CS 480 - Notes

Bilal Khan (b54khan)
bilal.khan@student.uwaterloo.ca

December 18, 2024

Perceptron Algorithm If linearly separable with margin γ and $||x|| \leq R$ convergence by update $k \leq R^2/\gamma^2$. Can do one-vs-all (classifier for each class) or one-vs-one (classifier for each pair of classes)

```
y_hat = w.T @ x + b
if y y_hat < delta:
  w = w + y x
  b = b + y
```

Batch vs Online learning Batch learning has IID data.

Empirical Risk Minimization make probability large / make expected loss low $\text{argmin}_w \frac{1}{n} \sum l_w(x_i, y_i)$

Regression solve for grad 0. $A^T A$ not necessarily invertible

$$\nabla_w L = 2A^T A w - 2A^T z$$
$$\nabla_w^2 L = 2A^T a$$
$$w = (A^T A)^{-1} A^T z$$

Jensen's Inequality $f(\lambda x_1 + (1-\lambda)x_2) \leq \lambda f(x_1) + (1-\lambda) f(x_2)$. A twice-differentiable function is convex iff its Hessian is positive semidefinite everywhere. PSD matrices have non-negative determinants.

Convex M is convex if $v^T M v \geq 0$ for all v. grad zero $\leftrightarrow$ global minima if the function is convex. The least-squares loss is convex.

Ridge $\lambda ||w||_2^2$

Lasso $\lambda ||w||_1$

Gaussian $p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$

Bernoulli $p^k (1-p)^{1-k}$

Covariance $E[X - E[X]]^T E[X - E[X]]$, symmetric + PSD

$$\frac{1}{\sqrt{2\pi \det[\Sigma]}} e^{-\frac{1}{2}(X-\mu)^T \Sigma^{-1}(X-\mu)}$$

Bayes $p(x, y) = p(y|x)p(x)$

Likelihood $\mathcal{L}(\mu, \sigma|x) = \Pi \frac{1}{2\pi\sigma^2} e^{-\frac{(x_i-\mu)^2}{2\sigma^2}}$

MLE when predicting $\mu = Wx$, log likelihood, grad, set to zero, solve for mu, σ^2. For a Gaussian:

$$\hat{\mu} = \frac{1}{n}\sigma x_i$$

$$\hat{\sigma^2} = \frac{1}{n}\sigma(x_i - \mu)^2$$

For Bernoulli: $\hat{p} = (1/n)y_i$

MAP Maximize the posterior using information about the prior. posterior = likelihood x prior

Entropy uncertainty in a distribution. expectation of negative log P(x). Gaussian: $1/2 \log(2\pi e\sigma^2)$. Bernoulli: $p \log p + (1 - p) \log(1 - p)$

KLD difference between distributions/relative entropy $D_{KL}(P||Q) = P(x) \log(P(x)/Q(x))$

Kernel Density Estimation $p(x) = \frac{1}{n\lambda} \sum K_\lambda(x, x_i)$

KNN classification Classify new point with majority vote of K nearest neighbor's classes.

bias variance tradeoff High bias (underfitting to complex model) vs high variance (overfitting to data points)

Curse of dimensionality max distance decreases exponentially in higher dimensions

valid kernel symmetric, distance fn $f(x, z) = f(x, y) + f(y, z)$, similarity measure $f(x, x) = 1$.

KNN clustering

$$\min_{C_1,\cdots,C_k} \sum_{j=1}^{k} \frac{1}{|C_j|} \sum_{x_i, x'_i \in C_j} ||x_i - x'_i||_2^2$$

$$\mu_j = \frac{1}{|C_j|} \sum_{x_i \in C_j} x_i \text{Centroid}$$

$$\min_{C_1,\cdots,C_k} \sum_{j=1}^{k} \frac{1}{|C_j|} \sum_{x_i \in C_j} ||x_i - \mu_j||_2^2$$

Algorithm 1 Lloyd's k-Clustering method

Input:
Dataset $D = \{x_1, x_2 \ldots, x_n\}$ $x_i \in \mathbb{R}^{\mathrm{d}}$
Number of clusters k

Output: Partition $C = \{C_1, C_2, \ldots C_k\}$

```
Randomly initialize partition C^0 = {C_1, C_2, ... C_k}
while ∃i C_i^(t) ≠ C_i^(t-1) do
    for C^t = {C_1^t, C_2^t, ... C_k^t} do
        μ_j = 1/|C_j^t| Σ_{x_i ∈ C_j^t} x_i
                                        ▷ Compute centroid of cluster j
    end for
    for {x_1, x_2 ..., x_n} do
        j = argmin_j ||x_i − μ_j||_2^2
        C_j^{t+1} ∪ {x_i}               ▷ Reassign sample i to cluster j
    end for
end while
return Partition C = {C_1, C_2, ... C_k}
```

Logistic Regression

$$p(x) = \frac{1}{1 - e^{-w \cdot x}}$$

Logistic Regression LLH

$$\log \mathcal{L}(w|x, y) = \sum_{i=0}^{n} - \log\left(1 + e^{\hat{y}(w \cdot x_i)}\right)$$

Newton's method $w_1 = w_0 - \dfrac{f'(w_0)}{f''(w_0)}$

Softmax $\dfrac{e^{w_k \cdot x_i}}{\sum_{i=0}^{c} e^{w_i \cdot x}}$

Distance between point and decision boundary: $d = \dfrac{w^T x + b}{||w||}$

Margin is min distance over all points

Maximum margin problem $\hat{w}, \hat{b} = \mathrm{argmax}_{w,b} \dfrac{1}{||w||} \min_i y_i(w^T x_i + b)$

As a constrained optimization problem $\hat{w}, \hat{b} = \mathrm{argmin}_{w,b} \frac{1}{2}||w||^2$ st $y_i(w^T x_i + b) \geq 1 \forall i$

Lagrangian dual

$$\min_{w,b} \max_{\lambda \geq 0} \mathcal{L}(w, b, \lambda) = \frac{1}{2}||w||^2 - \sum_{i=1}^{n} \lambda_i (y_i (w^T x_i + b) - 1)$$

Soft-margin SVM This is the hinge loss

$$\hat{w}, \hat{b} = \underset{w,b}{\text{argmin}} \frac{1}{2}||w||^2 + C \sum_{i=1}^{n} \max(0, 1 - y_i(w^T x_i + b))$$

$$w = \sum_{i=1}^{N} \lambda_i y_i x_i$$

It has the same objective loss fn as hard-margin, but $0 \leq \lambda \leq C$ while hard-margin is $\lambda \geq 0$

Classification Incorrectly, weakly correct (in decision boundary), strongly correct

Mercer kernels Able to be written as a dot product of a function of each input. equivalent to the kernel function being positive semidefinite

Determinant $1/(ad - bc)[d, -b, -c, a]$

SVM

$$\min_{0 \leq \lambda_i \leq C} \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \lambda_i \lambda_j y_i y_j k(x_i, x_j) - \sum_{i=1}^{n} \lambda_i$$

st $\sum_{i=0}^{n} \lambda_i y_i = 0$.

Beta distribution

$$\frac{\pi^{\alpha-1}(1-\pi)^{\beta-1}}{\sum \pi^{\alpha-1}(1-\pi)^{\beta-1}}$$

$$p(\pi) = \pi^{\alpha-1}(1-\pi)^{\beta-1}$$

$$E[\pi] = \alpha/(\alpha + \beta)$$

Beta w a bernoulli prior:

$$p(\pi|y) = Beta(k + \alpha, n - k + \beta)$$

MAP estimate is then the expectation of $p(\pi|y)$

Bayesian Linear regression

$$\bar{w} = A^{-1}(1/\sigma^2)X^T y$$

$$A = \sigma^{-2} X^T X + \Sigma^{-1}$$

$$p(w|X, y) = N(\bar{w}, A^{-1})$$

Prediction:

$$p(y^\star | x^\star, X, y) = N((x^\star)^T \bar{w}, \sigma^2 + (x^\star)^T A^{-1} x^s tar)$$

Gaussian processes

$$t_i = y_i + \epsilon_i$$
$$C_{ij} = \alpha^{-1}k(x_i, x_j) + \beta^{-1}\delta_{ij}$$
$$p(t) = N(t|0, C)$$
$$p(y) = N(y|0, \alpha^{-1}K)$$
$$p(t|y) = N(t|y, \beta^{-1}I_n)$$

prediction

$$p(t_{N+1}) = N(t_{N+1}|0, C_{N+1})$$
$$C_{N+1} = [C_n, k, k^T, c]$$
$$k = [\alpha^{-1}k(x_1, x_{N+1}), \cdots]$$
$$c = \alpha^{-1}k(x_{N+1}, x_{N+1}) + \beta^{-1}$$
$$\mu_{N+1} = k^T C_N^{-1} t$$
$$\sigma^2_{N+1} = c - k^T C^{-1} k$$

Decision Tree Predict by majority vote of class in new point's region

Gini index $\sum p_{mk}(1 - p_{mk})$

Bootstrapping Reduce variance by resampling original data with replacement

Bootstrap Aggregation (Bagging) Aggregate of multiple predictions, Can resample observations or features. Reduces variance when prediction errors are uncorrelated.

Boosting Sequentially trained ensembles can be used to reduce bias as well.

Expectation Maximization Estimate data parameters with latent variables. Compute expecation of latent variables given current parameters. Use them to compute parameters that maximize the log likelihood of the data.

In practice.

$$\mathcal{L}(\theta_t) = \sum \log p(y|\theta_t)$$
$$\theta_{t+1} = \underset{\theta}{\operatorname{argmax}} \sum \log p(y|\theta_t)$$
$$\ell_t(\theta) \geq \sum_n [-D_{KL}(q_n(z_n)||p(z_n|y_n, \theta)) + \log p(y_n|\theta)]$$
$$\hat{\gamma}_i = \frac{\pi \mathcal{N}_{\mu_2, \sigma_2^2}(x_i)}{(1-\pi)\mathcal{N}_{\mu_1, \sigma_1^2}(x_i) + \pi \mathcal{N}_{\mu_2, \sigma_2^2}(x_i)}$$
$$\hat{\pi} = \frac{\sum_{i=1}^N \hat{\gamma}_i}{N}$$

$$\hat{\mu}_1 = \frac{\sum_{i=1}^N (1-\hat{\gamma}_i) y_i}{\sum_{i=1}^N (1-\hat{\gamma}_i)}$$

$$\hat{\mu}_2 = \frac{\sum_{i=1}^N \hat{\gamma}_i y_i}{\sum_{i=1}^N \hat{\gamma}_i}$$

$$\hat{\sigma^2}_1 = \frac{\sum_{i=1}^N (1-\hat{\gamma}_i)(y_i - \hat{\mu}_1)^2}{\sum_{i=1}^N (1-\hat{\gamma}_i)}$$

$$\hat{\sigma^2}_2 = \frac{\sum_{i=1}^N \hat{\gamma}_i (y_i - \hat{\mu}_2)^2}{\sum_{i=1}^N \hat{\gamma}_i}$$

Jensen's Inequality $\log E_{q_n}[Z] \geq E_{q_n}[\log Z]$

Regularization Early stopping, weight decay, augmentation, dropout

Normalization Consistent scale, large enough batch size, IID required

CNN Locality, spatial invariance. $W_{out} = W_{in} + 2P - F)/S + 1$. Translation equivariant (adaps) (pooling) but not invariant (ignores)