

# Machine Learning Cheat Sheet

## General

### Dataset and Definitions

$\mathcal{D}$  is a set of training examples, the  $n$ -th Training Example ( $n = 1, 2, \dots, N$ ), of this set is:

$$\mathbf{x}_n = [x_{n1} \quad x_{n2} \quad \dots \quad x_{nD}]$$

We write all  $N$  training examples in a matrix:

$$\mathbf{X} = [\mathbf{x}_1; \quad \mathbf{x}_2; \quad \dots; \quad \mathbf{x}_N]$$

In supervised learning, you are also given a set of observations corresponding to the training examples:

$$\mathbf{y} = [y_1 \quad y_2 \quad \dots \quad y_N]^T$$

We assume a *true* model behind the data, the observations are noisy versions of this ground truth:  $y = y_{true} + \epsilon$ .

The final goal is to predict a  $\hat{y} = f(\hat{\mathbf{x}})$  given any  $\hat{\mathbf{x}}$ .

### Distributions

Multivariate gaussian distribution:  $\mathcal{N}(\mathbf{X}|\mu, \Sigma)$

$$\Rightarrow p(\mathbf{X} = \mathbf{x}) = (2\pi)^{-d/2} |\Sigma|^{-1/2} \exp[-\frac{1}{2}(\mathbf{x} - \mu)^T \Sigma^{-1}(\mathbf{x} - \mu)]$$

$$\text{Gaussian distribution: } \mathcal{N}(X|\mu, \sigma^2) \Rightarrow p(X = x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{1}{2}(\frac{x-\mu}{\sigma})^2)$$

Poisson distribution:  $\mathcal{P}(X|\lambda)$

$$\Rightarrow p(X = k) = \frac{\lambda^k}{k!} \exp(-\lambda)$$

### Properties

**Bayes rule:**  $p(A, B) = p(A|B)p(B) = p(B|A)p(A)$

A matrix  $\mathbf{M}$  is **positive semidefinite**  $\iff \forall$  nonzero vector  $\mathbf{a}$ , we have  $\mathbf{a}^T \mathbf{M} \mathbf{a} \geq 0$

**Jensen's inequality** applied to log:

$$\log(\mathbb{E}[X]) \geq \mathbb{E}[\log(X)] \Rightarrow \log(\sum_x x \cdot p(x)) \geq \sum_x p(x) \log(x)$$

**Matrix inversion lemma:**

$$(\mathbf{P}\mathbf{Q} + \mathbf{I}_N)^{-1} \mathbf{P} = \mathbf{P}(\mathbf{Q}\mathbf{P} + \mathbf{I}_M)^{-1}$$

Useful derivative:  $\frac{\partial \mathbf{x}^T \mathbf{B} \mathbf{x}}{\partial \mathbf{x}} = (\mathbf{B} + \mathbf{B}^T) \mathbf{x}$

Marginal and Conditional Gaussians:

$$p(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\mu, \Lambda^{-1})$$

$$p(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\mathbf{y}|\mathbf{A}\mathbf{x} + \mathbf{b}, \mathbf{L}^{-1})$$

$\Downarrow$

$$p(\mathbf{y}) = \mathcal{N}(\mathbf{y}|\mathbf{A}\mu + \mathbf{b}, \mathbf{L}^{-1} + \mathbf{A}\Lambda^{-1}\mathbf{A}^T)$$

$$p(\mathbf{x}|\mathbf{y}) = \mathcal{N}(\mathbf{x}|\Sigma\{\mathbf{L}(\mathbf{y} - \mathbf{b}) + \Lambda\mu\}, \Sigma)$$

$$\text{where } \Sigma = (\Lambda + \mathbf{A}^T \mathbf{L} \mathbf{A})^{-1}$$

## Optimization methods

### Grid Search

Simply try values for all parameters at regular intervals. Complexity:  $\mathcal{O}(M^D ND)$ , where  $M$  is the number of values tried in each dimension.

## Gradient Descent

Update rule:  $\beta^{(k+1)} = \beta^{(k)} - \alpha \frac{\partial \mathcal{L}(\beta^{(k)})}{\partial \beta}$

Complexity:  $\mathcal{O}(IND)$  where  $I$  is the number of iterations we take.

$\alpha$  is a parameter that needs to be chosen carefully.

The gradient for MSE comes out as:

$$\frac{\partial \mathcal{L}}{\partial \beta} = -\frac{1}{N} \tilde{\mathbf{X}}^T (\mathbf{y} - \tilde{\mathbf{X}}\beta)$$

### Newton's method

It uses second order derivatives information to converge faster (we approximate the objective function by a quadratic function rather than a line).

General rule:  $\beta^{(k+1)} = \beta^{(k)} - \alpha \mathbf{H}_k^{-1} \frac{\partial \mathcal{L}(\beta^{(k)})}{\partial \beta}$

where  $\mathbf{H}_k$  is the  $D \times D$  Hessian at step  $k$ :

$$\mathbf{H}_k = \frac{\partial^2 \mathcal{L}(\beta^{(k)})}{\partial \beta^2}$$

Newton's method is equivalent to solving many least-squares problems.

Complexity:  $\mathcal{O}(IND^2 + ID^3)$ , with the  $D^3$  cost coming from the inversion of  $\mathbf{H}_k$ .

### Expectation-Maximization

EM is a family of methods to find a MLE/MAP estimator if some of the data is missing (e.g. latent variables):

Given is some data  $\mathbf{x}$  and a model with some latent variables  $\mathbf{z}$  and a likelihood  $\mathcal{L}(\theta; \mathbf{x}, \mathbf{z})$  (e.g. a gaussian where we need to find the mean and the covariance). Goal is to find  $\beta = \arg \max_{\beta} \mathcal{L}_{lik}(\theta; \mathbf{x})$ , i.e. the MLE given only the data, without knowing the latent variables.

The problem is, that this likelihood can oftentimes not be optimized in closed form with respect to  $\theta$ . The solution is EM: Iteratively find better  $\theta$ , start with  $\theta_0$  and iterate with variable  $t$ :

- E-step calculates the  $\mathbb{E}$  of log likelihood, with respect to  $\mathbf{Z}$  given  $\mathbf{X}$  under the current estimate  $\theta_t$ :  $Q(\theta, \theta_t) = \mathbb{E}_{\mathbf{Z}|\mathbf{X}, \theta_t} [\log(p(\mathbf{X}, \mathbf{Z}|\theta)) | \mathbf{X} = \mathbf{x}]$
- M-step finds next estimate that maximizes:  $\theta_{t+1} = \arg \max_{\theta} Q(\theta, \theta_t)$
- Iterate until convergence.

## Regression

Simple linear regression:  $y_n \approx \beta_0 + \beta_1 x_{n1}$

Multiple linear regression:

$$y_n \approx f(\mathbf{x}_n) := \beta_0 + \beta_1 x_{n1} + \beta_2 x_{n2} + \dots + \beta_D x_{nD}$$

### Linear basis function model

We can create more complex models while staying in the linear framework by transforming the inputs  $X$  of dimensionality  $D$  through a function  $\phi: D \rightarrow M$ .  $y_n = \beta_0 + \sum_{i=1}^M \beta_i \phi_i(\mathbf{x}_n) = \tilde{\phi}^T(\mathbf{x}_n^T) \beta$ . The optimal  $\beta$  can be computed in closed form by  $\beta = (\tilde{\Phi}^T \tilde{\Phi})^{-1} \tilde{\Phi}^T \mathbf{y}$  where  $\tilde{\Phi}$  is a matrix with  $N$  rows and the  $n$ -th row is  $[1, \phi_1(\mathbf{x}_n), \dots, \phi_M(\mathbf{x}_n)]$ .

But note this requires  $\tilde{\Phi}^T \tilde{\Phi}$  to be invertible (well-conditioned:  $\tilde{\Phi}$  full column-rank).

Ridge regression:  $\beta_{ridge} = (\tilde{\Phi}^T \tilde{\Phi} + \lambda \mathbf{I})^{-1} \tilde{\Phi}^T \mathbf{y}$

### Cost functions

Cost function / Loss:  $\mathcal{L}(\beta) = \mathcal{L}(\mathcal{D}, \beta)$

Mean square error (MSE):  $\frac{1}{2N} \sum_{n=1}^N [y_n - f(\mathbf{x}_i)]^2$

MSE matrix formulation:  $\frac{1}{2N} (\mathbf{y} - \mathbf{X}\beta)^T (\mathbf{y} - \mathbf{X}\beta)$

Mean absolute error (MAE):  $\frac{1}{2N} \sum_{n=1}^N |y_n - f(\mathbf{x}_i)|$

Huber loss:  $\mathcal{L}_{\delta}(a) = \begin{cases} \frac{1}{2}a^2 & \text{for } |a| \leq \delta, \\ \delta(|a| - \frac{1}{2}\delta), & \text{otherwise.} \end{cases}$

Root mean square error (RMSE):  $\sqrt{2 * \text{MSE}}$

Epsilon insensitive ("hinge loss", used for SVM):

$$\mathcal{L}_{\epsilon}(y, \hat{y}) = \begin{cases} 0 & \text{if } |y - \hat{y}| \leq \epsilon, \\ |y - \hat{y}| - \epsilon, & \text{otherwise.} \end{cases}$$

## Least squares

Complexity:  $\mathcal{O}(ND^2 + D^3)$  The gradient of the MSE set to zero is called the normal equation:

$$-\mathbf{y}^T \mathbf{X} + \mathbf{X}^T \mathbf{X} \beta = 0$$

We can solve this for  $\beta$  directly (by matrix manipulation)  $\beta = (\tilde{\mathbf{X}}^T \tilde{\mathbf{X}})^{-1} \tilde{\mathbf{X}}^T \mathbf{y}$

## Classification

Logistic Function:  $\sigma(t) = \frac{\exp(t)}{1 + \exp(t)}$

Derivative:  $\frac{\partial \sigma(t)}{\partial t} = \sigma(t)[1 - \sigma(t)]$

Classification with linear regression: Use  $y = 0$  as class  $\mathcal{C}_1$  and  $y = 1$  as class  $\mathcal{C}_2$  and then decide a newly estimated  $y$  belongs to  $\mathcal{C}_1$  if  $y < 0.5$ .

### Logistic Regression

$$\frac{\partial \mathcal{L}(\beta)}{\partial \beta} = \tilde{\mathbf{X}}^T [\sigma(\tilde{\mathbf{X}}\beta) - \mathbf{y}]$$

There's no closed form, we can use gradient descent.

### Generalized linear model

The GLM consists of three elements:

- A probability distribution from the exponential family.
- A linear predictor  $\hat{y} = \mathbf{X}\beta$ .
- A link function  $g$  such that  $E(y) = \mu = g^{-1}(\eta)$ .

In a generalized linear model (GLM), each outcome of the dependent variables  $y$  is assumed to be generated from a particular distribution in the exponential family, a large range of probability distributions that includes the normal, binomial, Poisson and gamma distributions, among others.

### Cost functions

$$\text{RMSE: } \sqrt{\frac{1}{N} \sum_{n=1}^N [y_n - \hat{p}_n]^2}$$

$$0-1 \text{ Loss: } \frac{1}{N} \sum_{n=1}^N \delta(y_n, \hat{y}_n)$$

Log-Loss:

$$-\frac{1}{N} \sum_{n=1}^N y_n \log(\hat{p}_n) + (1 - y_n) \log(1 - \hat{p}_n)$$

## Probabilistic framework

The Likelihood Function maps the model parameters to the probability distribution of  $\mathbf{y}$ :

$\mathcal{L}_{lik}: \text{parameter space} \rightarrow [0; 1] \quad \beta \mapsto p(\mathbf{y} | \beta)$  An underlying  $p$  is assumed before. If the observed  $y$  are IID,  $p(\mathbf{y} | \beta) = \prod_n p(y_n | \beta)$ .

$\mathcal{L}_{lik}$  can be viewed as just another cost function.

Maximum likelihood then simply chooses the parameters  $\beta$  such that observed data is most likely.

$$\beta = \arg \max_{\beta} \mathcal{L}(\beta)$$

Assuming different  $p$  is basically what makes this so flexible. We can choose e.g.:

$$\begin{array}{ll} \text{Gaussian } p & \mathcal{L}_{lik} \hat{=} -\mathcal{L}_{MSE} \\ \text{Poisson } p & \mathcal{L}_{lik} \hat{=} -\mathcal{L}_{MAE} \end{array}$$

It is a sample approximation of the expected likelihood:  $\mathcal{L}_{lik}(\beta) \approx E_y[p(y | \beta)]$

## Bayesian methods

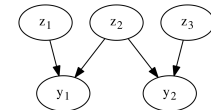
Bayes rule:  $p(A, B) = p(A|B)p(B) = p(B|A)p(A)$

The **prior**  $p(\mathbf{f}|\mathbf{X})$  encodes our prior belief about the "true" model  $\mathbf{f}$ . The **likelihood**  $p(\mathbf{y}|\mathbf{f})$  measures the probability of our (possibly noisy) observations given the prior.

Least-squares tries to find model parameters  $\beta$  which maximize the likelihood. Ridge regression maximizes the **posterior**  $p(\beta|\mathbf{y})$

## Bayesian networks

We can use a Directed Acyclic Graph (DAG) to define a joint distribution of events. For example, we can express the relationship between *latent factors* (possible "causes")  $z_i$  and *observations* (results)  $y_i$ :



This example can be factorized as follows:

$$p(y_1, y_2, z_1, z_2, z_3) =$$

$$p(y_1|z_1, z_2)p(y_2|z_2, z_3)p(z_1)p(z_2)p(z_3)$$

We can then obtain the distribution over latent factors ( $z_i$ ) by marginalizing over the unknown variables:

$$p(z_1, z_2, z_3|y_1, y_2) = \frac{\text{joint}}{p(y_1, y_2)} \Rightarrow p(z_1|y_1, y_2) = \sum_{z_2, z_3} \frac{\text{joint}}{p(y_1, y_2)}$$

## Belief propagation

Belief propagation is a message-passing based algorithm used to compute desired marginals (e.g.  $p(z_1|y_1, y_2)$ ) efficiently. It leverages the factorized expression of the joint. Messages passed:

$$m_{z_i \rightarrow y_j} =$$

$$p(z_i) \prod (\text{messages received except from } y_j)$$

$$m_{y_j \rightarrow z_i} =$$

$$\sum_{z_k \neq z_i} p(y_j|z_k) \prod (\text{messages received except from } z_i)$$



## Occam's Razor

It states that among competing hypotheses, the one with the fewest assumptions should be selected.

Other, more complicated solutions may ultimately prove correct, but in the absence of certainty, the fewer assumptions that are made, the better.

TODO: K-fold cross-validation, definition of Test-Error, Train-Error

TODO: statistical goodness (robust to outliers, ...) vs. computational goodness (convex, low computational complexity, ...) tradeoff. No free lunch theorem.

TODO: Decision Trees and Random Forests and Model averaging

## Credits

Most material was taken from the lecture notes of Prof. Emtiyaz Khan.

Cost functions figure from Patrick Breheny's slides.

Biais-variance decomposition figure from Hastie, Tibshirani, Friedman, *The Elements of Statistical Learning*. The SVD figure from Kevin P. Murphy,

*Machine Learning, A Probabilistic Perspective*.

Rendered January 11, 2015. Written by Dennis Meier and Merlin Nimier-David.

© Dennis Meier. This work is licensed under the Creative Commons

Attribution-ShareAlike 3.0 Unported License. To view a copy of this license, visit

<http://creativecommons.org/licenses/by-sa/3.0/> or send a letter to Creative

Commons, 444 Castro Street, Suite 900, Mountain View, California, 94041, USA.

