

Summary for Introduction to Machine Learning 2019

General

P-Norm: $\|x\|_p = (\sum_{i=1}^n |x_i|^p)^{\frac{1}{p}}$

Frobenious Norm: $\|A\|_F = \sqrt{\sum_{i,j} a_{ij}^2}$

Derivation rules: Chain rule:

$$D(f(g(x))) = Df(g(x)) * Dg(x)$$

positive definiteness: A is p.s.d., then A is a real symmetric matrix and $x^T A x \geq 0$ for all x

Joint distribution: X, Y are RVs

$$F_{X,Y}(x, y) = \mathbb{P}(X \leq x, Y \leq y)$$

$$\text{Joint density: } f_{X,Y}(x, y) = \frac{\delta^2 F}{\delta x \delta y}(x, y)$$

$$\text{Conditional Probability: } \mathbb{P}(A|B) = \frac{\mathbb{P}(A \cap B)}{\mathbb{P}(B)}$$

Law of total probability:

$$\mathbb{P}(B) = \sum_{i=1}^n \mathbb{P}(B|A_i) \mathbb{P}(A_i)$$

$$\text{Bayes rule: } \mathbb{P}(A|B) = \mathbb{P}(B|A) \frac{\mathbb{P}(A)}{\mathbb{P}(B)}$$

Variance:

$$\text{Var}(X) = \mathbb{E}[(X - \mathbb{E}[X])^2] = \mathbb{E}[X^2] - \mathbb{E}[X]^2 \geq 0$$

Convexity: A twice differentiable function

$f: \mathbb{R}^d \rightarrow \mathbb{R}$ is convex iff for any $x \in \mathbb{R}^d$ its

Hessian is p.s.d

Convex functions are closed under addition

Regression: Predict real valued labels

Linear Regression

Goal: Measure distance between predicted and target values

$$f(x) = w_1 x_1 + \dots + w_d x_d + w_0 = \tilde{w}^T \tilde{x} \text{ with}$$

$$\tilde{w} = [w_1 \dots w_d, w_0] \text{ and } \tilde{x} = [x_1 \dots x_d, 1]$$

Residual: $r_i = y_i - w^T x_i$, $x_i \in \mathbb{R}^d$, $y_i \in \mathbb{R}$

Cost / Objective function (is convex):

$$\hat{R}(w) = \sum_{i=1}^n r_i^2 = \sum_{i=1}^n (y_i - w^T x_i)^2$$

Optimal weights:

$$w^* = \underset{w}{\operatorname{argmin}} \sum_{i=1}^n (y_i - w^T x_i)^2$$

Closed form solution: $w^* = (X^T X)^{-1} X^T y$

$$\text{Gradient: } \nabla_w \hat{R}(w) = [\frac{\delta}{\delta w_1} \hat{R}(w) \dots \frac{\delta}{\delta w_d} \hat{R}(w)] = -2 \sum_{i=1}^n r_i x_i^T$$

Non-linear functions: $f(x) = \sum_{i=1}^D w_i \phi_i(x)$

Fisher consistency

Given a surrogate loss function $\psi: Y \times S \rightarrow \mathbb{R}$,

the surrogate is said to be consistent with

respect to the loss $L: Y \times S \rightarrow \mathbb{R}$, if every minimizer f of the surrogate risk function $R_\psi(f)$ is also a minimizer of the risk function $R_L(f)$.

E.g. the hinge and the logistic losses are

consistent with respect to the 0-1 loss.

Classification losses

$$L_{\text{perceptron}}: \{-1, 1\} \times \mathbb{R} \rightarrow \mathbb{R}: y, f(x) \rightarrow$$

$$\max(0, -yf(x))$$

Find the best separation hyperplane

$$L_{\text{hinge}}: \{-1, 1\} \times \mathbb{R} \rightarrow \mathbb{R}: y, f(x) \rightarrow$$

$$\max(0, 1 - yf(x))$$

Find large separation margin

$$L_{\text{perceptron}}: \{-1, 1\} \times \mathbb{R} \rightarrow \mathbb{R}: y, f(x) \rightarrow$$

$$\max \log(1 + \exp(-yf(x)))$$

Link to cross entropy and probabilistic interpretation

Classification

$$\text{Accuracy: } \frac{TP+TN}{TP+TN+FP+FN}$$

Recall/Sensitivity/True positive rate/TPR:

$$\frac{TP}{TP+FN}$$

Specify or True negative rate/TNR: $\frac{TN}{TN+FP}$

$$\text{F1 score: } 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

Convex function

$f: \mathbb{R}^d \rightarrow \mathbb{R}$ is convex $\Leftrightarrow x_1, x_2 \in \mathbb{R}^d, \lambda \in [0, 1]:$

$$f(\lambda x_1 + (1 - \lambda)x_2) \leq \lambda f(x_1) + (1 - \lambda)f(x_2)$$

Gradient Descent

1. Start at an arbitrary $w_0 \in \mathbb{R}^d$

2. For $t = 1, 2, \dots$ do $w_{t+1} = w_t - \eta_t \nabla \hat{R}(w_t)$

Gaussian/Normal Distribution

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

Multivariate Gaussian

$$f(x) = \frac{1}{2\pi\sqrt{|\Sigma|}} e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)}$$

$$\Sigma = \begin{pmatrix} \sigma_1^2 & \sigma_{12} \\ \sigma_{21} & \sigma_2^2 \end{pmatrix}, \mu = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}$$

Empirical risk minimization

Assumption: Data set generated iid from

unknown distribution $P: (x_i, y_i) \sim P(X, Y)$.

True risk: $R(w) = \int P(x, y)(y - w^T x)^2 dx dy = \mathbb{E}_{x,y}[(y - w^T x)^2]$

Empirical risk:

$$\hat{R}_D(w) = \frac{1}{|D|} \sum_{(x,y) \in D} (y - w^T x)^2$$

Generalization error: $|R(w) - \hat{R}_D(w)|$

Uniform convergence:

$$\sup_w |R(w) - \hat{R}_D(w)| \rightarrow 0 \text{ as } |D| \rightarrow 0$$

In general, it holds that:

$$\mathbb{E}_D[\hat{R}_D(\hat{w}_D)] \leq \mathbb{E}_D[R(\hat{w}_D)], \text{ where } \hat{w}_D = \underset{w}{\operatorname{argmin}} \hat{R}_D(w).$$

Cross-validation

For each model m

For $i = 1:k$

1. Split data: $D = D_{train}^{(i)} \uplus D_{val}^{(i)}$

2. Train model: $\hat{w}_{i,m} = \underset{w}{\operatorname{argmin}} \hat{R}_{train}^{(i)}(w)$

3. Estimate error: $\hat{R}_m^{(i)} = \hat{R}_{val}^{(i)}(\hat{w}_{i,m})$

After all iterations, select model:

$$\hat{m} = \underset{m}{\operatorname{argmin}} \frac{1}{k} \sum_{i=1}^k \hat{R}_m^{(i)}$$

Ridge regression

Regularization (corresponds to MAP estimation):

$$\min_w \frac{1}{n} \sum_{i=1}^n (y_i - w^T x_i)^2 + \lambda \|w\|_2^2 =$$

$$\underset{w}{\operatorname{argmax}} P(w) \Pi_i P(y_i | x_i w)$$

Sparse regression (L1, convex) encourages

coefficients to be exactly 0 - automatic feature selection

Closed form solution: $\hat{w} = (X^T X + \lambda I)^{-1} X^T y$

Gradient: $\nabla_w (\frac{1}{n} \sum_{i=1}^n (y_i - w^T x_i)^2 + \lambda \|w\|_2^2) =$

$$\nabla_w \hat{R}(w) + 2\lambda w$$

Standardization

Goal: each feature: $\mu = 0, \sigma^2 = 1$:

$$\tilde{x}_{i,j} = \frac{(x_{i,j} - \hat{\mu}_j)}{\hat{\sigma}_j}$$

$$\hat{\mu}_j = \frac{1}{n} \sum_{i=1}^n x_{i,j}, \hat{\sigma}_j^2 = \frac{1}{n} \sum_{i=1}^n (x_{i,j} - \hat{\mu}_j)^2$$

SVM

orthogonal distance

Let $w \in \mathbb{R}^d$ and $H = \{x \in \mathbb{R}^d | \langle w, x \rangle = 0\}$ be a

hyperplane. The orthogonal distance of a point

$z \in \mathbb{R}^d$ to H can be computed as $\frac{|\langle w, z \rangle|}{\|w\|}$.

Specifically, if w is a unit vector, the inner product $\langle w, z \rangle$ directly gives the distance of z to H .

Dimension Reduction in

unsupervised learning

Principal Component Analysis (linear)

Given $D \subseteq \mathbb{R}^d, 1 \leq k \leq d, \Sigma = \frac{1}{n} \sum_{i=1}^n x_i x_i^T, \mu = \frac{1}{n} \sum_i x_i = 0$ (data is centered)

$$(W, z_1, \dots, z_n) = \underset{W}{\operatorname{argmin}} \sum_{i=1}^n \|W z_i - x_i\|_2^2$$

where $W \in \mathbb{R}^{d \times k}$ is orthogonal, $z_1, \dots, z_n \in \mathbb{R}^k$ is

given by $W = (v_1 | \dots | v_k)$ and $z_i = W^T x_i = f(x)$

where $\Sigma = \sum_{i=1}^d \lambda_i v_i v_i^T$ where $\lambda_1 \geq \dots \geq \lambda_d \geq 0$

The projection is chosen to minimize the

reconstruction error, choose k such that most of

the variance is explained (like k-means)

Kernel PCA (nonlinear)

For $k = 1$: Kernel PCA

$$\alpha^* = \underset{\alpha}{\operatorname{argmax}} \alpha^T K^T K \alpha$$

With $K = \sum_{i=1}^n \lambda_i v_i v_i^T$ ($\lambda_1 \geq \dots \geq \lambda_d \geq 0$)

$$\alpha^* = \frac{1}{\sqrt{\lambda_1}} v_1$$

For general k : Kernel PCA

The kernel principal components are given by

$$\alpha^{(1)}, \dots, \alpha^{(k)} \in \mathbb{R}^n$$

$$\alpha^{(i)} = \frac{1}{\sqrt{\lambda_i}} \text{ with } K = \sum_{i=1}^n \lambda_i v_i v_i^T$$

A new point x is projected as z ,

$$z_i = \sum_{j=1}^n \alpha_j^{(i)} k(x, x_j)$$

Kernel-PCA corresponds to applying PCA in

the feature space induced by the kernel k .

centering a kernel: $K' = K - KE - EK + EKE$

where $E = \frac{1}{n} [1, \dots, 1][1, \dots, 1]^T$

- complexity grow with number of data points,

requires data specified as kernel

Autoencoders

Goal: learn identity function $x \approx f(x; \theta)$

$$f(x; \theta) = f_{dec}(f_{enc}(x; \theta_1); \theta_2)$$

NN autoencoders are ANNs where one output

unit for each of d input units, nr of hidden units

smaller than nr of inputs. Optimize w s.t.

output agrees with input.

If activation func. is the identity, fitting NN

autoencoder is equivalent to PCA.

Decision Theory

Bayesian Decision Theory

Given: $P(y|x)$, set of actions A and cost

function $C: Y \times A \rightarrow \mathbb{R}$

$$a^* = \underset{a \in A}{\operatorname{argmin}} \mathbb{E}_y[C(y, a)|x] \text{ (cost for prediction } a$$

when true label is y)

for logistic

$$\text{regression: } \underset{y}{\operatorname{argmax}} P(y|x) = \text{sign}(w^T x) \text{ (most$$

likely class)}

Doubtful logistic regression is when we pick the

most likely class only if we are confident enough.

MAP

1. choose likelihood function \rightarrow loss function

2. choose prior \rightarrow regularizer

3. optimize for MAP parameters, choose

hyperparameters through cross-validation

4. make predictions via Bayesian Decision

Theory