Class Notes (experimental)

Jonathan Rosenblatt

April 9, 2015

Contents

1 Estimation

In this section, we present several estimation principeles. Their properties are not discussed, as the section is merely a reminder and a preparation for Section 2. These concepts and examples can be found in many introductory books to statistics. I particularly recommend [Wasserman, 2004].

1.1 Moment matching

The fundamental idea: match empirical moments to theoretical. I.e., estimate

$$E[g(X)]$$

by

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

where $\mathbb{E}[g(X)] := \frac{1}{n}\sum_i g(X_i)$, is the empirical mean.

Example 1 (Exponential Rate)**.** Estimate λ in $X_i \sim exp(\lambda)$, $i = 1, \ldots, n$, i.i.d. $E[X] = 1/\lambda$. $\Rightarrow \hat{\lambda} = 1/\mathbb{E}[X]$

Example 2 (Linear Regression)**.** Estimate β in $Y \sim \mathcal{N}(X\beta, \sigma^2 I)$, a p dimensional random vector. $E[Y] = X\beta$ and $\mathbb{E}[Y] = y$. Clearly, moment mathing won't work because no β satistifes $X\beta = Y$. A technical workaround: Since β is p dimensional, I need to find some $g(Y) : \mathbb{R}^n \mapsto \mathbb{R}^p$. Well, $g(Y) := XY$ is such a mapping. I will use it, even though my technical justification is currently unsatisfactory. We thus have: $E[X'Y] = X'X\beta$ which I match to $\mathbb{E}[X'Y] = X'y$:

$$X'X\beta = X'y \Rightarrow \hat{\beta} = (X'X)^{-1}X'y.$$

1.2 Quantile matching

The fundamental idea: match empirical quantiles to theoretical. Denoting by $F_X(t)$ the CDF of X, then $F_X^{-1}(\alpha)$ is the α quantile of X. Also denoting by $\mathbb{F}_X(t)$ the Empirical CDF of $X_1, \ldots, X_n$, then $\mathbb{F}_X^{-1}(\alpha)$ is the α quantile of $X_1, \ldots, X_n$. The quantile matching method thus implies estimating

$$F_X^{-1}(\alpha)$$

by

$$\mathbb{F}_X^{-1}(\alpha).$$

Example 3 (Exponential rate)**.** Estimate λ in $X_i \sim exp(\lambda)$, $i = 1, \ldots, n$, i.i.d.

$$\begin{aligned} F_X(t) =& 1 - \exp(-\lambda t) = \alpha \Rightarrow \\ F_X^{-1}(\alpha) =& \frac{-\log(1-\alpha)}{\lambda} \Rightarrow \\ F_X^{-1}(0.5) =& \frac{-\log(0.5)}{\lambda} \Rightarrow \\ \hat{\lambda} =& \frac{-\log(0.5)}{\mathbb{F}_X^{-1}(0.5)}. \end{aligned}$$

1.3 Maximum Likelihood

The fundamental idea is that if the data generating process (i.e., the *sampling distribution*) can be assumed, then the observations are probably some high probability instance of this process, and not a low probability event: Let $X_1, \ldots, X_n \sim P_\theta$, with density (or probability) $p_\theta(X_1, \ldots, X_n)$. Denote the likelihood, as a function of θ: $\mathcal{L}(\theta) : p_\theta(X_1, \ldots, X_n)$. Then $\hat{\theta}_{ML} := argmax_\theta\{\mathcal{L}(\theta)\}$.

Example 4 (Exponential rate)**.** Estimate λ in $X_i \sim exp(\lambda)$, $i = 1, \ldots, n$, i.i.d. Using the exponential PDF and the i.i.d. assumption

$$\mathcal{L}(\lambda) = \lambda^n \exp(-\lambda \sum_i X_i).$$

Using a monotone mapping such as the log, does not change the $argmax$. Denoting $L(\theta) := \log(\mathcal{L}(\theta))$, we have

$$L(\lambda) = n \log(\lambda) - \lambda \sum_i X_i.$$

By differentiating and equating 0, we get $\hat{\lambda}_{ML} = 1/\mathbb{E}[X]$.

Example 5 (Discrete time Markov Chain)**.** Estimate the transition probabilities, p_1 and p_2 in a two state, $\{0,1\}$, discrete time, Markov chain where: $P(X_{t+1}=1|X_t=0)=p_1$ and $P(X_{t+1}=1|X_t=1)=p_2$. The likelihood:

$$\mathcal{L}(p_1,p_2)=P(X_1,\ldots,X_n;p_1,p_2)=\prod_{t=0}^{T}P(X_{t+1}=x_{t+1}|X_t=x_t).$$

We denote n_{ij} the number of observed transitions from i to j and get that $\hat{p}_1=\frac{n_{01}}{n_{01}+n_{00}}$, and that $\hat{p}_2=\frac{n_{11}}{n_{11}+n_{10}}$.

Remark 1. Well, this is a rather artificial example, as because of the Markov property, and the stationarity of the process, we only need to look at transition events, themselves Brenoulli distributed. This example does show, however, the power of the ML method to deal with non i.i.d. samples. As does the next example.

Example 6 (Brownian motion with drift)**.** Estimate the drift parameter a, in a discrete time Gaussian process where: $X_{t+1}=X_t+\varepsilon;\varepsilon\sim\mathcal{N}(0,\sigma^2)\Rightarrow X_{t+1}|X_t\sim\mathcal{N}(aX_t,\sigma^2)$.

We start with the conditional density at time $t+1$:

$$p_{X_{t+1}|X_t=x_t}(x_{t+1})=(2\pi\sigma^2)^{-1/2}\exp\left(-\frac{1}{2\sigma^2}(x_{t+1}-ax_t)^2\right).$$

Moving to the likelihood:

$$\mathcal{L}(a)=(2\pi\sigma^2)^{-T/2}\exp\left(-\frac{1}{2\sigma^2}\sum_{t=1}^{T}(x_{t+1}-ax_t)^2\right).$$

Differentiating with respect to a and equating 0 we get $\hat{a}_{ML}=\frac{\sum x_{t+1}x_t}{\sum x_t^2}$.

We again see the power of the ML device. Could we have arrive to this estimator by intuiton alone? Hmmmm... maybe. See that $Cov[X_{t+1},X_t]=a\ Var[X_t]\Rightarrow a=\frac{Cov[X_{t+1},X_t]}{Var[X_t]}$. So a can also be derived using the moment matching method which is probably more intuitive.

Example 7 (Linear Regression)**.** Estimate β in $Y\sim\mathcal{N}(X\beta,\sigma^2 I)$, a p dimensional random vector. Recalling the multivariate Gaussian PDF:

$$p_{\mu,\Sigma}(y)=(2\pi)^{-n/2}|\Sigma|^{-1/2}\exp\left(-\frac{1}{2}(y-\mu)'\Sigma^{-1}(y-\mu)\right)$$

So in the regression setup:

$$\mathcal{L}(\beta)=p_{\beta,\sigma^2}(y)=(2\pi)^{-n/2}|\sigma^2 I|^{-1/2}\exp\left(-\frac{1}{2\sigma^2}\|y-X\beta\|^2\right)$$

1.4 M-Estimation and Empirical Risk Minimization

M-Estimation, know as Empirical Risk Minimizaton (ERM) in the machine learning literature, is a very wide framework which stems from statistical desicion theory. The underlying idea is that each realization of X incurs some loss, and we seek to find a "policy", in this case a parameter, θ^* that minimizes the average loss. In the econometric literature, we dot not incur a loss, but rather a utility, we thus seek a policy that maximizes the average utility.

Define a loss function $l(X;\theta)$, and a risk function, being the expected loss, $R(\theta) := E[l(X;\theta)]$. Then

Risk Function

$$\theta^* := argmin_\theta \{R(\theta)\}. \tag{1}$$

As we do not know the distribution of X, we cannot solve Eq.(1), so we minimize the *empirical* risk. Define the empirical risk as $\mathbb{R}(\theta) := \mathbb{E}[l(X;\theta)]$, then

Empirical Risk

$$\hat{\theta} := argmin_\theta \{\mathbb{R}(\theta)\}. \tag{2}$$

Remark 2. The risk function, $R(\theta)$ defined above

Example 8 (Squared Loss)**.** Let $l(X;\theta) = (X-\theta)^2$. Then $R(\theta) = E[(X-\theta)^2] = (E[X]-\theta)^2 + Var[X]$. Clearly $Var[X]$ does not depend on θ so that $R(\theta)$ is minimized by $\theta^* = E[X]$. **We thus say that the expectation of a random variable is the minimizer of the squared loss.**

How do we estimate the population expectation? Well a natural estimator is the empirical mean, which is also the minimizer of the empirical risk $\mathbb{R}(X)$. The proof is immediate by differentiating.

Example 9 (Least Squares Regression)**.** Define the loss $l(Y,X;\beta) := \frac{1}{2}(Y - X\beta)^2$. Computing the risk, $E[\|Y - X\beta\|^2]$ will require dealing with the X's by either assuming the *Generative Model*[1], as expectation is taken over X and Y. We don't really care about that right now. We merely want to see that the empirical risk minimizer, is actually the classical OLS Regression. And well, it is, by definition...

Generative Model

$$\mathbb{R}(\beta) = \sum_{i=1}^{n} \frac{1}{2}(y - x_i\beta)^2 = \frac{1}{2}\|y - x\beta\|^2.$$

[1] A Generative Model is a supervised learning problem where the we use the assumed distribution of the Xs and not only $Y|X$. The latter are know as Discriminative Models.

Minimization is easiest with vector derivatives, but I will stick to regular derivatives:

$$\frac{\partial \mathbb{R}(\beta)}{\partial \beta_j} = \sum_i \left[(y_i - \sum_{j=1}^{p} x_{ij}\beta_j)(-x_{ij}) \right]$$

Equating 0 yields $\hat{\beta}_j = \frac{\sum_i y_i x_{ij}}{\sum_i x_{ij}^2}$. Solving for all j's and putting in matrix notation we get

$$\hat{\beta}_{OLS} = (X'X)^{-1}X'y. \tag{3}$$

1.5 Notes

Maximum Likelihood If we set the loss function to be the negative log likelihood of the (true) sampling distribution, we see that maximum likelihood estimators in independent samples are actually a certain type of M-estimators.

2 From Estimation to Supervised Learning

This section draws from Hastie et al. [2003] and Shalev-Shwartz and Ben-David [2014]. The former is freely available online. For a softer introduction, with more hands-on examples, see James et al. [2013]. All books are very well written and strongly recommended.

2.1 Empirical Risk Minimization (ERM) and Inductive Bias

In Supervised Learning problems where we want to extract the relation $y = f(x)$ between attributes x and some outcome y. The attributes, also know as features, or predictors, are assumed to belong to some *feature space* $\mathcal{X}$.

Feature Space

In particular, we don't need to explain the causal process relating the two, so there is no need to commit to a sampling distribution. The implied ERM problem is thus

$$\hat{f}(x) = argmin_f \left\{ \sum_i l(y_i - f(x_i)) \right\}. \tag{4}$$

Alas, there are clearly infinitely many f for which $\mathbb{R}(\hat{f}(x)) = 0$, in particular, all those where $\hat{f}(x_i) = y_i$. All these f feel like very bad predictors, as they

overfit the observed data, at a cost of generalizability. We will formalize this intuition in Section 4.

Overfitting

We need to make sure that we do not learn overly complex poor predictors. Motivated by the fact that humans approach new problems equipped with their past experience, this regularization is called *Inductive Bias*. There are several ways to introduce this bias, which can be combined:

Inductive Bias

The Hypothesis Class We typically do not allow f to be "any function" but rather restrict it to belong to a certain class. In the machine learning terminology, f is a Hypothesis, and it belongs to $\mathcal{F}$ which is the Hypothesis Class.

Prior Knowledge We do not need to treat all $f \in \mathcal{F}$ equivalently. We might have prior preferences towards particular f's and we can introduce these preference in the learning process. This is called *Regularization*.

Non ERM Approaches Many learning problems can be cast as ERM problems, but another way to introduce bias is by learning f via some other scheme, which cannot be cast as an ERM problem. Learning algorithms that cannot be cast as ERMs include: Nearest Neighbour, Kernel Smoothing, Boosting. Naive Bayes and Fisher's LDA are also not considered ERMs, but they can be cast as such [TODO: verify].

We now proceed to show that many supervised learning algorithms are in fact ERMs with some type of inductive bias.

2.2 Linear Regression (OLS)

As seen in Example 9, by adopting a squared error loss, and restricting $\mathcal{F}$ by assuming f is a linear function of x, we get the OLS problem. In this case, learning f is effectively the same as learning β as they are isomorphic.

Remark 3. We distinguish between OLS and Linear Regression. In these notes, we refer to linear regression when we assume that the data generating process it actually $y = x\beta + \varepsilon$, whereas in OLS we merely fit a linear function without claiming it is the data generating one.

2.3 Ridge Regression

Consider the Ridge regression problem:

$$argmin_\beta \left\{ \frac{1}{n} \sum_i (y_i - x_i\beta)^2 + \frac{\lambda}{2}\|\beta\|^2 \right\} \tag{5}$$

$$\hat{\beta}_{Ridge} = (X'X + \lambda I)^{-1} X'y \tag{6}$$

We can see that again, $\mathcal{F}$ is restricted to be the space of linear functions of x, but we also add a regularization that favors the linear functions with small coefficients.

The regularization of β can have several interpretations and justifications.

A mathematical device Strengthening the diagonal of $X'X$ makes is more easily invertible. This is a standard tool in applied mathematics called Tikhonov Regularization. It is also helpful when dealing with multicolinearity, as $(X'X + \lambda I)$ is always invertible.

A Subjective Bayesian View If we believe that β should be small; say our believes can be quantified by $\beta \sim \mathcal{N}(0, \lambda I)$, then the Ridge solution is actually the mean of our posterior believes on $\beta|y$.

Whatever the justification may be, it can be easily shown that $\frac{\partial R(\lambda,\beta)}{\partial \lambda}$ at $\lambda = 0$ is negative, thus, we can only improve the predictions by introducing some regularization.

For more on Ridge regression see Hastie et al. [2003].

2.4 LASSO

Consider the LASSO problem:

$$argmin_\beta \left\{ \frac{1}{n} \sum_i (y_i - x_i\beta)^2 + \lambda\|\beta\|_1^2 \right\} \tag{7}$$

As can be seen, just like in Ridge regression, $\mathcal{F}$ is restricted to linear functions. The regularization however differs. Instead of l_2 penalty, we use an l_1 penalty. Eq.(7) does not have a closed form solution for $\hat{\beta}$ but the LARS algorithm, a quadratic programming algorithm, solves it efficiently.

The LASSO has gained much popularity as it has the property that $\hat{\beta}_{LASSO}$ has many zero entries. It is thus said to be *sparse*. The sparsity property is very attractive as it acts as a model selection method, allowing to consider Xs where $p > n$, and making predictions computationally efficient.

Sparsity

The sparsity property can be demonstrated for the orthogonal design case ($X'X = I$) where $\hat{\beta}$ admits a closed form solution:

$$\hat{\beta}_{j,LASSO} = sign(\beta_j)\left[|\hat{\beta}_{j,OLS}| - \frac{\lambda}{2}\right]_+ . \tag{8}$$

We thus see that the LASSO actually performs *soft thresholding* on the OLS estimates.

Soft Thresholding

2.5 Logistic Regression

The logistic regression is the first categorical prediction problem. I.e., the outcome y is not a continuous variable, but rather takes values in some finite set $\mathcal{G}$. In the logistic regression problem, it can take two possible values. In the statistical literature, y is encoded as $\mathcal{G} = \{0, 1\}$ and f is assumed to take to take the following form:

Categorical Prediction

$$P(y = 1|x) = \Psi(x\beta) \tag{9}$$

$$\Psi(t) = \frac{1}{1 + e^{-t}} \tag{10}$$

The hypothesis class $\mathcal{F}$ is thus all $f(x) = \Psi(x\beta)$. In the $\{0, 1\}$ encoding, the loss is the negative log likelihood, i.e.:

$$l(y, x, \beta) = -\log\left[\Psi(x\beta)^y(1 - \Psi(x\beta))^{1-y}\right]. \tag{11}$$

In the learning literature it is more common for $\{1, -1\}$ encoding of y in which case the loss is

$$l(y, x, \beta) = -\log\left[1 + \exp(-yf(x))\right]. \tag{12}$$

How to classify? In the $\{0, 1\}$ encoding, we predict class 1 if $\Psi(x\beta) > 0.5$ and class 0 otherwise. The logistic problem thus defines a separating hyperplane $\mathbb{L}$ between the classes: $\mathbb{L} = \{x : f(x) = 0.5\}$.

In the $\{1, -1\}$ encoding, we predict class 1 if $\Psi(x\beta) > 0$ and class 0 otherwise. The plane $\mathbb{L}$ is clearly invariant to the encoding of y.

Remark 4 (Log Odds Ratio)**.** The formulation above, implies that the log odds ratio is linear in the predictors:

$$\log\frac{P(y = 1|x)}{P(y = 0|x)} = x\beta$$

Remark 5 (GLMs)**.** Logistic regression is a particular instance of the very developped theory of Generalized Linear Models. These models include the OLS, Probit Regression, Poisson Regression, Quasi Likelihood, Multinomial Regression, Proportional Odds regression and more. The ultimate reference on the matter is McCullagh and Nelder [1989]. GLM

2.6 Regression Classifier

Can we use the OLS framework for prediction? Yes! With proper encoding of y. Solving the same problem from Example 9 by encoding y as $\{0, 1\}$ gives us the linear separating hyperplane $\mathbb{L} : \{x : x\hat{\beta}_{OLS} = 0.5\}$.

Remark 6. We can interpret $\hat{y}$ as the probability of an event, but there a slight technical difficulty as $\hat{y}$ might actually be smaller than 0 or larger than 1.

2.7 Linear Support Vector Machines (SVM)

We will now not assume anything on the data, and seek for a hyperplane that seperates two classes. This, purely geometrical intuition, was the one that motivated Vapnik's support vector classifier [Vapnik, 1998]. In this section we will see the this geometrical intuition can also be seen as an ERM problem over a linear hypothesis class.

Problem Setup Encode y as $\mathcal{G} = \{-1, 1\}$. Define a plane $\mathbb{L} = \{x : f(x) = 0\}$, and assume a linear hypothesis class, $f(x) = x\beta + \beta_0$. Now find the plane $\mathbb{L}$ that maximizes the (sum of) distances to the data points. We call the minimal distance from $\mathbb{L}$ to the data points, the *Margin*, and denote it by M.

To state the optimization problem, we need to note that $f(x) = x\beta + \beta_0$ is not only the value of our classifier, but it is actually proportional to the signed distance of x from $\mathbb{L}$.

Proof. The distance of x to $\mathbb{L}$ is defined as $\min_{x_0 \in \mathbb{L}}\{\|x - x_0\|\}$. Note $\beta^* := \beta/\|\beta\|$ is a normal vector to $\mathbb{L}$, since $\mathbb{L} = \{x : x\beta + \beta_0 = 0\}$, so that for $x_1, x_2 \in \mathbb{L} \Rightarrow \beta(x_1 - x_2) = 0$. Now $x - x_0$ is orthogonal to $\mathbb{L}$ because x_0 is, by definition, the orthogonal projection of x onto $\mathbb{L}$. Since $x - x_0$ are both orthogonal to $\mathbb{L}$, they are linearly dependent, so that by the Cauchy Schwarz inequality $\|\beta^*\|\|x - x_0\| = \|\beta^*(x - x_0)\|$. Now recalling that $\|\beta^*\| = 1$ and $\beta^* x_0 = -\beta_0/\|\beta\|$ we have $\|x - x_0\| = \frac{1}{\|\beta\|}(x\beta + \beta_0) = \frac{1}{\|\beta\|} f(x)$. □

Using this fact, then $y_i f(x_i)$ is the distance from $\mathbb{L}$ to point i, positive for correct classifications and negative for incorrect classification. The (linear) support vector classifier is defined as the solution to

$$\max_{\beta,\beta_0} \{M \quad s.t. \quad \forall i : y_i f(x_i) \geq M, \quad \|\beta\| = 1\} \tag{13}$$

If the data is not separable by a plane, we need to allow some slack. We thus replace $y_i f(x_i) \geq M$ with $y_i f(x_i) \geq M(1 - \xi_i)$, for $\xi_i > 0$ but require that the missclassifications are controlled using a regularization parameter C: $\sum_i \xi_i \leq C$. Eq.(13) now becomes [Hastie et al., 2003, Eq.(12.25)]

$$\max_{\beta,\beta_0} \left\{ M \quad s.t. \quad \forall i : y_i f(x_i) \geq M(1 - \xi_i),\ \|\beta\| = 1,\ \sum_i \xi_i \leq C,\ \forall i : \xi_i \ \geq 0 \right\} \tag{14}$$

This is the classical geometrical motivation for support vector classification problem. The literature now typically discusses how to efficiently optimize this problem, which is done via the dual formulation of Eq.(14). We will not go in this direction, but rather, note that Eq.(14) can be restated as an ERM problem:

$$\min_{\beta,\beta_0} \left\{ \sum_i [1 - y_i f(x_i)]_+ + \frac{\lambda}{2} \|\beta\|_2^2 \right\} \tag{15}$$

Eq.(15) thus reveals that the linear SVM is actually an ERM problem, over a linear hypothesis class, with l_2 regularization of β.

See Section 12 in Hastie et al. [2003] for more details on SVMs.

Remark 7 (Name Origins)**.** SVM takes its name from the fact that $\hat{\beta}_{SVM} = \sum_i \hat{\alpha}_i y_i x_i$. The explicit form of $\hat{\alpha}_i$ can be found in [Hastie et al., 2003, Section 12.2.1]. For our purpose, it suffices to note that $\hat{\alpha}_i$ will be 0 for all data points far away from $\mathbb{L}$. The data points for which $\hat{\alpha}_i > 0$ are the *support vectors*, which give the method its name.

Remark 8 (Solve the right problem)**.** Comparing with the logistic regression, and the linear classifier, we see that the SVM cares only about the decision boundary $\mathbb{L}$. Indeed, if only interested in predictions, estimating probabilities is a needless complication. As Put by Vapnik:

> When solving a given problem, try to avoid a more general problem as an intermediate step.

Then again, if the assumed logistic model of the logistic regression, is actually a good approximation of reality, then it will outperform the SVM as it borrows information from all of the data, and not only the support vectors.

2.8 Generalized Additive Models (GAMs)

A way to allow for a more broad hypothesis calss $\mathcal{F}$, that is still not too broad, so that overfitting is hopefully under control, is by allowing the predictor to be an additive combination of simple functions. We thus allow $f(x) = \beta_0 + \sum_{j=1}^{p} f_j(x_j)$. We also not assume the exact form of $\{f_j\}_{j=1}^{p}$ but rather learn them from the data, while constraining them to take some simple form. The ERM problem of GAMs is thus

$$argmin_{\beta_0, f \in \mathcal{F}} \left\{ \frac{1}{n} \sum_i (y_i - f(x_i))^2 \right\} \tag{16}$$

where f is as defined above.

Remark 9 (Not a pure ERM)**.** The learning of $\{f_j\}_{j=1}^{p}$ is in fact not performed by optimization, but rather by kernel smoothing (§ 3.2). The solution to Eq.(16) is not a pure ERM problem, but a hybrid between ERM and kernel smoothing.

2.9 Projection Pursuit Regression (PPR)

Another way to generalize the hypothesis class $\mathcal{F}$, which generalizes the GAM model, is to allow f to be some simple function of a linear combination of the predictors. Let

$$f(x) = \sum_{m=1}^{M} g_m(w_m x) \tag{17}$$

where both $\{g_m\}_{m=1}^{M}$ and $\{g_m\}_{m=1}^{M}$ are learned from the data. The regularization is now performed by choosing M and the class of $\{g_m\}_{m=1}^{M}$. The ERM problem is the same as in Eq.(16), with the appropriate f.

Remark 10 (Not a pure ERM)**.** Just like the GAM problem, in the PPR problem $\{g_m\}_{m=1}^{M}$ are learned by kernel smoothing. Solving the PPR problem is thus a hybrid of ERM and Kernel smoothing.

Remark 11 (Universal Approximator)**.** By choosing a sufficiently large M, the class $\mathcal{F}$ can approximate any continuous function. This property of the class is called a *Universal Approximator*.

Universal Approximator

2.10 Neural Networks (NNETs)

2.11 Single Hidden Layer

In the spirit of [Hastie et al., 2003, Section 11], we introduce the NNET model via the PPR model, and not through its historically original construction. In the language of Eq.(17), a single-layer–feed-forward neural network, is a model where $\{g_m\}_{m=1}^M$ are not learned from the data, but rather assumed a-priori.

$$g_m(xw_m) := \beta_m \sigma(\alpha_0 + x\alpha_m)$$

where only $\{w_m\}_{m=1}^M = \{\beta_m, \alpha_m\}_{m=1}^M$ are learned from the data. A typical *activation function*, $\sigma(t)$ is the standard logistic CDF: $\sigma(t) = \frac{1}{1+e^{-t}}$. Another popular alternative are Gaussian radial functions.

Activation Function

As can be seen, the NNET is merely a non-linear regression model. The parameters of which are often called *weights*.

NNETs have gained tremendous popularity, as it strikes a good balance between model complexity and regularity; particularly in the machine vision, sound analysis and natural language processing domains, where data samples are abundant.

Loss Functions For regression the squared loss is used. For classification, one can still use the squared error (as in the Regression Classifier), or the binomial likelihood leading to what is know as the *deviance*, or *cross-entropy* loss.

Deviance & Cross Entropy

Universal Approximator Like the PPR, even when $\{g_m\}_{m=1}^M$ are fixed beforehand, the class is still a universal approximator (although it might require a larger M than PPR).

Regularization Regularization of the model is done via the selection of the σ, the number of nodes/variables in the network and the number of layers. More that one hidden layer leads to *Deep Neural Networks* which offer even more flexibility at the cost of complexity, thus requiring many data samples for fitting.

Deep Neural Networks

Back Propagation The fitting of such models is done via a coordinate-wise gradient descent algorithm called *back propagation*.

Further Reading For more on NNETs see [Hastie et al., 2003, Chapter 11]. For a recent overview of Deep Learning in Neural Networks see Schmidhuber [2015].

2.12 Classification and Regression Trees (CARTs)

CARTs are a type of ERM where $f(x)$ include very non smooth functions that can be interpreted as "if-then" rules, also know as *decision trees*.

Decision Tree

The hypothesis class of CARTs includes functions of the form

$$f(x) = \sum_{m=1}^{M} c_m I_{x \in R_m} \tag{18}$$

where I_A is the indicator function of the event A. The parameters of the model are the different conditions $\{R_m\}_{m=1}^M$ and the function's value at each condition $\{c_m\}_{m=1}^M$.

Regularization is done by the choice of M which is called the *tree depth*.

Tree Depth

As $f(x)$ is defined over indicator functions, CARTs have no difficulty to deal with categorical variables and even missing values, on top of the more standard continuous predictors. More on the many advantages of CARTs can be found in the references.

Optimization As searching over all possible partitions of the x's to optimize $\{R_m\}_{m=1}^M$ is computationally impossible, optimization is done is a greedy fashion, by splitting on each variable x_j at a time.

Loss Functions As usual, a squared loss can be used for continuous outcomes y. For categorical outcomes, the loss function is called the *impurity measure*. One can use either a misclassification error, the multinomial likelihood (knows as the deviance, or cross-entropy), or a first order approximation of the latter known as the *Gini Index*.

Impurity Measure

Dendogram As CARTs are essentially decision trees, they can typically viewed as a hirarchy of if-then rules applied on the data. This type of visualization if called a *dendogram*.

Dendogram

A Personal Note While CARTs are very rich hypothesis classes, and easily interpretable, I have poor personal experience with them. I suspect it is their very non smooth nature that is simply inadequate for the problems I have encountered. Then again, the Bagging algorithm, deals with this matter nicely by averaging many trees.

Bagging

Further Reading For more on CARTs and Bagging see [Hastie et al., 2003, Section 9].

2.13 Smoothing Splines

[TODO]

3 Non ERM Methods

Up until now, we have focused on purely ERM, or hybrid ERM methods. Inductive bias, however, can also be introduced by avoiding the optimization approach of ERM. ERM decouples the motivation of a method and the particular learning algorithm (ultimately, some optimization algorithm). In the following methods, the learning algorithm is an integral part of the method. This restricts the learnable hypothesis class, thus acting as a regularizer.

Note that in most of the previous sections[2], the restriction of the hypothesis class $\mathcal{F}$ has been imposed by some parametric representation of $\mathcal{F}$, over which optimization is performed. The methods in this section do not impose any such parametrization. They are thus also known as *non parametric*.

Non Parametric

3.1 k-Nearest Neighbour (KNN)

The fundamental idea behind the KNN approach is that an observation is similar in y to its surroundings in x. So say I want to classify the type of a bird (y), I merely need to look at the classes of birds which are similar in their attributes (x's). Clearly this requires some notion of distance in the feature space $\mathcal{X}$, as typically all non-parametric methods do[3].

3.2 Kernel Smoothing

Not to be confused with the Kernel Trick in Kernel SVM, Kernel PCA, and Kernel Regression. Kernel *smoothing* is essentially a generalization of a moving average. The *Nadaraya-Watson* weighted average is defined as

Nadaraya-Watson

$$\hat{f}(x) := \frac{\sum_i \mathcal{K}_\lambda(x, x_i) y_i}{\sum_i \mathcal{K}_\lambda(x, x_i)} \tag{19}$$

[2]Well, bar CARTs and Smoothing Splines

[3]Why is this the case? Well, because non parametric methods replace the idea of optimization in a parameter space, with the idea of similarity in neighbourhoods.

where $\mathcal{K}_\lambda$ is the kernel function, which is merely a measure of the distance between the evaluation point x and the data points $\{x_i\}_{i=1}^n$ and weights the contribution of each $\{y_i\}_{i=1}^n$ to the value at x. The denominator merely ensures the weights sum to 1. Regularization is controlled by the λ parameter.

The moving average in a window of width λ is an instance where $\mathcal{K}_\lambda(x, x_i)$ is fixed in the window.

Metric Spaces Clearly, as $\mathcal{K}_\lambda(x, x_i)$ measures distances, it only makes sense if there indeed exists a notion of distance between the x's, which is saying that the feature space $\mathcal{X}$ is a metric space. This can be far from trivial when dealing with categorical predictors such as countries, genomes, gender, etc. Have no worries however, as even with these type of variables, one can define some notion of distance (not claiming it will prove useful in application).

Relation to KNN There is an intimate relation between KNN and Kernel Smoothing. Essentially, both predict the value of y at some x by averaging their neighbourhood. Averaging can be weighted or not, in both cases. The important distinction is how neighbourhoods are defined. In KNN, the neighbourhood of x is the k nearest data points. The radius of the neighbourhood thus varies, while the number of data points does not. In Kernel Smoothing, the neighbourhood of x is all the data points in the support of the kernel. The radius of the neighbourhood is now fixed, while the number of data points can vary.

3.3 Local Regression (LOESS)

[TODO]

3.4 Local Likelihood and Local ERM

Although presented as two competing concepts, they can augment each other. We have already seen that Kernel Smoothing plays a part within some ERM algorithms such as GAM (§2.8) and PPR (§2.9).

Another way to marry the two ideas, is by performing EMR only locally, in each neighbourhood of data points defined by a kernel. This idea is very powerful and leads to, e.g., the LOESS (§3.3) and *Local Likelihood* methods.

4 Statistical Decision Theory

[TODO]

5 Unsupervised Learning

[TODO]

6 Dimensionality Reduction

[TODO]

6.1 PCA

[TODO]

7 Latent Space Models

[TODO]

References

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, July 2003. ISBN 0387952845.

G. James, D. Witten, T. Hastie, and R. Tibshirani. *An Introduction to Statistical Learning: with Applications in R*. Springer, New York, 1st ed. 2013. corr. 4th printing 2014 edition edition, Aug. 2013. ISBN 9781461471370.

P. McCullagh and J. A. Nelder. *Generalized Linear Models, Second Edition*. Chapman and Hall/CRC, Boca Raton, 2 edition edition, Aug. 1989. ISBN 9780412317606.

J. Schmidhuber. Deep Learning in Neural Networks: An Overview. *Neural Networks*, 61:85–117, Jan. 2015. ISSN 08936080. doi: 10.1016/j.neunet.2014.09.003. URL `http://arxiv.org/abs/1404.7828`. arXiv: 1404.7828.

S. Shalev-Shwartz and S. Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, New York, NY, May 2014. ISBN 9781107057135.

V. N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, New York, 1 edition edition, Sept. 1998. ISBN 9780471030034.

L. Wasserman. *All of statistics: a concise course in statistical inference*. Springer, New York, 2004. ISBN 0387402721 9780387402727.