Core Concepts for Machine Learning

Abdul Samad Khan

May 2025

Learning Algorithms

We aim to minimize a cost function $J(\theta)$ to learn the best parameters θ. This is usually done using gradient-based optimization. The model defines a family of functions, and the algorithm tunes θ to make predictions align with data. Objective: find $\theta^* = \arg\min_\theta J(\theta)$.

Capacity, Overfitting and Underfitting

Capacity = model's ability to fit a wide variety of functions. Too little $\rightarrow$ underfitting, too much $\rightarrow$ overfitting. Goal: balance generalization and performance. Regularization helps control capacity (e.g., L1, L2 penalties). Bias–variance trade-off lives here.

Hyperparameters and Validation Sets

Hyperparameters aren't learned from training data (e.g., learning rate, number of layers). We tune them using a validation set — separate from both training and test sets. Good validation strategy = good generalization.

Estimators, Bias and Variance

Estimator: a function of data to approximate some quantity (e.g., sample mean). Bias: error due to wrong assumptions. Variance: sensitivity to data fluctuations. Trade-off between the two defines generalization performance.

Maximum Likelihood Estimation (MLE)

MLE = choose parameters that maximize likelihood of observed data:

$$\theta^* = \arg\max_\theta \prod_{i=1}^{m} p_{\text{model}}(x^{(i)}; \theta)$$

In practice, we minimize negative log-likelihood instead. MLE connects directly to loss functions like cross-entropy.

Bayesian Statistics

Bayes = put a distribution over θ, then update via Bayes' rule:

$$p(\theta|\mathcal{D}) \propto p(\mathcal{D}|\theta)p(\theta)$$

Gives uncertainty in predictions and parameters. More principled but computationally heavier than MLE.

Supervised Learning Algorithms

Learn mapping $x \mapsto y$ from labeled data $(x^{(i)}, y^{(i)})$. Includes logistic regression, neural nets, SVMs. Optimized with loss functions like MSE (for regression) or NLL (for classification).

Unsupervised Learning Algorithms

No labels. Goal: discover structure in x. Clustering (e.g., k-means), density estimation (e.g., Gaussian Mixture Models), dimensionality reduction (e.g., PCA, autoencoders).

Stochastic Gradient Descent (SGD)

Instead of computing full gradient $\nabla_\theta J(\theta)$ (which is $O(m)$), we estimate it using minibatches:

$$g = \frac{1}{m'} \sum_{i=1}^{m'} \nabla_\theta L(x^{(i)}, y^{(i)}, \theta)$$

Then update: $\theta \leftarrow \theta - \eta g$. Cheap, scalable, works well with large datasets.

Building a Machine Learning Algorithm

Design pipeline: define model class, choose objective, pick optimizer, tune hyperparameters. Training + validation cycle matters more than model class alone. Practical tuning = half the game.

Challenges Motivating Deep Learning

Traditional ML plateaus on raw data. Feature engineering is hard. Deep learning solves this by learning representations hierarchically — especially useful in vision, speech, and NLP. Data + compute + depth = power.