CS 480 Cheat Sheet

Perceptron

We assume $(x_1, y_1), ..., (x_n, y_n)$ belongs to some distribution.
Choose predictive function h_n such that $\max Pr(h(x_i) = y_i)$
Dot Product: $\langle w, x\rangle = \sum w_i x_i$
Padding: $\langle w, x\rangle + b = \langle (w, b), (x, 1)\rangle$
Note: $z = (w, b)$, $a_i = y_i(x_i, 1)$
Linear Seperable: if $s > 0$ and $Az > s1$
If data is not linearly seperable perceptron stalls.
Margin is determined by closest point to the hyperplane.
Perceptron finds a solution, no guarantee its the best solution.
l2 Norm: $||x||_2 = \sqrt{\sum_i x_i^2}$
Error Bound $\leq \frac{R^2||z||_2^2}{s^2}, R = \max ||a_i||_2$
Margin: $\gamma = \max_{||z||_2=1} \min_i \langle a_i, z\rangle$
One-versus-all: $\hat{y} = \text{argmax}_k \; w_k^T x + b_k$
One-versus-one: $\#\{x^T w_{k,k'} + b_{k,k'} > 0, x^T w_{k',k} + b_{k',k} < 0\}$

Linear Regression

Gradient: if $f(x)\mathbb{R}^d \to \mathbb{R}$, $\Delta f(v) = \left(\frac{\delta f}{\delta v_1}, ..., \frac{\delta f}{\delta v_d}\right) \mathbb{R}^d \to \mathbb{R}^d$

Hessian: $\Delta^2 f(v) = \begin{bmatrix} \frac{\delta^2 f}{\delta^2 v_1^2} & ... & \delta v_d^2 \delta v_1^2 \\ \vdots & & \vdots \\ \frac{\delta^2 f}{\delta v_1^2 \delta v_d^2} & ... & \delta^2 v_d^2 \end{bmatrix} : \mathbb{R}^d \to \mathbb{R}^{d\times d}$

Emprical Risk Minimization: $\text{argmin}_w \frac{1}{n} \sum_d l_w(x, y)$
Convexity #1: $f(\lambda x_1 + (1-\lambda)x_2) \leq \lambda f(x_1) + (1-\lambda) f(x_2)$
Convexity #2: if $\Delta^2 f(x)$ is positive semi definite.
Positive Semidefinite: if $M \in \mathbb{R}^{d\times d}$, PSD iff $v^T M v \geq 0$
Loss function needs to be convex to optimizes.
Setting loss function to 0 optimizes our solution.
MLE principle: pick paramaters that maximize likelihood.
Ridge Regularization: $\arg\min_w ||Aw - z||_2^2 + \lambda ||w||_2^2$
Lasso Regularization: $\arg\min_w ||Aw - z||_2^2 + \lambda ||w||_2$

k-Nearest Neighbour Classification

Bays Optimal Classifier: $f^*(x) = \arg\max_c \Pr(y = c|x)$
NN Assumption: $Pr[y = c|x] \approx Pr[y' = c|x']$, $x \approx x'$ $y \approx y'$
No classifier can do as good as bayes.
Cant be descriped as a parameter vector.
Can express non linear relationships.
Takes 0 training time, and $O(nd)$ to $O(d \log n)$ testing time.
Small values of k lead to overfitting.
Large values of k lead to high error.
1NN Limit: as $n \to \infty$ then $L_{1NN} \leq 2L_{Bayes}(1 - L_{Bayes})$

Logistic Regression

Classifications are take into account confidence: $\hat{y} \in (-1, 1)$
Bernoulli Model: $Pr[y = 1|x, w] = p(x, w) \in (-1, 1)$
Logit Transform: $\log\left(\frac{p(x,w)}{1-p(x,w)}\right) = \langle x, w\rangle = \frac{1}{1+exp(-\langle x,w\rangle)}$
Optimizing Loss: $\Delta_w l_w(x_i, y_i) = (p_i(x_i, w) - y_i)x_i$
Iterative Update: $w_t = w_{t-1} - \eta d_i$
Gradient Descent: $d_t = \frac{1}{n}\sum_i^n \Delta_w l_{wt-1}(x_i, y_i)$
Stochastic GD: Let $B \in [n], d{-}t = \frac{1}{|B|}\sum_{i\in B} \Delta_w l_{wt-1}(x_i, y_i)$
Newton's Method d_t is given by the equation below:
$$d_t = (\tfrac{1}{n}\textstyle\sum_i^n \Delta_w^2 l_{wt-1}(x_i, y_i))^{-1}(\tfrac{1}{n}\sum_i^n \Delta_w l_{wt-1}(x_i, y_i))$$
Multiclass Logisitc Regression where k = class:
$Pr[y = k|x, w] = \frac{exp(\langle w_k, x\rangle)}{\sum_l exp(\langle w_l, x\rangle)}$

Hard-Margin SVM

Classifications are take into account confidence: $\hat{y} \in (-1, 1)$
Bernoulli Model: $Pr[y = 1|x, w] = p(x, w) \in (-1, 1)$
Logit Transform: $\log\left(\frac{p(x,w)}{1-p(x,w)}\right) = \langle x, w\rangle = \frac{1}{1+exp(-\langle x,w\rangle)}$
Optimizing Loss: $\Delta_w l_w(x_i, y_i) = (p_i(x_i, w) - y_i)x_i$
Iterative Update: $w_t = w_{t-1} - \eta d_i$
Gradient Descent: $d_t = \frac{1}{n}\sum_i^n \Delta_w l_{wt-1}(x_i, y_i)$
Stochastic GD: Let $B \in [n], d{-}t = \frac{1}{|B|}\sum_{i\in B} \Delta_w l_{wt-1}(x_i, y_i)$
Newton's Method d_t is given by the equation below:
$$d_t = (\tfrac{1}{n}\textstyle\sum_i^n \Delta_w^2 l_{wt-1}(x_i, y_i))^{-1}(\tfrac{1}{n}\sum_i^n \Delta_w l_{wt-1}(x_i, y_i))$$
Multiclass Logisitc Regression where k represents class:
$$Pr[y = k|x, w] = \frac{exp(\langle w_k, x\rangle)}{\sum_l exp(\langle w_l, x\rangle)}$$