Statistical Methods in AI

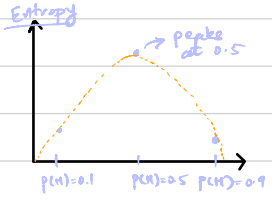
Information $I(x) = \log\left(\frac{1}{p(x)}\right)$ when $p(x) \neq 0$

$= 0$ when $p(x) = 0$.

Average Information of RV $\longleftrightarrow$ Expectation of Information

$$E[I(x)] = \sum_i p(X = x_i) \log\left(\frac{1}{p(x_i)}\right) = -\sum_i p(x_i) \log(p(x_i))$$

This average information is called shannon entropy



Entropy of 2 sided coin: $-\left(\frac{1}{2}\log\frac{1}{2} + \frac{1}{2}\log 1/2\right) = 1.$

$P(H) = 0.9$ | Entropy $= 0.9\log\left(\frac{10}{9}\right) + 0.1\log(10) = 0.46899$

Entropy ↑ → Information ↑ → uncertainty ↑ → Impurity ↑

Step-1: Compute impurity score of training distribution (Starting Impurity) 0.94

Step-2: Compute impurity score for each unique value of candidate attribute.

Step-3: Compute impurity score for candidate attribute (Take weighted average)

↳

Information Gain: $Gain(S, A) = E(S) - I(S, A)$

Step 6: Assign Root Node (with max Gain)

Recurse & Repeat step 1 to 6

properties of an impurity measure:

An Impurity measure is a function $i(V)$ s.t. .....

→ Entropy
→ Gini index
→ Misclassification rate

→ Intrinsic Information of split → more ↔ less useful (like id3 attribute) $I = -\sum \frac{|S_i|}{|S|} \log \frac{|S_i|}{|S|}$

→ Gain Ratio = $\frac{\text{Information Gain}}{\text{Intrinsic Information}}$

Cool / Mild / Hot

Sunny | 6 | 4

$-\left(2 \times \frac{4}{14} \log\left(\frac{4}{14}\right) + \frac{6}{14} \log \frac{6}{14}\right)$

$GR(\text{Out look}) = \frac{0.246}{1.5774} = 0.157$

$GR(\text{Day}) = 0.23$

$GR(\text{Temp}) = \frac{0.0289}{1.5566} = 0.018565 4292 \rightarrow$

$GR(\text{Windy}) = 0.052$

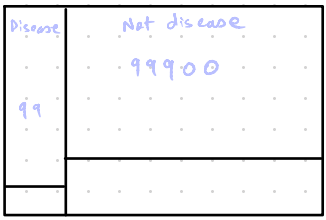
$GR(\text{Humidity}) = 0.151$

Handling Numerical attributes

Nominal can split once but not numeric attributes

Bayes Theorem

— Naresh Mawari



population = 100,000
disease = 100

Bayes Theorem: $P(H|E) = \frac{P(E|H) \cdot P(H)}{P(E|H) \cdot P(H) + P(E|\neg H) \cdot P(\neg H)}$

$P(MA) = 0.4 \quad P(MB) = 0.6$

$P(D|MA) = 0.1 \quad P(D|MB) = 0.01$

$P(MA|D) = ? \quad (4/7)$

Generalized Bayes Theorem:

If $\{E_i\}$ form a partition of sample space

Bayes Classifier

Bayes Classification

$$L_{0-1} = \begin{cases} 0 & \text{if } f(x) = y \\ 1 & \text{if } f(x) \neq y \end{cases}$$

$\{(x_i, y_i)\}\ \forall\ i \in 1 \dots N \Rightarrow$ Training set

Objective function: $\frac{1}{N} \sum_{i=1}^{N} L_{0-1}(x_i, y_i)$ ~ Average loss

IDEAL GOAL:

$$\min_{(x,y)} \mathbb{E}\left[L_{0-1}(f(x), y)\right]$$

Let

$$R(f) = \mathbb{E}\left[L_{0-1}(f(x), y)\right]$$

Lecture #9: Regression

- Idependent Variable / dependent Variable
  eg. Price of crude oil / Retail price of crude oil
- does not dependent on casuality
- Linear Regression

$$Y_i = \beta_0 + \beta_1 X_i + \varepsilon_i \rightarrow \text{Random Error}$$

(β_0: intercept, β_1: slope)

- Estimating Parameter: Least Square Method
- "Best fit" means between Act Y & pred. Y are minimum. Sint positive differences offset the the -Ve ones.
- While adding errors, +Ves may cancel out with -Ves, therefore we square the errors
- Therefore total error $E = \sum_{i=1}^{n} e_i^2$, where $e_i = y_i - \hat{y}$

$$= y_i - (\beta_0 + \beta_1 x_i)$$

$$\therefore E = \sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 x_i))^2 = f(\beta_0, \beta_1)$$

we want $f(\beta_0, \beta_1)$ minimized.

$$\therefore \frac{\partial f}{\partial \beta_0} = 0, \frac{\partial f}{\partial \beta_1} = 0$$

$$\Rightarrow \frac{\partial}{\partial \beta_0} \left(\sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 x_i))^2 \right) = 0$$

$$\Rightarrow \sum_{i=1}^{n} \frac{\partial}{\partial \beta_0} (y_i - (\beta_0 + \beta_1 x_i))^2 = 0$$

$$\Rightarrow \sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 x_i))(-1) = 0$$

$$\beta_0 = \frac{\sum y_i}{n} - \beta_1 \frac{\sum x_i}{n}$$

$$\beta_0 = \bar{y} - \beta_1 \bar{x}$$

$$f(\beta_0, \beta_1) = \sum_i (y_i - (\beta_0 + \beta_1 x_i))^2$$

$$\frac{\partial f}{\partial \beta_1} = \sum_i -2(y_i - \beta_0 - \beta_1 x_i) x_i = 0$$

$$= n\bar{y} - n\beta_0 - \beta_1 n\bar{x} = 0$$

$$\Rightarrow \beta_1 = \frac{\bar{y} - \beta_0}{\bar{x}^2}$$

$$\Rightarrow \beta_1 = \frac{\bar{y} - (\bar{y} - \beta_1 \bar{x})}{\bar{x}^2}$$

$$\beta_1 = \frac{\sum (y_i - \bar{y}) x_i}{\sum (x_i - \bar{x}) x_i}$$

$$= \frac{\frac{\sum x_i y_i}{n} - \frac{\sum x_i \bar{y}}{n}}{\frac{\sum x_i^2}{n} - \frac{\sum \bar{x} x_i}{n}} \Rightarrow \frac{\overline{xy} - \bar{x}\bar{y}}{\overline{x^2} - (\bar{x})^2}$$

$$\beta_1 =$$

$$\bar{y} - \beta_0 - \beta_1 \bar{x}$$

$$\bar{y} - \bar{y} + \beta_1 \bar{x}$$

$$\overline{xy} - \beta_0 \bar{x} - \beta_1 \overline{x^2} = 0$$

$$\overline{xy} - (\bar{y} - \beta_1 \bar{x})\bar{x} - \beta_1 \overline{x^2} = 0$$

$$\overline{xy} - \bar{x}\bar{y} - (\bar{x}^2 + \overline{x^2})\beta_1 = 0$$

$$\beta_1 = \frac{\overline{xy} - \bar{x}\bar{y}}{\bar{x}^2 - \overline{x^2}}$$

MAE = Mean Absolute Error $= \sum |y - \hat{y}| / n$

MSE = Mean squared Error $= \sum |y - \hat{y}|^2 / n$ / RMSE

look at the distribution error since MAE/MSE prone to outlier

MPE $= \frac{100}{n} \times \sum \left(\frac{y - \hat{y}}{y} \right)$ → Mean Positive error → large MPE → underestimating (MAPE)

↳ cost of underestimating / overestimating

Linear Regression – Matrix Form

$$Y = \beta X + \varepsilon$$

$$Y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \quad X = \begin{pmatrix} 1 & x_{11} & \cdots & x_{1p} \\ 1 & x_{21} & \cdots & x_{2p} \\ \vdots & \vdots & & \vdots \\ 1 & x_{n1} & \cdots & x_{np} \end{pmatrix} \quad \beta = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{pmatrix} \quad \varepsilon = \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{pmatrix}$$

$$E = e^T e$$

$$\min_{e} e^T e = \min_{\beta} (Y - X\beta)^T (Y - X\beta) = f(\beta)$$

$$\nabla_\beta f(\beta) = 0 \Rightarrow \nabla_\beta (Y^T - \beta^T X^T)(Y - X\beta)$$

$$\Rightarrow \nabla_\beta (Y^T Y - Y^T X\beta - \beta^T X^T Y + \beta^T X^T X \beta) = 0$$

Dimensions: $X = n \times p+1$

$Y = n \times 1$

$\beta = (p+1) \times 1$

$$\nabla_\beta (Y^T Y - 2\beta^T X^T Y + \beta^T X^T X \beta)$$

$$\Rightarrow 0 - 2X^T Y + 2X^T X \beta = 0$$

$$\Rightarrow X^T X \beta = X^T Y$$

$$\Rightarrow \boxed{\beta = (X^T X)^{-1} X^T Y} \Rightarrow H = (X^T X)^{-1} X^T = \text{influence Matrix}$$

$$\beta = HY$$

$n \sim p \rightarrow \beta$ comp. stable

$n > p \rightarrow$ more samples than unknown

$n < p \rightarrow$ less sample than unknowns

computing inverse → $O(n^3)$

PCA - Eigen analysis

$$\begin{bmatrix} 1 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \end{bmatrix} = 1+2^2 = 5$$

$$\begin{bmatrix} 1 \\ 2 \end{bmatrix} \begin{bmatrix} 1 & 2 \end{bmatrix} = \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix}$$

$V^T V = S$

$V^T V$

$V V^T$

Gradient descent

Linear Regression: linear in coefficients (parameters) and NOT Variables

$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \epsilon$ → we can do linear regression

correlation ≠ causation

Regression (k-NN) (& DT)

Unsupervised learning

learning the density of the data

Given $x \in X$, learn $f(x)$

Expectation Maximization

K-medians

↳ use L1 instead of L2 & instead of mean, median in KNN

Maximum Likelihood Estimation

Mid-sem will be proof based

for a concave function f and a random variable X,

$f(E[X]) \geq E[f(X)]$ → Jensen's Inequality

$$L(\theta) = \prod_{i=1}^{N} p(x_i^o ; \theta) \qquad LL(\theta) = \sum_{i=1}^{N} \log\left(p(x_i^o ; \theta)\right)$$ ← can be anything

$$\theta^* = \underset{\theta}{\operatorname{argmax}}\, L(\theta) \qquad LL(\alpha) = \sum_{i=1}^{N} \log\left(p(x_i ; \alpha)\right)$$

$$\alpha^* = \underset{\alpha}{\operatorname{argmax}}\, LL(\alpha)$$

To prove $\alpha^* = \theta^*$

Assume not, then by contradiction it can be proven

$\alpha^* = \theta^*$

$$L(\theta) = \sum_i Q_i(z_i^j) \log\left(\frac{P(x_i^o ; z_i^j, \theta)}{Q_i(z_i^j)}\right)$$ → Evidence Lower Bound (ELBO)

EM is co-ordinate ascent on ELBO

$\max(f(a,b))$

$\hat{b} = \max(f_b(\hat{a}, b))$ iteratively

$\hat{a} = \max(f_a(a, \hat{b}))$

KL-Divergence : Maximizing ELBO is minimizing $D_{KL}(Q_i(z_i) || P(z_i|x_i))$

$$KL\text{-}div(p,q) \triangleq \sum_{i=1}^{B} p_i \log\left(\frac{p_i}{z_i}\right)$$

ELBO maximized when $Q_i(z_i) \sim p(x_i|z_i)$

proof of convergence for EM.

$$\ell(\theta^{(t+1)}) \geq \ell(\theta^{(t)})$$

$$\ell(\theta) \geq \sum_i \sum_{z_i} Q_i(z_i) \log\left(\frac{P(x_i, z_i, \theta)}{Q_i(z_i)}\right) \quad -①$$

① hold for any of $Q_i(z_i)$ and θ

In particular, it holds for $Q_i = Q^{(t)}(z_i)$ and $\theta = \theta^{(t+1)}$

$$\ell(\theta^{(t+1)}) \geq \sum_i \sum_{z_i} Q_i^{(t)}(z_i) \log\left(\frac{P(x_i, z_i; \theta^{(t+1)})}{Q_i^t(z_i)}\right)$$

$$\ell(\theta^{(t)}) \geq \sum_i \sum_{z_i} Q_i^{(t)}(z_i) \log\left(\frac{P(x_i, z_i; \theta^{(t+1)})}{Q_i^t(z_i)}\right)$$

what is $\ell(\theta^{(t+1)}) - \ell(\theta^{(t)})$?

GMM advantages → Flexible, density estimation (outliers)

useful as a generative model → 1) pick cluster id by sampling from $\pi \to K$
2) sample from $N(\mu_K, \Sigma_K)$

How to choose K ?

GMM does not work in | discontinuous dataset

higher dimensions / Numerical instability

Insufficient data (N v/s # params)

Restricting Σ

EM is a general-purpose algorithm

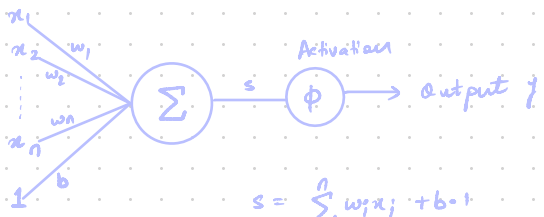
PCA is about find a good co-ordinate system such that you are able to represent your data better.

Generalized Linear Model (GLM)

Neural Networks

Neuron

Input



$$s = \sum_{i=1}^{n} w_i x_i + b \cdot 1$$

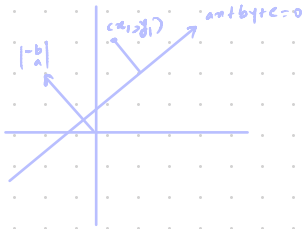
$$y = \phi(s)$$

Perceptron Rule : Reduces the number of misclassification (can be proved)

Linear classifier : Represent decision boundary by hyperplane

Batch perceptron

Difference : online learning / Batch learning



d 5:title 10:Bittorrent 7:subject 3:ACN 8:keywords 23:28P10:Bittorrent e

$bx - ay + c' = 0$

$bx - ay + ay_1 - bx_1 = 0$

$b(x - x_1) - a(y - y_1) = 0$

$$\frac{x - x_1}{a} = \frac{y - y_1}{b} = t$$

$x = at + x_1, \; y = bt + y_1$

$a(at + x_1) + b(bt + y_1) + c = 0$

$-(a^2 + b^2)t = ax_1 + by_1 + c$

$$t = -\frac{(ax_1 + by_1 + c)}{a^2 + b^2}$$

$$\sqrt{a^2t^2 + b^2t^2} = |t|\sqrt{a^2 + b^2}$$

$$= \frac{|ax_1 + by_1 + c|}{\sqrt{a^2 + b^2}}$$

$$\frac{\sinh(x)}{\cosh(x)} = \frac{e^x - e^{-x}}{}$$

$$e^{ix} = \cos x + i\sin x$$

$$e^{-ix} = \cos x - i\sin x$$

$$\cos x = (e^{ix} + e^{-ix})/2$$

$$\sin x = (e^{ix} - e^{-ix})/2$$

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

$$= \frac{1 - e^{-2x}}{1 + e^{-2x}}$$

$$= \frac{e^{2x} - 1}{e^{2x} + 1}$$

$x \to \infty$, $x \to -\infty$

$t \to 1$. $t \to -1$

Convolutional Neural Network

filters look for patterns

Head + body + Tail + legs = sparrows

each filter will identify individuals (maybe)

CNN will learn filters + weights + biases

filter → pattern detectors

3-d filter : 3×3×3 filter

Most assume depth (3 in this case)

Stride = jump

which filters to construct the network will decide / learn

dropout ↔ zero out some % of the image randomly