

- Boxplot:

- $Q_1 = N \times 0.25$
- $Q_2 = N \times 0.5$
- $Q_3 = N \times 0.75$
- $IQR = Q_3 - Q_1$
- $Bounds = [Q_1 - 1.5 \times IQR, Q_3 + 1.5 \times IQR]$

- Pearson =  $\frac{\Sigma(y_{1i} - \bar{y}_1)(y_{2i} - \bar{y}_2)}{\sqrt{\Sigma(y_{1i} - \bar{y}_1)^2 \times \Sigma(y_{2i} - \bar{y}_2)^2}}$

- Spearman: Assign ranks and apply Pearson formula.  
Example:  $[20, 10, 20, 30, 20] \rightarrow [3, 1, 3, 5, 3]$

- Normalization:

- MinMax:  $\frac{y_i - \min}{\max - \min}$
- Standardization:  $\frac{y_i - \mu}{\sigma}$

- Binarization:

- Range (equal width): Depends on variable range  
Example:  $y \in [-1, 1] : [0.2, -0.1, 0.6] \rightarrow [1, 0, 1]$
- Frequency (equal depth): Depends on variable mean  
Example:  $\bar{y} = 25 : [10, 40, 30, 20] \rightarrow [0, 1, 1, 0]$

- Confusion Matrix:

		True			B
		A	B	C	
Pred	A	TA	FA	FA	TP
	B	FB	TB	FB	TN
	C	FC	FC	TC	FP
					FN

- Metrics:

- Accuracy =  $\frac{TP + TN}{total}$
- Error rate =  $1 - Accuracy = \frac{FP + FN}{total}$
- Recall =  $\frac{TP}{TP + FN}$  (Sensitivity)
- Fallout =  $\frac{TN}{TN + FP}$  (Specificity)
- Precision =  $\frac{TP}{TP + FP}$
- $F_1 = \frac{TP}{TP + \frac{1}{2}(FP + FN)}$

- Error:

- Sum of Squares Error:  $\sum (z - \hat{z})^2$
- Root Mean Squared Error:  $\sqrt{\frac{1}{n} SSE}$
- Mean Absolute [Percentage] Error:  $\frac{1}{n} \sum \left| \frac{z - \hat{z}}{[z]} \right|$

- $IG(y) = E(class) - \sum \frac{|class_v|}{|class|} E(class_v)$ ,  $v \in \text{Values of } y$

- $E(y) = - \sum P(y_c) \log_2(P(y_c))$ ,  $c \in \text{Values of } class$

- Decision trees:

1. Choose feature with highest IG.
2. Split dataset by that feature, create leaves if necessary.
3. Repeat until unable to proceed.

Prune: (Given a twig)

1. Count it's leaves labels. Example:  $\#A = 5$ ,  $\#B = 6$
2. Remove it's leaves.
3. Relabel twig as a leaf. Example:  $B(6/11)$ ,  $\#B > \#A$

- Vector Norm:

$$\|x\|_p = \sqrt[p]{\sum |x_i|^p} \quad \|x\|_\infty = \max |x_i|$$

- Matrix Multiplication:

$$\begin{bmatrix} \dots & n \\ m & \dots \end{bmatrix} \cdot \begin{bmatrix} \dots & l \\ n & \dots \end{bmatrix} = \begin{bmatrix} \dots & l \\ m & \dots \end{bmatrix}$$

- Gaussian Distribution:

- Variance:  $var = \frac{\sum (y_i - \mu)^2}{n[-1]}$
- Standard Deviation:  $\sigma = \sqrt{var}$
- $P(y|\mu, \sigma) = \frac{1}{\sqrt{2\pi} \cdot \sigma} \cdot \exp\left(-\frac{1}{2} \left(\frac{y - \mu}{\sigma}\right)^2\right)$

- Gaussian Mixture:

- Covariance:  $cov(y_1, y_2) = \frac{\sum (y_{1i} - \mu_1)(y_{2i} - \mu_2)}{n[-1]}$
- Covariance Matrix:  $\Sigma(y_1, y_2) = \begin{bmatrix} var(y_1) & cov(y_1, y_2) \\ cov(y_1, y_2) & var(y_2) \end{bmatrix}$
- $det|\Sigma| = det \begin{vmatrix} a & b \\ c & d \end{vmatrix} = ad - bc$
- $P(y|\mu, \Sigma) = \frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} \cdot \exp\left(-\frac{1}{2} (y - \mu)^T \Sigma^{-1} (y - \mu)\right)$

- Naive Bayes:

- MAP:  $P(C|x) = \frac{P(C)P(x|C)}{P(x)}$
- ML:  $P(C|x) = P(x|C)$
- 1. Calculate  $P(C)$  for each class.
- 2. Calculate  $P(y|C)$  for each variable for each class.
- 3. Calculate likelihood:  $P(x|C) = P(y_1|C) \times \dots \times P(y_d|C)$

- K-Nearest Neighbors:

1. Distances: (for n variables)
  - Manhattan:  $\sum |y_{1i} - y_{2i}|$
  - Euclidean:  $\sqrt{\sum (y_{1i} - y_{2i})^2}$
  - Cosine:  $\frac{\sum y_{1i} y_{2i}}{\sqrt{\sum y_{1i}^2} \sqrt{\sum y_{2i}^2}}$
  - Hamming: #Differences
2. If weighted, for each variable multiply by weight.
3. Choose K nearest neighbors.
4. Classify using mean if variable is numeric, or mode if it is categorical.

- Regressions:  $\hat{z} = Xw = (w^T X^T)^T$

- Linear:  $\hat{z} = w_0 + w_1 x$
- Polynomial:  $w = (X^T X)^{-1} X^T z$
- Ridge:  $w = (X^T X + \lambda I)^{-1} X^T z$

- Perceptron:

$$\hat{z} = a(w^T x), \quad a \leftarrow \text{activation function}$$

Gradient Descent:

$$w^{new} = w^{old} + \Delta w \quad \text{where} \quad \Delta w = -\eta \frac{\partial E}{\partial w}$$

- Neural Networks (MLP):

- Forward:  $x^{[0]} \rightarrow z^{[1]} = w^{[1]} x^{[0]} + b^{[1]} \rightarrow x^{[1]} = a(z^{[1]}) \rightarrow \dots \rightarrow z^{[i]} = w^{[i]} x^{[i-1]} + b^{[i]} \rightarrow x^{[i]} = a(z^{[i]})$
- Backward:

$$* \delta^{[last]} = \frac{\partial E}{\partial x^{[last]}} \circ \frac{\partial x^{[last]}}{\partial z^{[last]}}$$

$$* \delta^{[i]} = \left(w^{[i+1]}\right)^T \cdot \delta^{[i+1]} \circ \frac{\partial x^{[i]}}{\partial z^{[i]}}$$

$$* w^{[i]'} = w^{[i]} - \eta \cdot \frac{\partial E}{\partial w^{[i]}}$$

$$* b^{[i]'} = b^{[i]} - \eta \cdot \frac{\partial E}{\partial b^{[i]}}$$

- Multiple observations:
  1. Apply forward propagation for each observation.
  2. Calculate  $\delta_i^{[l]}$  for each observation.
  3.  $\frac{\partial E}{\partial w^{[l]}} = \sum_{i=1}^n \delta_i^{[l]} \cdot \left(x_i^{[l-1]}\right)^T$
  4.  $\frac{\partial E}{\partial b^{[l]}} = \sum_{i=1}^n \delta_i^{[l]}$
- Derivatives:

Name	Error function	$\frac{\partial E}{\partial x^{[i]}}$
Squared Error	$\frac{1}{2} \left(x^{[i]} - t\right)^2$	$x^{[i]} - t$
Cross-entropy	$-\sum_{i=1}^n t_i \log \left(x_i^{[i]}\right)$	$\frac{x^{[i]} - t}{x^{[i]} (1 - x^{[i]})}$

Name	Activation function	$\frac{\partial x^{[i]}}{\partial z^{[i]}}$
Sigmoid $\sigma(x)$	$\frac{1}{1 + e^{-x}}$	$x^{[i]} (1 - x^{[i]})$
ArcTan $\arctan(x)$	$\arctan(x)$ or $\tan^{-1}(x)$	$\frac{1}{(x^{[i]})^2 + 1}$
Hyperbolic tangent	$\frac{e^x - e^{-x}}{e^x + e^{-x}}$	$1 - (x^{[i]})^2$
ReLU	0 if $x < 0$ $x$ if $x \geq 0$	0 if $x^{[i]} < 0$ 1 if $x^{[i]} \geq 0$
Softmax	$\frac{e^{x_i}}{\sum_{j=1}^n e^{x_j}}$	$x^{[i]} (1 - x^{[i]})$
Sign	1 if $x \geq 0$ 0 if $x < 0$	

NOTE:

- \* When cross-entropy and softmax are combined:  

$$\delta^{[last]} = \frac{\partial E}{\partial z^{[last]}} = x^{[last]} - t$$

- More Derivatives:

Function	Derivative	Function	Derivative
$ax$	$a$	$f[g(x)]$	$f'[g(x)] \times g'(x)$
$x \pm y$	$x' \pm y'$	$f(x)^a$	$a f(x)^{a-1} f'(x)$
$xy$	$x'y + y'x$	$a^{f(x)}$	$a^{f(x)} f'(x) \ln a$
$\frac{x}{y}$	$\frac{xy' + y'x}{y^2}$	$\log_a f(x)$	$\frac{f'(x)}{f(x) \ln a}$

- K-Means:

1. Assign each point to a cluster.
2. Update centroids:  $\text{centroid}_{new} = \mu$  of cluster's points
3. Repeat until centroids don't change.

- Silhouette:  $\in [-1, 1]$  (the closer to 1 the better)

- For an observation  $x_i$ :
  - \*  $a$  = average distance of  $x_i$  to the points in it's cluster
  - \*  $b$  = average distance of  $x_i$  to points in closest cluster
  - \*  $s_{observation} = \frac{b - a}{\max(a, b)}$
- For a cluster:
  - Average of the cluster's observations silhouettes.
- For the solution:
  - Average of the solutions's clusters silhouettes.

- EM:

- Initializaion: Initial mixture parameters
- Expectation (E-step):  
Calculate weights for observation  $x_i$  for each cluster  $c_k$ :  

$$\gamma_{ki} = \frac{\mathcal{N}(x_i | \mu_k, \Sigma_k) \cdot \pi_k}{\sum_{j=1}^k \mathcal{N}(x_i | \mu_j, \Sigma_j) \cdot \pi_j}$$
- Maximization (M-step):  
Update parameters for each cluster: (for n observations)

$$\begin{aligned}
 * N_k &= \sum_{i=1}^n \gamma_{ki} \\
 * \mu_k &= \frac{1}{N_k} \sum_{i=1}^n \gamma_{ki} \cdot x_i \\
 * \Sigma_k &= \frac{1}{N_k} \sum_{i=1}^n \gamma_{ki} \cdot (x_i - \mu_k) \cdot (x_i - \mu_k)^T \\
 * \pi_k &= \frac{N_k}{N}
 \end{aligned}$$

- Quadratic Formula:  $ax^2 + bx + c = 0 \Rightarrow x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$

- PCA:  $\Sigma u = (\lambda I)u$

- $\Sigma$ : Covariance Matrix
- $I$ : Identity Matrix

- Eigenvalue ( $\lambda$ ):  $\det |\Sigma - \lambda I| = 0$  or  $\lambda = \Sigma u \circ \frac{1}{u}$

Example:

$$\begin{aligned}
 \lambda &= \left( \begin{bmatrix} 2.917 & 2.667 \\ 2.667 & 2.667 \end{bmatrix} \begin{bmatrix} -0.690 \\ 0.723 \end{bmatrix} \right) \circ \begin{bmatrix} -0.690^{-1} \\ 0.723^{-1} \end{bmatrix} \\
 &= \begin{bmatrix} -0.084 \\ 0.088 \end{bmatrix} \circ \begin{bmatrix} -0.690^{-1} \\ 0.723^{-1} \end{bmatrix} \\
 &= \begin{bmatrix} 0.122 \\ 0.122 \end{bmatrix}
 \end{aligned}$$

- Eigenvector ( $u$ ):  $(\Sigma - \lambda I)u = \vec{0}$

Example:

$$\begin{aligned}
 u &\Leftrightarrow \left( \begin{bmatrix} 1.333 & 0.667 \\ 0.667 & 1.667 \end{bmatrix} - \begin{bmatrix} 2.187 & 0 \\ 0 & 2.187 \end{bmatrix} \right) \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \\
 &\Leftrightarrow \begin{bmatrix} -0.854 & 0.667 & 0 \\ 0.667 & -0.52 & 0 \end{bmatrix} \Leftrightarrow \begin{matrix} L_1 \times -0.854^{-1} \\ L_2 \times 0.667^{-1} \end{matrix} \\
 &\Leftrightarrow \begin{bmatrix} 1 & -0.781 & 0 \\ 1 & -0.781 & 0 \end{bmatrix} \Leftrightarrow L_2 - L_1 \\
 &\Leftrightarrow \begin{bmatrix} 1 & -0.781 & 0 \\ 0 & 0 & 0 \end{bmatrix} \Leftrightarrow x_1 = 0.781x_2 \\
 &= \begin{bmatrix} 0.781x \\ x \end{bmatrix}
 \end{aligned}$$

- Projecting (bivariate to univariate):

1. Choose highest  $\lambda$  (higher  $\lambda$  means more variation)
2. Calculate eigenvector ( $u$ )
3. Apply formula:  $\phi = u^T X^T$

- Model complexity: ( $n$  observations,  $m$  number of features)

- K-Nearest Neighbors:  $n \times m$
- Regressions:  $m$
- Perceptron:  $m + 1$
- Tree with  $m$  features that assume  $n$  values:  $n^m$
- MLP: estimated by the number of weights
- Bayesian Classifier: estimated by the #parameters
  - \* With  $c$  classes:  $c - 1$  priors
  - \* With  $m$  features: ( $m$  averages +  $\frac{m(m+1)}{2} \Sigma$ )  $\times c$