CS229-Cheatsheet

Supervised Learning

- **Gradient Descent:** to minimize $J(\theta)$, we perform $\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta} J(\theta)$
- $\nabla_A AB = B^T$, $\nabla_{A^T} f(A) = (\nabla_A f(A))^T$, $\nabla_A trABA^TC = CAB + C^TAB^T$, $\nabla_A |A| = |A|(A^{-1})^T$
- **Normal Equations and Least Squares** $J(\theta) = \frac{1}{2}\sum_{i=1}^{m}(h_\theta(x^i) - y^i)^2 \rightarrow \nabla_\theta J(\theta) =$ $\nabla_\theta \frac{1}{2}(X\theta - y)^T(X\theta - y) = X^TX\theta - X^Ty = 0 \rightarrow$ $X^TX\theta = X^Ty \rightarrow \theta = (X^TX)^{-1}X^Ty.$
- **Locally Weighted Regression** Fit θ to minimize $\sum_{i=0}^{m}(y^i - \theta^Tx^i)^2$ where $w^i = e^{-\frac{(x^i-x)^2}{2\tau^2}}$
- **Logistic Regression:** $h_\theta(x) = g(\theta^Tx) = \frac{1}{1+e^{-\theta^Tx}}$, $g(z) = \frac{1}{1+e^{-z}}$, $g'(z) = \frac{d}{dz}\frac{1}{1+e^{-z}} = g(z)(1-g(z))$, $p(y|x;\theta) = (h_\theta(x))^y(1-h_\theta(x))^{1-y}$. $l(\theta) = logL(\theta) = \sum_{i=1}^{m} y^i logh(x^i) + (1-y^i)log(1-h(x^i))$, $\frac{\partial}{\partial\theta_j}l(\theta) = (y - h_\theta(x))x_j$
- **Perceptron Learning Algorithm** $\theta_j := \theta_j + \alpha(y^i - h_\theta(x^i))x_j^i$
- **Newton's Method:** $\theta := \theta - \frac{f(\theta)}{f'(\theta)}$, we want the first derivative to be zero, then $\theta := \theta - \frac{l'(\theta)}{l''(\theta)}$, if θ is a vector then $\theta := \theta - H^{-1}\nabla_\theta l(\theta)$ where $H_{ij} = \frac{\partial^2 l(\theta}{\partial\theta_i\partial\theta_j}$
- **Exponential Family** $p(y;\eta) = b(y)exp(\eta^TT(y) - a(\eta))$
- **General Linear Model Assumptions:** 1. $y|x;\eta \sim ExponentialFamily(\eta)$. 2. Given x our goal is to predict the expected value of $T(y)$ which is usually just y, so we would like our hypothesis to satisfy $h(x) = E(y|x)$. 3. The natural parameter η and inputs x are related linearly. $\eta = \theta^Tx$.
- **Canonical response function:** the distribution's mean as a function of the natural parameter $g(\eta) = E(T(y);\eta)$.

Generative Learning Algorithm

Gaussian Discriminant Analysis

- $y \sim Bernoulli(\phi), x|y=0 \sim N(\mu_0, \Sigma), x|y=1 \sim N(\mu_1, \Sigma)$.
- $p(y) = \phi^y(1-\phi)^{1-y}$
- $p(x|y=0) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}}exp(-\frac{1}{2}(x-\mu_0)^T\Sigma^{-1}(x-\mu_0))$
- $p(x|y=1) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}}exp(-\frac{1}{2}(x-\mu_1)^T\Sigma^{-1}(x-\mu_1))$
- $l(\phi,\mu_0,\mu_1,\Sigma) = log\prod_{i=1}^{m}p(x^i,y^i;\phi,\mu_0,\mu_1,\Sigma) =$ $log\prod_{i=1}^{m}p(x^i|y^i;\phi,\mu_0,\mu_1,\Sigma)p(y^i,\phi)$.
- By maximizing l with respect to the parameters, we find the maximum likelihood of the parameters to be: $\phi = \frac{1}{m}1\{y^i = 1\}$ $\mu_0 = \frac{\sum_{i=1}^m\{y^i=0\}x^i}{\sum_{i=1}^m\{y^i=0\}}$ $\mu_1 = \frac{\sum_{i=1}^m\{y^i=1\}x^i}{\sum_{i=1}^m\{y^i=1\}}$ $\Sigma = \frac{1}{m}\sum_{i=1}^{m}(x^i-\mu_{y^i})^T(x^i-\mu_{y^i})$

Naive Bayes

- **Naive Assumption:** $p(x_1,x_2,\ldots|y) = p(x_1|y)p(x_2|y)\ldots. = \prod_{i=1}^{n}p(x_i|y)$
- **Laplace Smoothing** $\phi_j = \frac{\sum_{i=1}^m 1\{z^i=j\}}{m} \rightarrow \frac{\sum_{i=1}^m 1\{z^i=j+1\}}{m+k}$, where k represent the number of possible outcomes for z.
- **Event Driven Text Classification:** $\phi_{k|y=1} = \frac{\sum_{i=1}^m\sum_{j=1}^{n_i}1\{x_j^i = k \wedge y^i = 1\}}{\sum_{i=1}^m 1\{y^i=1\}n_i}$ $\phi_{k|y=0} = \frac{\sum_{i=1}^m\sum_{j=1}^{n_i}1\{x_j^i = k \wedge y^i = 0\}}{\sum_{i=1}^m 1\{y^i=0\}n_i}$ $\phi_y = \frac{\sum_{i=1}^m 1\{y^i=1\}}{m}$

Support Vector Machines

- **Classifier**: $h_{w,b}(x) = g(w^Tx+b)$ where $g(z) = 1$ if $z > 0$ and $g(z) = -1$ otherwise.
- **Functional Margins:** $\hat{\gamma}^i = y^i(w^Tx+b)$, the smallest functional margin in the training set is called: $\hat{\gamma} = min_{i=1,2,\ldots,m}\hat{\gamma}^i$
- **Geometric Margins:** $\gamma^i = y^i((\frac{w}{||w||})^Tx^i + \frac{b}{||w||})$, the smallest geometric margin in a training set is :$\gamma = min_{i=1,\ldots,m}\gamma^i$
- **Optimal Margin Classifier**: $min_{\gamma,w,b}\frac{1}{2}||w||^2$ s.t $y^i(w^Tx^i+b) \geq 1, i = 1,2,\ldots,m$
- **Lagrangian** $L(w,b,\alpha) = \frac{1}{2}||w||^2 - \sum_{i=1}^{m}\alpha_i(y^i(w^Tx^i+b)-1)$
- **The dual problem** $max_\alpha W(\alpha) = \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{m}y^iy^j\alpha_i\alpha_j(x^i)^Tx^j$ s.t $\sum_{i=1}^{m}\alpha_iy^i = 0$ $\alpha_i \geq 0 fori = 1,\ldots,m$
- **Observations:** 1. Most of the α_is will be zero 2. $w^Tx+b = (\sum_{i=1}^{m}\alpha_iy^ix^i)^Tx+b$
- **KKT Conditions:**

$$\frac{\partial}{\partial w_i}L(w^*,\alpha^*,\beta^*) = 0, i = 1,\ldots n$$
$$\frac{\partial}{\partial \beta_i}L(w^*,\alpha^*,\beta^*) = 0, i = 1,\ldots l$$
$$\alpha^*g_i(w^*) = 0, i = 1,\ldots,k$$
$$g_i(w^*) \leq 0, i = 1,\ldots,k$$
$$\alpha^* \geq 0, i = 1,\ldots,k$$

- **Mercer Theorem:** Let $K: \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}$ be given, then for K to be a valid kernel, it is necessary and sufficient that for any $\{x_1,x_2,\ldots,x^m\}$, the corresponding kernel matrix is symmetric positive semi-definite.
- **Regularization (revised optimal margin classifier** $min_{\gamma,w,b}\frac{1}{2}||w||^2 + C\sum_{i=1}^{m}\xi_i$ s.t $y^i(w^Tx^i+b) \geq 1-\xi_i, i = 1,2,\ldots,m$ $\xi_i \geq 0, i = 1,\ldots,m$
- **Dual of Regularization** $max_\alpha W(\alpha) = \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{m}y^iy^j\alpha_i\alpha_j(x^i)^Tx^j$ s.t $\sum_{i=1}^{m}\alpha_iy^i = 0$ $C \geq \alpha_i \geq 0 fori = 1,\ldots,m$

Learning Theory

- **Union Bound:** Let $A_1,A_2,\ldots,A_k$ be k different events (that may not be independent). Then

$$P(A_1 \cup A_2 \ldots A_k) \leq P(A_1) + P(A_2) + \ldots P(A_k)$$

- **Hoeffding inequality** Let $Z_1,Z_2,\ldots Z_m$ be m independent and identically distributed random variables drawn from Bernoulli(ϕ) distribution. Let $\hat{\phi} = \frac{1}{m}\sum_{i=1}^{m}Z_i$ be the mean of these random variables and let any $\gamma > 0$ be fixed. Then

$$P(|\phi - \hat{\phi} > \gamma) \leq 2exp(-2\gamma m)$$

- **Generational Error Bound:** $P(|\epsilon(h_i) - \hat{\epsilon}(h_i)| > \gamma) \leq 2exp(-2\gamma^2 m)$
- **Uniform Convergence:**

$$\begin{aligned} &P(\neg\exists h \in H.|\epsilon(h_i) - \hat{\epsilon}(h_i)| > \gamma) \\ &= P(\forall h \in H.|\epsilon(h_i) - \hat{\epsilon}(h_i)| \leq \gamma) \\ &\geq 1 - 2kexp(-2\gamma^2 m) \end{aligned}$$

- **Solving** m, γ, δ We just need to use the equation $\delta = 2kexp(-2\gamma^2 m)$ to solve for one variable given the other two.
- Let $|H| = k$, and let any m, δ, be fixed. Then with probability at least $1 - \delta$, we have that

$$\epsilon(\hat{h}) \leq (\min_{h \in H} \epsilon(h)) + 2\sqrt{\frac{1}{2m} log(\frac{2k}{\delta})}$$

- Let H be given, and let $d = VC(H)$, then with probability at least $1 - \delta$, we have that for all $h \in H$,

$$|\epsilon(h) - \hat{\epsilon}(h)| \leq O(\sqrt{\frac{d}{m} log\frac{m}{d} + \frac{1}{m} log\frac{1}{d}})$$

we also have

$$|\hat{\epsilon}(h) - \epsilon(h^*)| \leq O(\sqrt{\frac{d}{m} log\frac{m}{d} + \frac{1}{m} log\frac{1}{d}})$$

Regularization and Model Selection

Cross Validation

- **Simple Cross Validation:** 1. Randomly split S into S_{train} (say 70 percent of the data and S_{cv}. Here, S_{cv} is called the hold-out cross validation set. 2. Train each model M_i on S_{train} only, to get some hypothesis h_i. 3. Select and output the hypothesis h_i that had the smallest error $\hat{\epsilon}(h_i)$ on the hold out crosss validation set.
- ***k*-fold cross validation** 1. Randomly split S into k disjoint subsets of m/k training examples each. Lets call these subsets $S_1, ..., S_k$. 3. For each model M_i, we evaluate it as follows: For $j = 1, ..., k$ Train the model M_i on $S_1 \cup \cup S_{j1} \cup S_{j+1} \cup S_k$(i.e., train on all the data except S_j) to get some hypothesis h_{ij}. Test the hypothesis h_{ij} on S_j , to get $\epsilon_{S_j}(h_{ij})$. The estimated generalization error of model M_i is then calculated as the average of the $\epsilon_{S_j}(h_{ij})$s (averaged over j).

Feature Selection

- **Forward Search:**1. Initialize$F = \emptyset$. 2. Repeat (a)For $i = 1, ..., n$if $i \notin F$,let $F_i = F \cup \{i\}$,and use some version of cross validation to evaluate features F_i. (I.e., train your learning algorithm using only the features in F_i, and estimate its generalization error.) (b) Set F to be the best feature subset found on step (a). 3. Select and output the best feature subset that was evaluated during the entire search procedure.

Bayesian Statistics

- $P(\theta|S) = \dfrac{P(S|\theta)p(\theta)}{p(S)} = \dfrac{(\prod_{i=1}^m p(y^i|x^i, \theta))p(\theta)}{\int_\theta (\prod_{i=1}^m p(y^i|x^i, \theta)p(\theta))d\theta}$
- $P(y|x, S) = \int_\theta p(y|x, \theta)p(\theta|S)d\theta$
- $E(y|x, S) = \int_y yp(y|x, S)dy$.
- $\theta_{MAP} = arg\max_\theta$