1: Given a model (i.e., model + parameter), find the likelihood of the given outcome $P(y_i \mid \{x_j\}, \{\beta_j\}, \vec{\theta})$, i.e., all the measurements corresponding to a specific set of inputs $\{x_j\}$ where
  - y_i is the observation i , given

$$i = 1, \cdots, n$$

  - $\{x_j\}$ is the input vector j, given

$$j = 1, \cdots, m$$

  - $\{\beta_j\}$ is the parameter co-efficient vector j, given

$$j = 1, \cdots, m$$

  - $\vec{\theta}$ is the set of parameters characterizing the error term ε_i in the input distribution, i.e.

$$y_i = \sum_{j=1}^{m} \beta_j x_j + \varepsilon_i$$

- Step #2: Convolve the measurement set across the same input set, i.e., extract the joint likelihood

$$P(\vec{Y} \mid \vec{X}, \beta, \vec{\theta}) = \prod_{i=1}^{n} P(y_i \mid \{x_j\}, \{\beta_j\}, \vec{\theta})$$

- Step #3: Maximize the joint likelihood $P(\vec{Y} \mid \vec{X}, \beta, \vec{\theta})$
    - Alternatively, maximize the log of joint likelihood.
    - For exponential conjugates, this typically works out to the maximization of some closed form, tractable cost function. Identify that cost function.
- Step #4: Minimize the Cost Function

$$\frac{\partial CostFunction}{\partial \vec{\beta}} = 0$$

gives the corresponding $\hat{\vec{\beta}}$. Verify that

$$\left\| \frac{\partial^2 CostFunction}{\partial \vec{\beta}^2} \right\|_{\vec{\beta}=\hat{\vec{\beta}}} > 0$$

i.e., the cost function at $\hat{\vec{\beta}}$ is a true minimum.

- Step #5: There will need to be m equations, one each for each β_j. Set these up to be solved using a linear/non-linear framework.

11. Multi-node, Multi-factor Frequentist EM Based Regression – Practice Steps:

$$y_i = \sum_{j=1}^{m} \beta_j x_{ij} + \varepsilon_i$$

where

$$\varepsilon_i \approx \mathcal{N}(0, \sigma^2)$$

$$i = 1, \cdots, n$$

and

$$j = 1, \cdots, m$$

The corresponding calibrated β_k is

$$\beta_k = \frac{\sum_{i=1}^{n} x_{ik} \left[y_i - \sum_{\substack{j=1 \\ j \neq k}}^{m} \beta_j x_{ij} \right]}{\sum_{i=1}^{n} {x_{ik}}^2}$$

Re-cast the above to form a frameable equation set as

$$\sum_{j=1}^{m} \beta_j \sum_{i=1}^{n} x_{ij} x_{ik} = \sum_{i=1}^{n} y_i x_{ik}$$

or as

$$\sum_{j=1}^{m} \beta_j \alpha_{jk} = \gamma_k$$

where

$$\alpha_{jk} = \sum_{i=1}^{n} x_{ij} x_{ik}$$

and

$$\gamma_k = \sum_{i=1}^{n} y_i x_{ik}$$

Matrix form of the above is

$$\vec{\alpha}_{JK} = \vec{X}_K{}^T \vec{X}_J$$

and

$$\vec{\gamma}_K = \vec{X}_K{}^T \vec{Y}$$

Thus

$$\vec{\beta} \vec{X}_K{}^T \vec{X}_J = \vec{X}_K{}^T \vec{Y}$$

results in

$$\vec{\beta} \vec{X}^T \vec{X} = \vec{X}^T \vec{Y}$$

Therefore

$$\beta = \left[\vec{X}^T \vec{X}\right]^{-1} \vec{X}^T \vec{Y}$$

Bayesian Regression Analysis

1. Non-Parametric Frequentist/Bayesian Regression: Answers the following question: Which among the input models (plus parameters) provides the most likelihood and/or MAP?
2. Parametric Frequentist/Bayesian Regression: Answers the following question: Assuming that the observation set satisfies MLE/MAP, what should the model parameters be?
3. Terminology fix – Regression Likelihood vs. Typical Bayesian Likelihood: While in typical frequentist treatments the likelihood is written as $(y_i \mid x_i)$ in Bayesian likelihood it is written as $L(y_i \mid x_i, \theta)$ or $L(y_i \mid \theta)$.
4. Problem with MLE-based Calibration: MLE dictates that only the model (due to the presence of instrumentation uncertainty and/or the inherent model transform stochasticity) causes stochasticity in the output. Bayesian techniques, however, allow for the stochasticity to arise out of imprecision resulting from model specification too.
5. Bayesian Basis for Observations Analysis: Treating the observations $\{y_i\}_{i=1}^{m}$ as the fixed axiomatic starting points, both the frequentist and the Bayesian techniques try to say something about the world that is limited to these observations – Bayesian techniques, however, also try to say something about the parameter set that result in these observations.
6. Parameter vs. Model Confusion: Wherever a Bayesian treatise says $P(y \mid \theta)$ it means $P(y \mid Model, \theta)$. Further, both $P(y \mid \theta)$ or $P(y \mid Model, \theta)$ simply refer to the probabilistic belief of the truthness of the model (the prior is the model belief).
    - Probability of y given θ (represented as $P(y \mid \theta)$) is to be read as: ***What is the probability that y is valid in that particular world where θ is valid***? (See Figure 1). Specifically

$$P(y \mid \theta) = P(y = VALID \mid \theta = VALID)$$
$$= \frac{P(y = VALID, \theta = VALID)}{P(y = INVALID, \theta = VALID) + P(y = VALID \mid \theta = VALID)}$$

or

$$P(y \mid \theta) = \frac{P(y, \theta)}{P(!\, y, \theta) + P(y, \theta)}$$

- Likewise, probability of θ given y (represented as $P(\theta \mid y)$) is to be read as: ***What is the probability that*** θ ***is valid in that particular world where*** y ***is valid***? (Again, see Figure 1). Specifically

$$P(\theta \mid y) = P(\theta = VALID \mid y = VALID)$$
$$= \frac{P(y = VALID, \theta = VALID)}{P(y = VALID, \theta = INVALID) + P(y = VALID \mid \theta = VALID)}$$

or

$$P(\theta \mid y) = \frac{P(y, \theta)}{P(y, !\, \theta) + P(y, \theta)}$$

7. Bayes' Theorem Derivation:

$$P(\theta = VALID \mid y = VALID)$$
$$= \frac{P(y = VALID, \theta = VALID)}{P(y = VALID, \theta = INVALID) + P(y = VALID, \theta = VALID)}$$

$$P(\theta = VALID \mid y = VALID) = \frac{P(y = VALID, \theta = VALID)}{P(y = VALID)}$$
$$= \frac{P(y = VALID, \theta = VALID)}{P(y = VALID)} \frac{P(\theta = VALID)}{P(\theta = VALID)}$$

$$P(\theta = VALID \mid y = VALID) = \frac{P(y = VALID, \theta = VALID)}{P(y = VALID)}$$
$$= \frac{P(\theta = VALID)}{P(y = VALID)} \frac{P(y = VALID, \theta = VALID)}{P(\theta = VALID)}$$

$$P(\theta = VALID \mid y = VALID)$$
$$= \frac{P(\theta = VALID)}{P(y = VALID)} \frac{P(y = VALID, \theta = VALID)}{P(y = INVALID, \theta = VALID) + P(y = VALID, \theta = VALID)}$$

$$P(\theta = VALID \mid y = VALID) = \frac{P(\theta = VALID)}{P(y = VALID)} P(y = VALID \mid \theta = VALID)$$

where we have used

$$P(y = VALID \mid \theta = VALID)$$
$$= \frac{P(y = VALID, \theta = VALID)}{P(y = INVALID, \theta = VALID) + P(y = VALID, \theta = VALID)}$$

8. Bayes' Theorem Derivation: Circling back the Definitions:

$$P(y = VALID \mid \theta = VALID) = P(y \mid \theta)$$

$$P(\theta = VALID \mid y = VALID) = P(\theta \mid y)$$

Thus, from above

$$P(\theta \mid y) = P(y \mid \theta) \frac{P(\theta)}{P(y)}$$

Finally

$$Likelihood = \frac{P(\theta)}{P(y)} = \frac{P(y \mid \theta)}{[P(y, !\theta) + P(y, \theta)][P(!y, \theta) + P(y, \theta)]}$$

9. Posterior Maximum vs. Maximum of $P(y, \theta)$: We are really interested in maximizing

$$P(y, \theta) = P(y \mid \theta)P(\theta) = P(\theta \mid y)P(y)$$

Since $P(y)$ is the same, no matter what θ is, the above, in practice, reduces to maximizing

$$P(\theta \mid y) = \frac{P(y \mid \theta)}{P(y)} P(\theta)$$

i.e., it corresponds to maximizing the posterior.

10. Incorporating Parameter Belief Uncertainty: Assuming that the parameter belief process is modeled as

$$\beta_k \sim \mathcal{N}(\mu_k, {\sigma_k}^2)$$

$$\beta_{k,MAP} = \frac{{\sigma_k}^2 \sum_{i=1}^{n} x_{ik} \left[y_i - \sum_{\substack{j=1 \\ j \neq k}}^{m} \beta_j x_{ij} \right] + \sigma^2 \mu_k}{{\sigma_k}^2 \sum_{i=1}^{n} {x_{ik}}^2 + \sigma^2 \mu_k} = \frac{{\sigma_k}^2 \beta_{k,MLE} \sum_{i=1}^{n} {x_{ik}}^2 + \sigma^2 \mu_k}{{\sigma_k}^2 \sum_{i=1}^{n} {x_{ik}}^2 + \sigma^2 \mu_k}$$

11. Matrix Form of the above:

$$\vec{\beta}_{MAP} = \left[{\vec{\sigma}_\beta}^2 \vec{X}^T \vec{X} + {\vec{\sigma}_k}^2 \vec{\mu}_\beta \right]^{-1} \left[{\vec{\sigma}_\beta}^2 \vec{\beta}_{MLE} \vec{X}^T \vec{X} + {\vec{\sigma}_k}^2 \vec{\mu}_\beta \right]$$

12. Multi-factor Bayesian EM Based Regression – Practice Steps: The extra steps required to enhance MLE based regression for Bayesian analysis are listed here:
    - Choose a parameter prior
    - Convolve the likelihood with the parametric distribution
    - Work out the prior parameters that calibrate to the maximization of the log MAP
    - Re-express $\vec{\beta}_{MAP}$ in terms of $\vec{\beta}_{MLE}$
13. Inferring the Prior from the Observed Posterior: In place of parametrically specifying the prior, its moments may also be inferred from the observed posterior and the likelihood.
14. Alternate Parameter Calibration using Cost Functions: In addition to MAP extensions to MLE, calibration may also be performed by optimizing (minimizing typically) a cost function from the given posterior distribution $\Pi(\theta)$, i.e.

$$\mathbb{E}_{\Pi}[\hat{\theta}(x)] = \int C\left(\theta, \hat{\theta}(x)\right) \Pi(\theta) d\theta$$

Here $C\left(\theta, \hat{\theta}(x)\right)$ is the Cost Function, $\Pi(\theta)$ is the posterior distribution, and $\hat{\theta}(x)$ is referred to as the Bayes' estimator.

- The Bayes' estimator may be looked at as a pivot $\hat{\theta}(x)$ in the parameter space from which the departure cost is measured, i.e.

$$C\left(\theta, \hat{\theta}(x)\right) = f\left(\theta - \hat{\theta}(x)\right)$$

- Practical meaning of the Bayes' estimator => It provides an alternate "central cost pivot" anchor (this pivot, in general, will not be a traditional central measure such as mean/median/mode), and depending upon the model/prior combination, the prior may simply end up being the mean/median/mode.
- MAP viewed in the Cost Function/Bayes' Estimator Framework => MAP is essentially a zero-order cost function, so there is not "estimator" corresponding to it – really!

15. <u>Typical Cost Function/Bayes' Estimator Combination</u>:
    - Mean Squared Error (MSE) Minimization as the Cost Function => Minimization of MSE $\mathbb{E}\left[\left\{\theta - \hat{\theta}(x)\right\}^2\right]$ results in

$$\hat{\theta}(x) \Longrightarrow \mathbb{E}[\theta]$$

      the mean.
    - MSE as Cost Function

$$x \sim \mathcal{N}(0, \sigma^2)$$

      and prior

$$\theta \sim \mathcal{N}(\mu, \tau^2)$$

In this case, the posterior is also normal, and the Bayes' estimator is given as

$$\hat{\theta}(x) = \frac{\sigma^2}{\sigma^2 + \tau^2}\mu + \frac{\sigma^2}{\sigma^2 + \tau^2}x$$

- MSE as Cost Function, x is Poisson i.i.d, and prior

$$\theta \sim \Gamma(a, b)$$

In this case, the posterior is also Gamma (Γ), and the Bayes' estimator is given as

$$\hat{\theta}(x) = \frac{nx + a}{n + \frac{1}{a}}$$

- MSE as Cost Function, x is Uniform i.i.d

$$x \sim U(0, \theta)$$

and prior

$$\theta \sim Pareto(\theta_0, a)$$

In this case, the posterior is also Pareto, and the Bayes' estimator is given as

$$\hat{\theta}(x) = \frac{(n + a)\max(\theta_0, x_1, \cdots, x_n)}{a + n - 1}$$

- Quintile Cost Function =>

$$C\left(\theta, \hat{\theta}(x)\right) = a\left|\theta - \hat{\theta}(x)\right|$$

for

$$\theta - \hat{\theta}(x) \geq 0$$

and

$$C\left(\theta, \hat{\theta}(x)\right) = b\left|\theta - \hat{\theta}(x)\right|$$

for

$$\theta - \hat{\theta}(x) \leq 0$$

In this case

$$\hat{\theta}(x) = \frac{a}{a+b}$$

Extensions to Regression Analysis

1. Penalizing Smootheners: Penalizing smootheners are the consequence of Bayes' estimation applied on the Quadratic Penalties with Gaussian Priors (also referred to with maxim "The Penalty is the Prior").
2. Non-Gaussian Priors: In this case, the smoothing estimation process is called the Generalized Linear Model.
3. Inference Schemes:
    - Help probabilistically infer what happened in the past.
    - Set up/identify a probabilistic causal framework.

- Inference schemes may be less dependent on the precision of the models/physics than prediction schemes.

4. Prediction Schemes:
    - Help predict probabilistic future outcomes.
    - Significantly dependent upon the precision of the models.
    - Automatically built for outcome variance handling/hedging.
5. Inference/Prediction vs. "Smoothness": "Smoothness" schemes are simply just good mathematical/cognition citizen schemes. Clearly this is different from inference/prediction schemes as laid out.
6. Inference Based Curve Fitting: Here, the target function is treated as a realization of a stochastic process. Hence the model hypothesis estimation, credible interval estimate, etc: are all automatically available as part of the Bayesian Inference Framework.
7. Bayesian Optimizing Inference Schemes: Are all optimizing inference schemes castable as Bayesian? Clearly, given that regression schemes are part of a calibration framework, they can be reduced to Bayesian formulation, with specific priors having been extracted for smoothing/optimizing regressors. How about other inference schemes?

Spline Analysis of Bayesian Systems

1. Prior Spline to Posterior Spline: Given that any function may be represented by a suitable choice of splines, the prior and posterior may also be represented by them, and might potentially simply some of the analysis/formulation.
    - However, since probabilities can extend in the variate ranges $[-\infty, +\infty]$ perhaps specialized basis representations should be chosen; e.g., from $[-\infty, a]$ for the left extrapolator spline, and $[b, +\infty]$ for the right extrapolator spline. $[a, b]$ should be the range of workability.
    - Prior to posterior spline formulation =>
        - Choose segment i.
        - Prior Spline => $\left[R_{jk}\right]_i$.

- Posterior Spline => $\left[O_{jk}\right]_i$.
- Stating that

$$\text{Prior}_i \times \Lambda_{ij} = \text{Posterior}_j$$

where Λ_{ij} is the Likelihood tensor, we can see that

$$\Lambda_{ij} = [R_{kl}]_i \times \left[R_{pq}\right]_j$$

- More specifically

$$\Lambda_i = \left[R_{jk}\right]_i [I]_{jkpq} \times \left[O_{pq}\right]_i [I]_{pqjk}$$

where $[I]_i$ is the identity matrix.

Figure #1
Observation/Parameter Quadrant

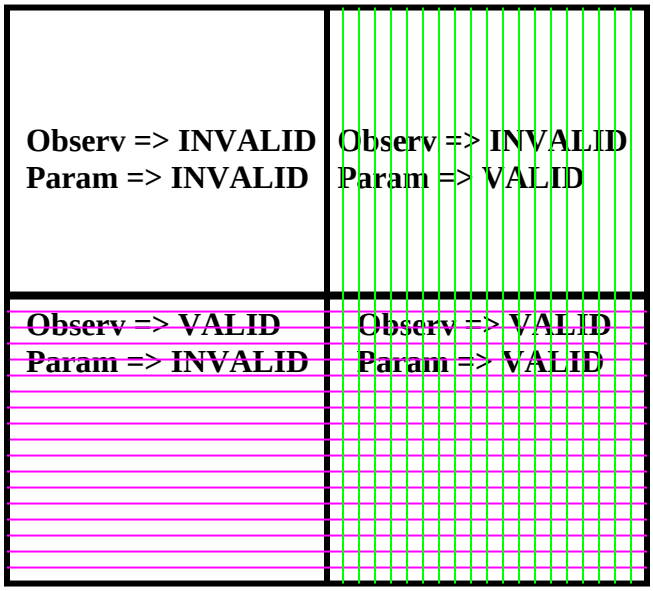

Figure #1
Data Set Orthogonal in its Native Representative Basis

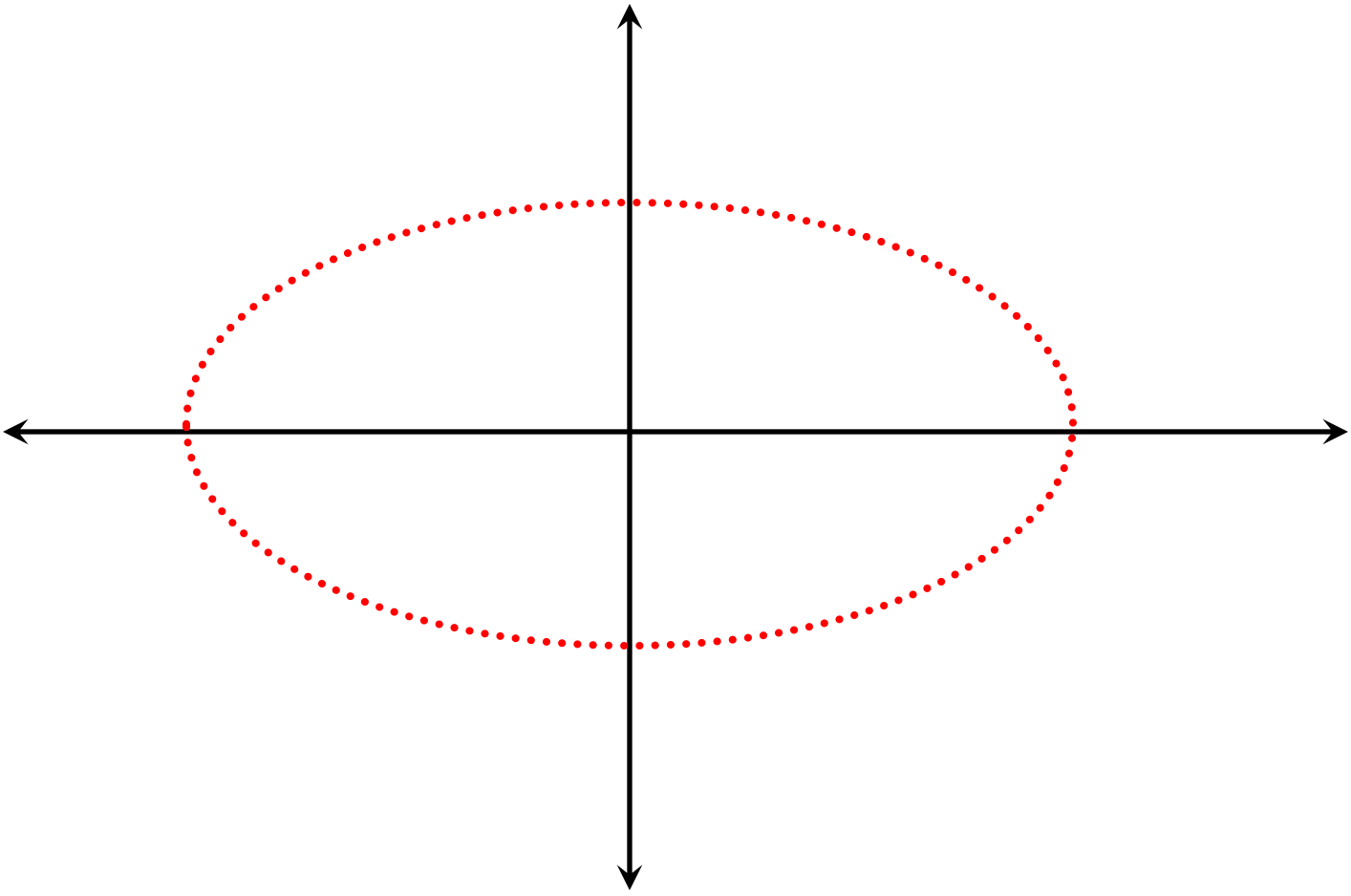

Figure #2
Data Set NOT Orthogonal in its Native Representative Basis

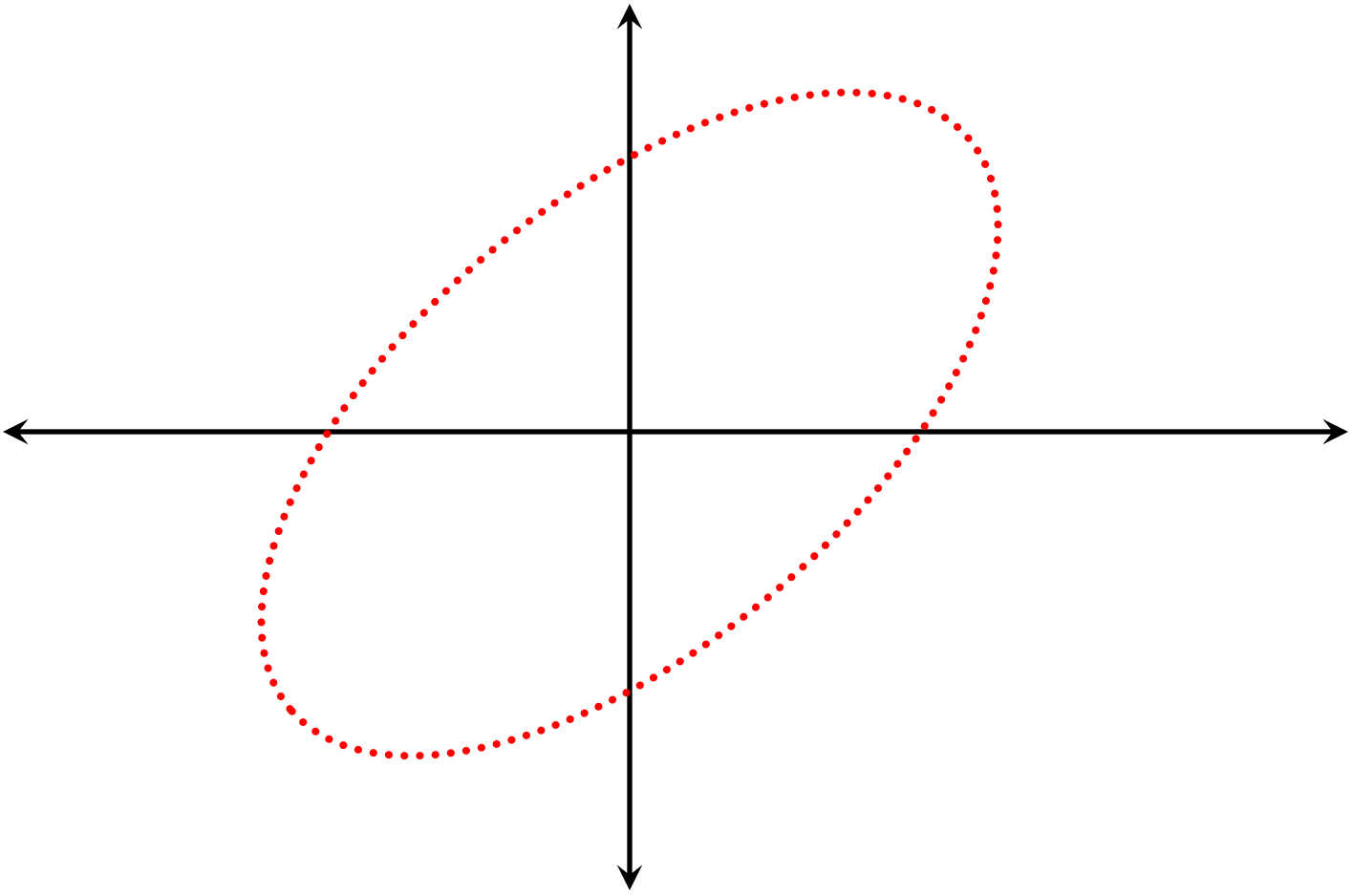

Figure #3
Principal Components that Diagonalize the Data Set