Perceptron

Decision line

$w_1 x_1 + w_2 x_2 = \theta$

Scaling

Z-score standardisation / normalization

$$z_i = \frac{x_i - \mu}{\sigma}$$

Min-Max Scaling

$$z_i = \frac{x_i - min(X)}{max(X) - min(X)}$$

Multiply by new range (if different than 0..1)

KNN

$$\bar{x}_j = \frac{1}{n_j} \sum_{i \in I_j} x_i \quad \text{für} \quad j = 1, \ldots, k$$

Die zu minimierende Zielfunktion bei k-means ist

$$\sum_{j=1}^{k} n_j \sum_{i \in I_j} \|x_i - \bar{x}_j\|^2 \longrightarrow \min .$$

Confusion table

		Actual value		
		true	false	
Test outcome	true	True positive (TP)	False positive (FP, Type I error)	Precision $\frac{TP}{TP + FP}$
	false	False negative (FN, Type II error)	True negative (TN)	
		Recall Sensitivity $\frac{TP}{TP + FN}$		Accuracy $\frac{TP + TN}{\#\ samples}$

$$F1Score = 2 * \frac{precision * recall}{precision + recall}$$

Linear regression

Residual sum of squares (RSS)

$$\min \mathrm{RSS}(w_0, w_1) = \sum_{i=1}^{N} (y_i - (w_0 + w_1 x_i))^2$$

Gradient Descent

$$w_i = w_i - \alpha \frac{\partial}{\partial w_i} RSS(w_0, w_1, \dots, w_n)$$

α ... *learning rate*

Ridge Regression (L_2 regularized regression)

$$\text{Total cost} = \mathrm{RSS}(w_0, \dots, w_n) + \lambda * ||w||^2$$

$$||w||^2 = w_0^2 + \dots + w_n^2$$

λ controls model complexity (Anzahl an Graden der Funktion)

Lasso Regression (L_1 regularized regression)

Total cost=RSS(w_0, …,w_n) + λ * $||w||_1$

$||w||_1 = |w_0| + \ldots + |w_n|$

Error measures

Mean Absolute Error	**Mean Squared Error**	**Relative Squared Error**
$\mathrm{MAE} = \frac{1}{n}\sum_{j=1}^{n}\lvert y_j - \hat{y}_j\rvert$ ©easycalculation.com	$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2$	$RSE = \frac{\sum_{i=1}^{n}(p_i - a_i)^2}{\sum_{i=1}^{n}(\bar{a} - a_i)^2}$
Root Relative Squared E	**Relative Absolute Error**	
= Root of RSE	$\boldsymbol{RAE = \frac{\sum_{i=1}^{n}(p_i - a_i)}{\sum_{i=1}^{n}(\bar{a} - a_i)}}$	

Correlation coefficient

Stichprobenkovarianz	**Stichprobenkorrelation**
$s_{jk} = \frac{1}{n-1}\sum_{i=1}^{n}(x_{ij} - \bar{x}_j)(x_{ik} - \bar{x}_k),$	$r_{jk} = \frac{s_{jk}}{\sqrt{s_{jj}s_{kk}}}$

Naïve Bayes

Probability of an event *H* given observed evidence *E:*

$$P(H \mid E) = P(E \mid H)P(H) / P(E)$$

A priori probability of *H* : $P(H)$

- Probability of event *before* evidence is seen

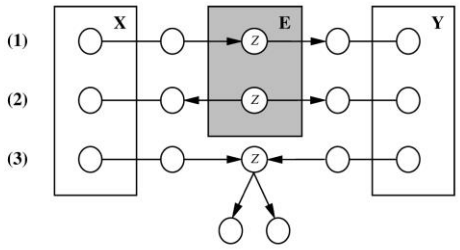
A posteriori probability of *H* : $P(H \mid E)$

- Probability of event *after* evidence is seen

Bayes Networks

- D-Separation
  - 3 Regel, die angeben, wenn zwei Knoten voneinander unabhängig sind mit einer gegebenen Evidenz
  - Fall 1
    - X ist Vorfahre von E und von Y
    - → X & Y sind voneinander statistisch unabhängig da $P(X|Y,E) = P(X|E)$
      - d.h. Y bringt kein zusätzliches Wissen
  - Fall 2
    - E ist Vorfahre von X und von Y
    - → die Information ob Y eintritt ist abhängig von E → Y bringt kein zusätzliches Wissen, wenn man E weiß
      - X & Y sind voneinander unabhängig es gilt $P(X|Y,E) = P(X|E)$
  - Fall 3
    - Kein Pfad von X zu Y beinhaltet E oder Vorfahren von E
    - X & Y haben einen gemeinsamen Abkömmling
      - da das Eintreten von Y für X nichts ändert, es gilt $P(X|Y,E) = P(X|E)$
      - Wissen wir jedoch etwas über einen gemeinsamen Abkömmling von X & Y, ändert dieses Wissen die Abhängigkeit von X zu Y

D-separation: Definition



Given: Set **E** of nodes, undirected path P.

Then, P is *blocked* relative to **E** if there is some node $Z \in P$ such that one of the following conditions hold:

- $Z \in$ **E** and one of the two arcs on P is incoming and the other is outgoing.
- $Z \in$ **E** and both arcs on P are outgoing.
- Neither Z nor a descendant of Z is in **E**, and both arcs on P are incoming.

Information Theory & Entropy

Entropy = # of bits needed for communication

Lower probability events have more information

Higher probability events have less information

for binary classification: Entropy = -(p(0) * log(P(0)) + p(1) * log(P(1)))

$$H(X) = \mathrm{E}(\mathrm{I}(X)) = \sum_{i=1}^{n} p(x_i) I(x_i) = -\sum_{i=1}^{n} p(x_i) \log_2 p(x_i)$$

H … Entropy

E … Expected value

I(X) …information content of X

p(...) … probability function

Remember:

$$\log_2(x) = \log(x) / \log(2)$$

,

Information Gain

$$\boldsymbol{IG(X_A, X_B) = H(X) - p(x_a)H(X_A) - p(x_B)H(X_B)}$$

H(X) … Entropy of unsplit data

$p(x_a)$ … Probability of Subset A = $\frac{\#Samples\ in\ A}{\#\ Samples\ in\ A\ \&\ B}$

$H(X_A)$ … Entropy of Subset A (again using all classes)

Gini impurity (Gini index)

- How often a randomly chosen element from the set would be incorrectly labeled, if it was randomly labeled according to the distribution of labels in the subset

$$I_G(p) = \sum_{i=1}^{|C|} p_i(1-p_i) = \sum_{i=1}^{|C|} (p_i - p_i^2) = \sum_{i=1}^{|C|} p_i - \sum_{i=1}^{|C|} p_i^{\,2} = 1 - \sum_{i=1}^{|C|} p_i^{\,2}$$

$$Gini\ index = I_G(p) = \mathbf{1} - \sum_{i=1}^{|C|} \boldsymbol{p_i^2}$$

$$Gini\ gain = \boldsymbol{GG(X_A, X_B) = I_G(X) - p(x_a)I_G(X_A) - p(x_B)I_G(X_B)}$$

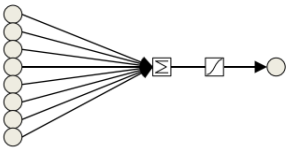
$I_{G(X)}$ … Gini index of unsplit data

$p(x_a)$ … Probability of Subset A = $\frac{\#Samples\ in\ A}{\#\ Samples\ in\ A\ \&\ B}$

$I_G(X_A)$ … Gini index of Subset A (again using all classes)

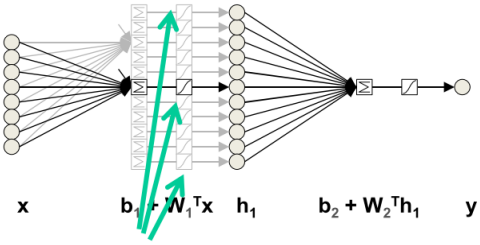
MLP

- Perceptron
- Computes: $y = \sigma(b_1 + W_1^T x)$



x b + W^T x y

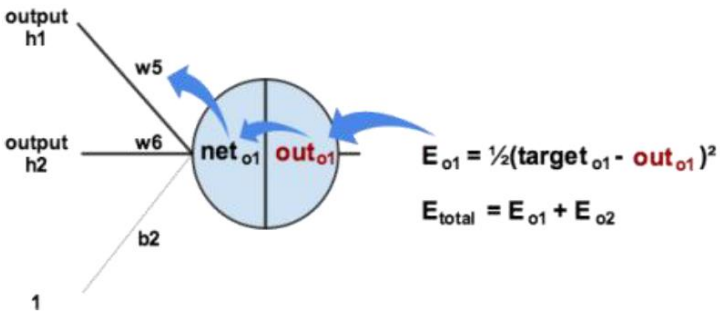
- MLP with *one* hidden layer
- Computes: $y = \sigma(b_2 + W_2^T \sigma(b_1 + W_1^T x))$



Layer composed of perceptrons (11)

E.g. weight w_5

- How much does a change in w_5 affect the error? $\frac{\partial E}{\partial w_5}$
- Via chain rule: $\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial out_{o1}} * \frac{\partial out_{o1}}{\partial net_{o1}} * \frac{\partial net_{o1}}{\partial w_5}$



Cross-Entropy Loss

$$Loss(\hat{y}, y, W) = -y \ln \hat{y} - (1 - y) \ln (1 - \hat{y})$$

$\alpha ||W||^2$: regularization term / penalty term

- Penalises complex models
- Reduces magnitude of weights

Batch Normalisation

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: γ, β
Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma^2_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma \widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

- Stochastic Gradient Descent (SGD) with Momentum:

$$\mathbf{v} \leftarrow \beta\mathbf{v} - \alpha \frac{\partial L}{\partial \theta}$$

$$\theta \leftarrow \theta - \mathbf{v}$$

• **Analogy**: Ball rolling downhill (along the cost function), building up momentum if subsequent gradients point into same direction; doesn't slow down immediately when it's flat
  – Dampen velocity according to friction coefficient ß (e.g., 0.9)
  – Increase velocity in direction of negative gradient
  – Move according to velocity

Majority Voting

$$\sum_{i=1}^{L} d_{i.k} = \max_{j=1}^{c} \sum_{i=1}^{L} d_{i,j}$$

L = # classifiers; $d_{i,k}$ = 1 if classifier i predicts class k

Weighted Majority Voting

$$\sum_{i=1}^{L} b_i d_{i.k} = \max_{j=1}^{c} \sum_{i=1}^{L} b_i d_{i,j}$$

Boosting/ AdaBoost

Classifier weight: e.g.: $\boldsymbol{\alpha(t) = \frac{1}{2} \times \log\left(\frac{1-TotalError}{TotalError}\right)}$

- if sample was classified correct: $\boldsymbol{D_{t+1}(i) = D_t(i) \times e^{-\alpha(t)}}$
  - → decrease sample weight
- if sample was classified incorrect: $\boldsymbol{D_{t+1}(i) = D_t(i) \times e^{\alpha(t)}}$
  - → increase sample weight

AdaBoost = H(x) … combination of the (weighted) individual classifiers h

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

Gradient Boosting

logistic function (converts Log(odds) to probability)

$$\frac{\boldsymbol{e^{log(x)}}}{\boldsymbol{1 + e^{log(x)}}}$$

Statistical Significance Testing

# samples correct in A & B **(N_{11})**	# samples correct A, wrong B **(N_{10})**
# samples wrong A, correct B **(N_{01})**	# samples wrong in A & B **(N_{00})**

Null hypothesis: $N_{01} = N_{10} = \frac{N_{01} + N_{10}}{2}$

$$x^2 = \frac{\left(N_{01} - \frac{N_{01} + N_{10}}{2}\right)^2}{\frac{N_{01} + N_{10}}{2}} + \frac{\left(N_{10} - \frac{N_{01} + N_{10}}{2}\right)^2}{\frac{N_{01} + N_{10}}{2}} = \frac{\left(N_{01} - N_{10}\right)^2}{N_{01} + N_{10}}$$

Actually: $$x^2 = \frac{\left(|N_{01} - N_{10}| - 1\right)^2}{N_{01} + N_{10}}$$

Paired t-Test: $$t = \frac{\overline{X}_D - \mu_0}{\frac{S_D}{\sqrt{n}}}$$

- n: number of samples
- X_D: difference of performance in each run
  - e.g. difference in *accuracy* between classifier 1 and 2
- S_D: standard deviation of difference
- μ_0 : normally set to 0

FACULTY OF

Test with t-Distribution

Number of degrees of freedom = Number of Runs – 1