Complete Mathematical Formulas in Machine Learning

Table of Contents

1. Foundational Mathematics
2. Statistical Measures
3. Linear Algebra in ML
4. Calculus in ML
5. Probability Theory
6. Information Theory
7. Loss Functions
8. Optimization Algorithms
9. Linear Models Mathematics
10. Tree-Based Models Mathematics
11. Support Vector Machines Mathematics
12. Neural Network Mathematics
13. Clustering Mathematics
14. Dimensionality Reduction Mathematics
15. Evaluation Metrics Mathematics

Foundational Mathematics

Basic Notation

- **Scalar**: Single value, denoted as x, y, a, b
- **Vector**: Column of values, denoted as **x**, **y** (bold)
- **Matrix**: 2D array, denoted as **X**, **W** (bold capital)
- **Tensor**: n-dimensional array

Summation and Product Notation

Summation:

$$\Sigma_{i=1}^{n} x_i = x_1 + x_2 + ... + x_n$$

Product:

$$\prod_{i=1}^{n} x_i = x_1 \times x_2 \times \ldots \times x_n$$

Statistical Measures

Mean (Average)

Arithmetic Mean:

$$\mu = \bar{x} = (1/n) \times \sum_{i=1}^{n} x_i$$

- μ (mu) or $\bar{x}$ represents the mean
- n is the number of observations
- x_i is the i-th observation

Variance

Population Variance:

$$\sigma^2 = (1/n) \times \sum_{i=1}^{n} (x_i - \mu)^2$$

Sample Variance (Bessel's correction):

$$s^2 = (1/(n-1)) \times \sum_{i=1}^{n} (x_i - \bar{x})^2$$

Standard Deviation

$$\sigma = \sqrt{(\sigma^2)} \quad \text{(population)}$$
$$s = \sqrt{(s^2)} \quad \text{(sample)}$$

Covariance

Between two variables X and Y:

$$Cov(X,Y) = (1/n) \times \sum_{i=1}^{n} (x_i - \mu_x)(y_i - \mu_y)$$

Correlation Coefficient

Pearson Correlation:

$$\rho(X,Y) = Cov(X,Y) / (\sigma_x \times \sigma_y)$$

- Range: [-1, 1]

- -1: perfect negative correlation
- 0: no linear correlation
- 1: perfect positive correlation

Standardization (Z-score)

$$z = (x - \mu) / \sigma$$

- Transforms data to have $\mu = 0$ and $\sigma = 1$

Linear Algebra in ML

Vector Operations

Dot Product (Inner Product):

$$a \cdot b = \Sigma_{i=1}^{n} a_i b_i = a_1 b_1 + a_2 b_2 + \ldots + a_n b_n$$

Vector Norm (Length):

- L1 Norm (Manhattan): $||x||_1 = \Sigma_i |x_i|$
- L2 Norm (Euclidean): $||x||_2 = \sqrt{(\Sigma_i x_i^2)}$
- L∞ Norm (Max): $||x||_\infty = \max_i |x_i|$

Matrix Operations

Matrix Multiplication:

$$C = AB \text{ where } C_{ij} = \Sigma_k A_{ik} B_{kj}$$

Matrix Transpose:

$$(A^T)_{ij} = A_{ji}$$

Matrix Inverse:

$$AA^{-1} = A^{-1}A = I$$

where I is the identity matrix

Eigenvalues and Eigenvectors

For matrix A and vector v:

```
Av = λv
```

- λ is the eigenvalue
- v is the eigenvector

Characteristic Equation:

```
det(A - λI) = 0
```

Calculus in ML

Derivatives

Basic Rules:

- Power Rule: $d/dx(x^n) = nx^{n-1}$
- Chain Rule: $d/dx[f(g(x))] = f'(g(x)) \times g'(x)$
- Product Rule: $d/dx[f(x)g(x)] = f'(x)g(x) + f(x)g'(x)$

Partial Derivatives

For function f(x,y):

```
∂f/∂x = partial derivative with respect to x
∂f/∂y = partial derivative with respect to y
```

Gradient

For function $f(x_1, x_2, ..., x_n)$:

$$\nabla f = [\partial f/\partial x_1, \partial f/\partial x_2, ..., \partial f/\partial x_n]^T$$

Gradient Descent Update Rule

$$\theta_{t+1} = \theta_t - \alpha\nabla f(\theta_t)$$

- θ: parameters
- α: learning rate
- ∇f: gradient of loss function

Hessian Matrix

Second-order partial derivatives:

$$H(f)_{ij} = \partial^2 f/(\partial x_i \partial x_j)$$

Probability Theory

Basic Probability

$$P(A) \in [0, 1]$$
$$P(\Omega) = 1 \quad \text{(probability of sample space)}$$
$$P(\emptyset) = 0 \quad \text{(probability of empty set)}$$

Conditional Probability

$$P(A|B) = P(A \cap B) / P(B)$$

- P(A|B): probability of A given B

Bayes' Theorem

$$P(A|B) = [P(B|A) \times P(A)] / P(B)$$

Extended form:

$$P(A|B) = [P(B|A) \times P(A)] / [\Sigma_i P(B|A_i) \times P(A_i)]$$

Probability Distributions

Bernoulli Distribution:

$$P(X = x) = p^x(1-p)^{1-x} \text{ for } x \in \{0,1\}$$

Binomial Distribution:

$$P(X = k) = C(n,k) \times p^k \times (1-p)^{n-k}$$

where C(n,k) = n!/(k!(n-k)!)

Normal (Gaussian) Distribution:

$$f(x) = (1/\sqrt{(2\pi\sigma^2)}) \times \exp(-(x-\mu)^2/(2\sigma^2))$$

Multivariate Normal:

$$f(x) = (1/((2\pi)^{k/2}|\Sigma|^{1/2})) \times \exp(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu))$$

- Σ: covariance matrix
- $|\Sigma|$: determinant of Σ

Expectation and Variance

Expectation (Expected Value):

- Discrete: $E[X] = \Sigma_x\, x \times P(X = x)$
- Continuous: $E[X] = \int x \times f(x)dx$

Variance:

$$Var(X) = E[(X - E[X])^2] = E[X^2] - (E[X])^2$$

Information Theory

Entropy

For discrete variable X:

$$H(X) = -\Sigma_i\, P(x_i) \times \log_2(P(x_i))$$

- Measures uncertainty/information content
- Units: bits ($\log_2$) or nats (ln)

Cross-Entropy

Between distributions P and Q:

$$H(P,Q) = -\Sigma_i\, P(x_i) \times \log(Q(x_i))$$

Kullback-Leibler (KL) Divergence

$$KL(P||Q) = \Sigma_i\, P(x_i) \times \log(P(x_i)/Q(x_i))$$

- Measures difference between distributions
- $KL(P||Q) \neq KL(Q||P)$ (not symmetric)

Mutual Information

$$I(X;Y) = \Sigma_x \Sigma_y P(x,y) \times \log(P(x,y)/(P(x)P(y)))$$

Loss Functions

Regression Loss Functions

Mean Squared Error (MSE):

$$MSE = (1/n) \times \Sigma_{i=1}^{n} (y_i - \hat{y}_i)^2$$

Mean Absolute Error (MAE):

$$MAE = (1/n) \times \Sigma_{i=1}^{n} |y_i - \hat{y}_i|$$

Huber Loss (Robust to outliers):

$$L_\delta(y,\hat{y}) = \begin{cases} \frac{1}{2}(y-\hat{y})^2 & \text{if } |y-\hat{y}| \le \delta \\ \delta|y-\hat{y}| - \frac{1}{2}\delta^2 & \text{if } |y-\hat{y}| > \delta \end{cases}$$

Classification Loss Functions

Binary Cross-Entropy (Log Loss):

$$BCE = -(1/n) \times \Sigma_{i=1}^{n} [y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)]$$

Categorical Cross-Entropy (Multi-class):

$$CCE = -(1/n) \times \Sigma_{i=1}^{n} \Sigma_{j=1}^{m} y_{ij} \log(\hat{y}_{ij})$$

- m: number of classes
- y_{ij}: 1 if sample i belongs to class j, 0 otherwise

Hinge Loss (SVM):

$$L = \max(0, 1 - y \times \hat{y})$$

Optimization Algorithms

Gradient Descent Variants

Batch Gradient Descent:

$$\theta = \theta - \alpha \times (1/n) \times \Sigma_{i=1}^{n} \nabla\theta L(x_i, y_i, \theta)$$

Stochastic Gradient Descent (SGD):

$$\theta = \theta - \alpha \times \nabla\theta L(x_i, y_i, \theta)$$

Mini-batch Gradient Descent:

$$\theta = \theta - \alpha \times (1/m) \times \Sigma_{i=1}^{m} \nabla\theta L(x_i, y_i, \theta)$$

Advanced Optimizers

Momentum:

$$v_t = \beta v_{t-1} + \alpha \nabla\theta L$$
$$\theta_t = \theta_{t-1} - v_t$$

RMSprop:

$$s_t = \beta s_{t-1} + (1-\beta)(\nabla\theta L)^2$$
$$\theta_t = \theta_{t-1} - \alpha \times \nabla\theta L / \sqrt{(s_t + \varepsilon)}$$

Adam (Adaptive Moment Estimation):

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)\nabla\theta L \quad \text{(1st moment)}$$
$$v_t = \beta_2 v_{t-1} + (1-\beta_2)(\nabla\theta L)^2 \quad \text{(2nd moment)}$$
$$\hat{m}_t = m_t / (1-\beta_1^t) \quad \text{(bias correction)}$$
$$\hat{v}_t = v_t / (1-\beta_2^t) \quad \text{(bias correction)}$$
$$\theta_t = \theta_{t-1} - \alpha \times \hat{m}_t / (\sqrt{\hat{v}_t} + \varepsilon)$$

Linear Models Mathematics

Linear Regression

Model:

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n = x^T \beta$$

Normal Equation (Closed-form solution):

$$\beta = (X^T X)^{-1} X^T y$$

Cost Function (MSE):

$$J(\beta) = (1/2n) \times \Sigma_{i=1}^{n} (y_i - x_i^T \beta)^2$$

Gradient:

$$\nabla_\beta J = -(1/n) \times X^T (y - X\beta)$$

Ridge Regression (L2 Regularization)

Cost Function:

$$J(\beta) = (1/2n) \times \Sigma_{i=1}^{n} (y_i - x_i^T \beta)^2 + \lambda ||\beta||_2^2$$

Solution:

$$\beta = (X^T X + \lambda I)^{-1} X^T y$$

Lasso Regression (L1 Regularization)

Cost Function:

$$J(\beta) = (1/2n) \times \Sigma_{i=1}^{n} (y_i - x_i^T \beta)^2 + \lambda ||\beta||_1$$

- No closed-form solution
- Solved using coordinate descent or proximal gradient

Logistic Regression

Sigmoid Function:

$$\sigma(z) = 1/(1 + e^{-z})$$

Model:

$$P(y=1|x) = \sigma(x^T \beta) = 1/(1 + e^{-x^T \beta})$$

Log-Likelihood:

$$LL = \Sigma_{i=1}^{n} [y_i \log(\sigma(x_i^T \beta)) + (1-y_i) \log(1-\sigma(x_i^T \beta))]$$

Gradient:

$$\nabla_\beta LL = X^T(y - \sigma(X\beta))$$

Tree-Based Models Mathematics

Decision Trees

Gini Impurity:

$$Gini = 1 - \Sigma_{j=1}^{c} p_j^2$$

- c: number of classes
- p_j: proportion of samples in class j

Entropy:

$$Entropy = -\Sigma_{j=1}^{c} p_j \log_2(p_j)$$

Information Gain:

$$IG = Entropy(parent) - \Sigma_i \ (n_i/n) \times Entropy(child_i)$$

- n_i: number of samples in child i
- n: total samples in parent

Variance Reduction (for regression):

$$VR = Var(parent) - \Sigma_i \ (n_i/n) \times Var(child_i)$$

Random Forest

Prediction (Regression):

$$\hat{y} = (1/B) \times \Sigma_{\beta=1}^{B} T_\beta(x)$$

- B: number of trees
- T_β: prediction from tree b

Prediction (Classification):

$$\hat{y} = \text{mode}\{T_1(x), T_2(x), ..., T^B(x)\}$$

Feature Importance:

$$\text{Importance}_j = (1/B) \times \Sigma_{\beta=1}^{B} \Sigma_{t \in \beta} 1(v(t)=j) \times p(t)\Delta_{it}$$

- v(t): variable used at node t
- p(t): proportion of samples reaching node t
- Δ_{it}: impurity decrease at node t

Gradient Boosting

Additive Model:

$$F(x) = \Sigma_{m=1}^{M} \gamma_m h_m(x)$$

Update Rule:

$$F_m(x) = F_{m-1}(x) + \gamma_m h_m(x)$$

Gradient Calculation:

$$r_{im} = -[\partial L(y_i, F(x_i))/\partial F(x_i)]_{F=F_{m-1}}$$

Support Vector Machines Mathematics

Hard Margin SVM

Objective:

$$\text{minimize: } \frac{1}{2}||w||^2$$
$$\text{subject to: } y_i(w^T x_i + b) \geq 1 \text{ for all } i$$

Soft Margin SVM

Objective:

$$\text{minimize: } \frac{1}{2}||w||^2 + C \times \sum_i \xi_i$$
$$\text{subject to: } y_i(w^T x_i + b) \geq 1 - \xi_i$$
$$\xi_i \geq 0$$

- ξ_i: slack variables
- C: regularization parameter

Kernel Trick

Kernel Function:

$$K(x, x') = \phi(x)^T \phi(x')$$

Common Kernels:

- Linear: $K(x, x') = x^T x'$
- Polynomial: $K(x, x') = (\gamma x^T x' + r)^d$
- RBF (Gaussian): $K(x, x') = \exp(-\gamma||x - x'||^2)$
- Sigmoid: $K(x, x') = \tanh(\gamma x^T x' + r)$

Dual Form Prediction:

$$f(x) = \sum_{i=1}^{n} \alpha_i y_i K(x_i, x) + b$$

Neural Network Mathematics

Forward Propagation

Linear Transformation:

$$z^{(l)} = W^{(l)} a^{(l-1)} + b^{(l)}$$

Activation:

$$a^{(l)} = g^{(l)}(z^{(l)})$$

Activation Functions

ReLU (Rectified Linear Unit):

$$\text{ReLU}(z) = \max(0, z)$$

Leaky ReLU:

$$\text{LeakyReLU}(z) = \max(\alpha z, z) \quad \text{where } \alpha \approx 0.01$$

Sigmoid:

$$\sigma(z) = 1/(1 + e^{-z})$$

Tanh:

$$\tanh(z) = (e^{z} - e^{-z})/(e^{z} + e^{-z})$$

Softmax (Multi-class output):

$$\text{softmax}(z_i) = e^{(z_i)} / \Sigma_j \, e^{(z_j)}$$

Backpropagation

Chain Rule Application:

$$\partial L/\partial W^{(l)} = \partial L/\partial z^{(l)} \times \partial z^{(l)}/\partial W^{(l)}$$

Error Propagation:

$$\delta^{(l)} = (W^{(l+1)})^{T}\delta^{(l+1)} \odot g'(z^{(l)})$$

- $\odot$: element-wise multiplication

Weight Update:

$$W^{(l)} = W^{(l)} - \alpha \times \delta^{(l)}(a^{(l-1)})^{T}$$

Batch Normalization

Normalization:

$$\hat{x}_i = (x_i - \mu B)/\sqrt{(\sigma B^{2} + \varepsilon)}$$

Scale and Shift:

$$y_i = \gamma \hat{x}_i + \beta$$

- γ, β: learnable parameters

Dropout

Training:

$$a^{(l)} = a^{(l)} \odot m^{(l)} / p$$

- $m^{(l)}$: binary mask (Bernoulli(p))
- p: keep probability

Clustering Mathematics

K-Means

Objective Function:

$$J = \Sigma_{i=1}^{n} \Sigma_{k=1}^{K} r_{ik} ||x_i - \mu_k||^2$$

- r_{ik}: 1 if x_i belongs to cluster k, 0 otherwise

Update Rules:

$$\mu_k = (\Sigma_i r_{ik} x_i)/(\Sigma_i r_{ik}) \quad \text{(centroid update)}$$
$$r_{ik} = 1 \text{ if } k = \text{argmin}_j ||x_i - \mu_j||^2 \quad \text{(assignment)}$$

DBSCAN

Core Point:

$$|N\varepsilon(p)| \geq \text{minPts}$$

- $N\varepsilon(p) = \{q \in D \mid \text{dist}(p,q) \leq \varepsilon\}$

Density-Reachable:

- p is reachable from q if there's a chain of core points

Gaussian Mixture Models

Probability Model:

$$p(x) = \Sigma_{k=1}^{K} \pi_k \times N(x|\mu_k, \Sigma_k)$$

E-step (Expectation):

$$\gamma_{ik} = (\pi_k \times N(x_i|\mu_k, \Sigma_k))/(\Sigma_j \pi_j \times N(x_i|\mu_j, \Sigma_j))$$

M-step (Maximization):

$$\pi_k = (1/n) \times \Sigma_i \gamma_{ik}$$
$$\mu_k = (\Sigma_i \gamma_{ik}x_i)/(\Sigma_i \gamma_{ik})$$
$$\Sigma_k = (\Sigma_i \gamma_{ik}(x_i - \mu_k)(x_i - \mu_k)^T)/(\Sigma_i \gamma_{ik})$$

Dimensionality Reduction Mathematics

Principal Component Analysis (PCA)

Objective: Find projection that maximizes variance

Covariance Matrix:

$$C = (1/n) \times X^TX$$

Eigendecomposition:

$$C = V\Lambda V^T$$

- V: eigenvectors (principal components)
- Λ: eigenvalues (diagonal matrix)

Projection:

$$Z = XV_k$$

- V_k: first k eigenvectors

Reconstruction:

$$\hat{X} = ZV_k^T$$

Explained Variance Ratio:

$$EVR = \lambda_i / \Sigma_j \lambda_j$$

Linear Discriminant Analysis (LDA)

Between-class Scatter:

$$S_B = \Sigma_k \, n_k(\mu_k - \mu)(\mu_k - \mu)^T$$

Within-class Scatter:

$$S_W = \Sigma_k \, \Sigma_{i \in C_k} \, (x_i - \mu_k)(x_i - \mu_k)^T$$

Objective:

$$\text{maximize: } (w^T S_B w)/(w^T S_W w)$$

Solution: Eigenvectors of $S_W^{-1} S_B$

t-SNE

Joint Probability (High-dimensional):

$$p_{ij} = (p_{j|i} + p_{i|j})/(2n)$$

where:

$$p_{j|i} = \exp(-||x_i - x_j||^2/(2\sigma_i^2)) \, / \, \Sigma_{k \neq i} \exp(-||x_i - x_k||^2/(2\sigma_i^2))$$

Joint Probability (Low-dimensional):

$$q_{ij} = (1 + ||y_i - y_j||^2)^{-1} \, / \, \Sigma_{k \neq l} \, (1 + ||y_k - y_l||^2)^{-1}$$

Cost Function (KL divergence):

$$C = KL(P||Q) = \Sigma_i \, \Sigma_j \, p_{ij} \log(p_{ij}/q_{ij})$$

Evaluation Metrics Mathematics

Classification Metrics

Confusion Matrix Elements:

- TP: True Positives
- TN: True Negatives
- FP: False Positives
- FN: False Negatives

Accuracy:

$$\text{Accuracy} = (TP + TN)/(TP + TN + FP + FN)$$

Precision:

$$\text{Precision} = TP/(TP + FP)$$

Recall (Sensitivity, True Positive Rate):

$$\text{Recall} = TP/(TP + FN)$$

Specificity (True Negative Rate):

$$\text{Specificity} = TN/(TN + FP)$$

F1 Score:

$$F1 = 2 \times (\text{Precision} \times \text{Recall})/(\text{Precision} + \text{Recall})$$

F-beta Score:

$$F\beta = (1 + \beta^2) \times (\text{Precision} \times \text{Recall})/((\beta^2 \times \text{Precision}) + \text{Recall})$$

Matthews Correlation Coefficient:

$$MCC = (TP \times TN - FP \times FN)/\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}$$

ROC and AUC

ROC Curve: Plot of TPR vs FPR at various thresholds

AUC (Area Under ROC Curve):

$$AUC = \int_0^1 TPR(FPR)\, dFPR$$

Approximation (Trapezoidal Rule):

$$AUC \approx \Sigma_i \tfrac{1}{2}(TPR_i + TPR_{i+1})(FPR_{i+1} - FPR_i)$$

Regression Metrics

R^2 Score (Coefficient of Determination):

$$R^2 = 1 - (SS_{res}/SS_{tot})$$

where:

$$SS_{res} = \Sigma_i (y_i - \hat{y}_i)^2 \quad \text{(residual sum of squares)}$$
$$SS_{tot} = \Sigma_i (y_i - \bar{y})^2 \quad \text{(total sum of squares)}$$

Adjusted R^2:

$$R^2_{adj} = 1 - [(1-R^2)(n-1)/(n-p-1)]$$

- p: number of predictors

Mean Absolute Percentage Error (MAPE):

$$MAPE = (100/n) \times \Sigma_i |y_i - \hat{y}_i|/|y_i|$$

Clustering Metrics

Silhouette Score:

$$s(i) = (b(i) - a(i))/\max(a(i), b(i))$$

where:

- a(i): average distance to points in same cluster
- b(i): minimum average distance to points in other clusters

Davies-Bouldin Index:

$$DB = (1/k) \times \Sigma_{i=1}^{k} \max_{j \neq i} [(\sigma_i + \sigma_j)/d(c_i, c_j)]$$

- σ_i: average distance of points in cluster i to centroid
- $d(c_i, c_j)$: distance between centroids

Calinski-Harabasz Index:

$$CH = [tr(B_k)/(k-1)] / [tr(W_k)/(n-k)]$$

- B_k: between-group dispersion matrix

- W_k: within-group dispersion matrix

Additional Important Formulas

Regularization

Elastic Net (L1 + L2):

$$J(\theta) = L(\theta) + \lambda_1||\theta||_1 + \lambda_2||\theta||_2^2$$

Distance Metrics

Euclidean Distance:

$$d(x,y) = \sqrt{(\Sigma_i (x_i - y_i)^2)}$$

Manhattan Distance:

$$d(x,y) = \Sigma_i |x_i - y_i|$$

Cosine Similarity:

$$\cos(x,y) = (x \cdot y)/(||x||_2 \times ||y||_2)$$

Minkowski Distance:

$$d(x,y) = (\Sigma_i |x_i - y_i|^p)^{(1/p)}$$

Bias-Variance Decomposition

Total Error:

$$E[(y - \hat{y})^2] = \text{Bias}^2 + \text{Variance} + \text{Irreducible Error}$$

where:

$$\text{Bias} = E[\hat{y}] - y$$
$$\text{Variance} = E[(\hat{y} - E[\hat{y}])^2]$$

Cross-Validation Error

k-Fold CV Error:

$$CV(k) = (1/k) \times \Sigma_{i=1}^{k} L(h_i, D_i)$$

- h_i: model trained on all folds except i
- D_i: validation data from fold i