# Machine Learning Cheat Sheet

## Regression

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

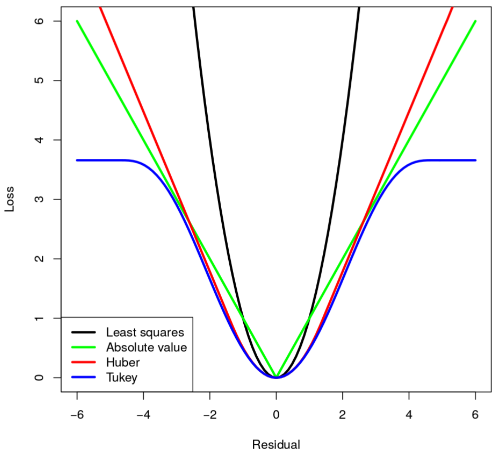
$$\mathbf{x}_n = \begin{bmatrix} x_{n1} \\ x_{n2} \\ \dots \\ x_{nD} \end{bmatrix}$$

The goal is to predict a $\hat{y}$ given a $\mathbf{x}$.
Simple linear regression: $y_n \approx \beta_0 + \beta_1 x_{n1}$
Multiple linear regression:
$y_n \approx f(\mathbf{x}_n) := \beta_0 + \beta_1 x_{n1} + \beta_2 x_{n2} + ... + \beta_D x_{nD}$

### Linear basis function model

$y_n = \beta_0 + \sum_{i=1}^{M} \beta_i \phi_i(\mathbf{x_n}) = \tilde{\phi}^{\mathbf{T}}(\mathbf{x}_n^T)\boldsymbol{\beta}$. The optimal $\beta$ is given by $\beta = (\tilde{\Phi}^T\tilde{\Phi})^{-1}\tilde{\Phi}^T y$ where $\tilde{\Phi}$ is a matrix with N rows and the n-th row is $[1, \phi_1(x_n)^T, ..., \phi_M(x_n)^T]$.
Ridge regression: $\beta_{ridge} = (\tilde{\boldsymbol{\Phi}}^T\tilde{\boldsymbol{\Phi}} + \lambda \boldsymbol{I})^{-1}\tilde{\boldsymbol{\Phi}}^T\boldsymbol{y}$

### Cost functions



Cost function / Loss: $\mathcal{L}(\boldsymbol{\beta}) = \mathcal{L}(\mathcal{D}, \boldsymbol{\beta})$
Mean square error (MSE): $\frac{1}{2N}\sum_{n=1}^{N}[y_n - f(\mathbf{x}_i)]^2$
Mean absolute error (MAE): $\frac{1}{2N}\sum_{n=1}^{N}|y_n - f(\mathbf{x}_i)|$

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

Root mean square error (RMSE): $\sqrt{2 * \text{MSE}}$
Epsilon insensitive (used for SVMs):

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

## Grid Search

Complexity: $\mathcal{O}(M^D N D)$, where $M$ is the number of grids in one dimension.

## Gradient Descent

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

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

Big questions are how to get a good $\alpha$.

The gradient for MSE comes out as:
$\frac{\partial\mathcal{L}}{\partial\boldsymbol{\beta}} = -\frac{1}{N}\tilde{X}^T(\boldsymbol{y} - \tilde{X}\boldsymbol{\beta})$

## Least squares

Complexity: $\mathcal{O}(ND^2 + D^3)$

$\beta = (\tilde{X}^T\tilde{X})^{-1}\tilde{X}^T y$

## Classification

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

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

### Logistic Regression

$\tilde{\mathbf{X}}^T[\sigma(\tilde{\mathbf{X}}\beta) - y] = 0$

TODO: Generalized Linear model

### Error Functions

Root Mean square error (RMSE):
$\sqrt{\frac{1}{N}\sum_{n=1}^{N}[y_n - \hat{p_n}]^2}$

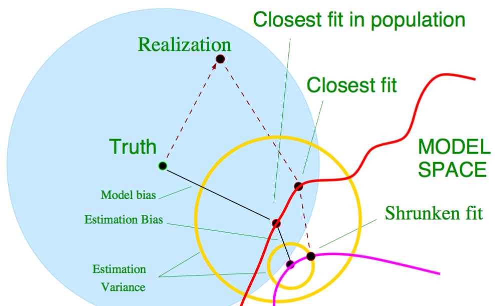
0-1 Loss: $\frac{1}{N}\sum_{n=1}^{N}\delta(y_n, \hat{y_n})$

logLoss:
$-\frac{1}{N}\sum_{n=1}^{N} y_n \log(\hat{p_n}) + (1 - y_n)\log(1 - \hat{p_n})$

## 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, butin the absence of certaintythe fewer assumptions that are made, the better.

## Bias-Variance Decomposition



| | bias | variance |
|---|---|---|
| regularization | + | - |
| reduce model complexity | + | - |
| more data | - | |

## Maximum Likelihood

The Likelihood Function maps the model parameters to the probability distribution of $\mathbf{y}$:
$\mathcal{L}_{lik}$: parameter space $\rightarrow [0; 1] \quad \beta \mapsto p(\mathbf{y} \mid \beta)$ An underlying $p$ is assumed before. If the observed $y$ are IID, $p(\mathbf{y} \mid \beta) = \prod_n p(y_n \mid \beta)$.
$\mathcal{L}_{lik}$ can be viewed as just another cost function. Maximum likelihood then simply choses the parameters $\beta$ such that observed data is most likely.
$\beta = \arg\max_{\text{all}\,\beta} L(\beta)$
Assuming different $p$ is basically what makes this so flexible. We can chose e.g.:

| | |
|---|---|
| Gaussian $p$ | $\mathcal{L}_{lik}\widehat{=}\mathcal{L}_{MSE}$ |
| Poisson $p$ | $\mathcal{L}_{lik}\widehat{=}\mathcal{L}_{MAE}$ |

## 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** $p(\beta|\mathbf{y})$

## Kernel

Basically, Kernels are a mean to measure distance, or "similarity" of two vectors. We define:
$(\mathbf{K})_{i,j} = \kappa(\mathbf{x_i}, \mathbf{x_j}) = \vec{\phi}(\mathbf{x_i})^T\vec{\phi}(\mathbf{x_j})$.
The $\phi$ are not that important in the end, because we only use the Kernel as is. Sometimes it's even impossible to write them down explicitly.

| | |
|---|---|
| Linear | $\kappa(\mathbf{x_i}, \mathbf{x_j}) = \mathbf{x_i}^T\mathbf{x_j}$ |
| Polynomial | $\kappa(\mathbf{x_i}, \mathbf{x_j}) = (\mathbf{x_i}^T\mathbf{x_j} + c)^d$ |
| RBF | $\kappa(\mathbf{x_i}, \mathbf{x_j}) = \exp\left(-\frac{\lvert\lvert\mathbf{x_i} - \mathbf{x_j}\rvert\rvert^2}{2\sigma^2}\right)$ |

## Support Vector Machines

Search for the hyperplane separating the data such that the gap is biggest. It minimizes the following cost function:
$\mathcal{L}_{SVM}(\beta) = \sum_{n=1}^{N}[1 - y_n\tilde{\phi}_n\beta]_+ + \frac{\lambda}{2}\sum_{j=1}^{M}\beta_j^2$
This is convex but not differentiable.

## Graphical Models

Bayes Net: Directed acyclic graph
Belief propagation: graph between the observations and the variables is a bi-partite graph.