Machine Learning — compressed "everything" map (with pros/cons)

Workflow invariants: define target + data → split correctly → pick baseline → tune/regularize → evaluate (incl. slices) → calibrate/threshold → deploy + monitor drift.

Problem types & core framing

- **Supervised:** regression, classification, ranking. Key: label quality, leakage, imbalance, costs.
- **Unsupervised:** clustering, density estimation, dimensionality reduction. Key: evaluation is indirect.
- **Self-supervised:** learn representations from pretext tasks (contrastive/masked).
- **Reinforcement learning:** sequential decisions, delayed reward; exploration vs exploitation.
- **Structured prediction:** sequences/graphs/sets; needs specialized losses/decoding.
- **Generative modeling:** model data distribution; sampling quality vs likelihood.

Data & splitting (common failure modes)

- **Split:** random (IID), **time-based** (avoid future leakage), user/group split (avoid identity leakage).
- **Leakage:** target leakage, time-travel joins, label leakage via features, preprocessing on full data.
- **Imbalance:** use PR-AUC, class weights, focal loss, resampling; calibrate after reweighting.
- **Missing/outliers:** explicit missing indicators; robust losses; winsorize where justified.
- **Bias/variance:** high bias → add features/model capacity; high variance → regularize/more data/ensembles.

Classical supervised algorithms (when to use)

Linear regression: **+** fast, interpretable, convex; great baseline – underfits nonlinearity; sensitive to outliers (use Huber/RANSAC)
Ridge/Lasso/Elastic Net: **+** controls overfit; L1 gives sparsity/feature selection – L1 unstable with correlated features; scaling matters
Logistic regression: **+** strong baseline; calibrated-ish; handles large sparse – linear boundary; needs feature engineering
Naive Bayes: **+** very fast; good for text; small data – strong independence assumption; weaker accuracy if violated
k-NN: **+** simple; non-parametric; good local patterns – slow at inference; curse of dimensionality; needs scaling
Decision tree: **+** interpretable; handles nonlinearity; mixed types – high variance; unstable; overfits without pruning
Random forest: **+** robust; strong default; handles interactions – larger models; less interpretable; slower than linear
Gradient boosting (XGBoost/LightGBM/CatBoost): **+** SOTA on tabular; handles nonlinearity; good with missing – tuning sensitive; can overfit; slower training; leakage still kills
SVM (linear/kernel): **+** strong margin; kernel handles nonlinearity – kernel doesn't scale; tuning C/γ; probability calibration needed
Gaussian processes: **+** uncertainty + flexible kernels; great small data – $O(n^3)$ training; hard at scale; kernel choice critical

Unsupervised & representation learning

PCA: **+** fast linear DR; denoise; interpretable components – linear only; variance ≠ usefulness; scaling matters
t-SNE/UMAP: **+** great visualization of manifolds – not for downstream metrics; unstable; parameters affect layout
k-means: **+** fast; simple; scalable – assumes spherical clusters; needs k; sensitive to init/outliers
GMM (EM): **+** soft clusters; ellipses; density estimate – local minima; assumes Gaussian; needs k/model selection
DBSCAN/HDBSCAN: **+** finds arbitrary shapes; detects noise – parameter sensitive; varying density is hard (HDBSCAN helps)
Isolation Forest: **+** strong anomaly baseline; works high-dim – interpretability limited; contamination assumptions
One-class SVM: **+** anomaly detection w/ boundary – scales poorly; parameter sensitive
Autoencoder (AE): **+** learn nonlinear embeddings; anomaly via recon error – may reconstruct anomalies; needs tuning/regularization

Neural networks: components, architectures, and training knobs

Activations (choose for gradient flow + expressivity)

ReLU: **+** simple; sparse; fast; default for MLP/CNN – dead neurons; unbounded outputs
Leaky ReLU/PReLU: **+** mitigates dead ReLU – extra hyperparams; still unbounded
ELU/SELU: **+** better mean/variance dynamics – slower; SELU requires specific init/alpha-dropout
GELU/Swish: **+** smooth; strong for Transformers – slightly slower; can be less stable in some setups
Sigmoid: **+** probabilities; gates – vanishing gradients; not for deep hidden layers
tanh: **+** zero-centered vs sigmoid – still saturates; vanishing gradients

Normalization & regularization

BatchNorm: **+** stabilizes training; allows higher LR – batch-size dependent; train/serve mismatch; less common in Transformers
LayerNorm: **+** stable for sequences/Transformers – extra compute; can hurt some CNNs vs BN
GroupNorm: **+** works with small batch sizes – often slower; may underperform BN at large batch
Dropout: **+** reduces co-adaptation; simple – can slow convergence; not always helpful with BN; tune rate
Weight decay (L2 / AdamW): **+** strong regularizer; simple – too much hurts fit; separate from LR schedule (AdamW best practice)
Early stopping: **+** cheap; prevents overfit – needs reliable val; can stop too early with noise

Optimizers & LR schedules

SGD: **+** good generalization; stable – needs tuning; slower convergence
SGD+momentum/Nesterov: **+** faster; standard for vision – still needs LR schedule tuning
Adam: **+** fast convergence; good default – may generalize worse; sensitive to weight decay coupling
AdamW: **+** fixes weight decay coupling – still needs warmup/schedule
RMSProp/Adagrad: **+** handles sparse/ill-conditioned – Adagrad LR decays too much; RMSProp less common now

- **Schedules:** cosine/linear decay, step, one-cycle, warmup (esp. Transformers), ReduceLROnPlateau.
- **Init:** Xavier/Glorot (tanh), He/Kaiming (ReLU), orthogonal for RNNs.

Loss functions (what they optimize) — pros/cons

MSE: **+** smooth; optimizes mean; standard regression – sensitive to outliers; blurs multimodal targets
MAE: **+** robust; optimizes median – non-smooth at 0; slower convergence
Huber: **+** robust + smooth – δ threshold tuning
Cross-entropy (softmax): **+** standard classification; well-behaved gradients – needs label noise handling; can be miscalibrated
Binary cross-entropy: **+** multi-label; logistic – thresholding/calibration needed
Hinge: **+** margin-based; SVM-style – not probabilistic; less common in deep nets
Focal loss: **+** handles class imbalance; hard examples – γ tuning; can hurt calibration
Label smoothing: **+** improves generalization; reduces overconfidence – hurts if you need true probabilities; tune ε
Contrastive / Triplet: **+** metric learning; embeddings – mining negatives/hard pairs required; collapse risk
InfoNCE: **+** self-supervised contrastive; strong reps – needs many negatives or large batch/memory bank
Pairwise ranking (hinge/logistic): **+** directly optimizes ordering – sampling bias; needs good negatives
Listwise (softmax/NDCG surrogates): **+** closer to ranking metric – more complex; can be unstable/tuning-heavy
CTC: **+** alignment-free sequence labeling – assumes conditional independence; decoding complexity
Quantile / pinball: **+** predicts conditional quantiles – needs quantile choice; can be unstable
GAN losses (minimax/hinge/WGAN-GP): **+** sharp samples; implicit density – training instability; mode collapse; sensitive to tricks
Diffusion training (noise pred / score): **+** stable; high sample quality – slow sampling (mitigated by distillation/fast samplers)
VAE ELBO: **+** likelihood + latent structure – posterior collapse; blurry samples vs GAN/diffusion

Architectures (when they shine)

MLP: **+** tabular + embeddings; simplest deep baseline – weak inductive bias for images/sequences
CNN: **+** translation equivariance; efficient; vision – global context harder; architecture tuning
RNN/LSTM/GRU: **+** streaming/low latency seq; small models – hard long-range deps; slower than attention
Transformer: **+** long-range deps; parallelizable; SOTA NLP/vision – $O(n^2)$ attention cost; memory heavy; needs data/compute
GNN (GCN/GAT/GraphSAGE): **+** relational data; inductive on graphs – oversmoothing; sampling; deployment complexity
Mixture-of-Experts: **+** scale capacity w/ sparse compute – routing instability; systems complexity; load balancing

Evaluation, calibration, uncertainty, interpretability, fairness

Metrics (pick to match business objective)

- **Classification:** accuracy (only if balanced), precision/recall/F1, ROC-AUC (ranking), PR-AUC (imbalance), logloss (prob quality).
- **Calibration:** ECE/MCE, reliability curves; calibrate with Platt/Isotonic/temperature scaling.
- **Ranking:** NDCG@k, MAP, MRR, Recall@k; report at multiple k; watch position bias.
- **Regression:** RMSE, MAE, R^2, MAPE/SMAPE (careful near 0).
- **Vision:** mAP, IoU/Dice, top-k acc; **NLP:** BLEU/ROUGE (limited), exact match; **Gen:** human eval + safety.
- **Operational:** p95/p99 latency, cost/query, memory, availability, drift metrics.

Model selection & tuning

Cross-validation: **+** better estimate with small data – expensive; leakage if groups/time ignored
Grid search: **+** simple; exhaustive for small spaces – blows up combinatorially
Random search: **+** efficient in high-dim – non-adaptive; still expensive
Bayesian optimization: **+** sample-efficient; adaptive – overhead; noisy objectives; implementation complexity
Early pruning (ASHA/Hyperband): **+** saves compute – can kill late-blooming configs

Imbalanced / noisy labels

- **Imbalance fixes:** class weights, focal loss, undersample negatives, oversample positives (careful), hard-negative mining.
- **Noise:** robust losses, label smoothing, bootstrapping, confident learning; audit labeling pipeline.
- **Thresholding:** optimize for cost curve; choose operating point per segment; consider calibration.

Uncertainty & ensembling

Deep ensembles: **+** strong uncertainty + accuracy – training cost ×N; deployment complexity
MC dropout: **+** cheap uncertainty approx – calibration varies; slower inference (multiple passes)
Bayesian methods / Laplace: **+** principled uncertainty – hard to scale; approximations
Quantile regression: **+** prediction intervals – only for certain losses/tasks

Interpretability

Coefficients (linear): **+** transparent global explanation – misses interactions/nonlinearities
Tree feature importance: **+** fast global signal – biased toward high-cardinality; unstable
SHAP: **+** strong local attributions – compute heavy; assumptions; can be misused
LIME: **+** model-agnostic local explanations – unstable; depends on perturbation distribution
Counterfactual explanations: **+** actionable what-if – hard constraints; may be unrealistic

Fairness / harm reduction

- **Metrics:** demographic parity, equalized odds/opportunity, calibration by group; also long-tail user harm.
- **Mitigations:** reweighting, constraints, adversarial debiasing, post-processing thresholds.
- **Tradeoff:** fairness constraints can reduce global metric; define policy + measurement plan.

Causal inference (when correlation is not enough)

A/B testing: **+** gold standard; causal – needs time/traffic; interference; novelty effects
Uplift modeling: **+** targets treatment effect heterogeneity – label is counterfactual; noisy; needs careful eval
Propensity/IPS: **+** debias observational – variance; needs good propensity model
Doubly robust: **+** less bias/variance – more complexity; still assumptions

Common pitfalls (say these to sound senior)

- Train/serve skew; leakage; wrong split; hidden objective mismatch (opt metric ≠ business).
- Over-optimizing offline metrics; ignoring calibration/threshold; ignoring slices/long tail.
- Ignoring data quality + label delay; not monitoring drift; no rollback.
- Spurious correlations; fairness regressions; feedback loops (recs/ads).

Modern ML: self-supervised, NLP/CV, generative models, RL, time series

Embeddings & similarity search

Dense embeddings: **+** capture semantics; reuse across tasks – need good negatives; drift; embedding staleness
Approx NN (HNSW/IVF/PQ): **+** fast retrieval at scale – recall/latency tradeoff; index build/update complexity
Bi-encoder vs cross-encoder: **+** bi: fast retrieval; cross: accurate rerank – bi loses interaction; cross expensive

Self-supervised learning

Contrastive (SimCLR/MoCo): **+** strong representations; label-free – needs augmentations/negatives; batch/memory heavy
Masked modeling (BERT/MAE): **+** works without negatives; scalable – pretext-task mismatch; compute heavy
Distillation: **+** smaller/faster models – teacher bias; needs careful objective

NLP / LLM training & adaptation

SFT (supervised fine-tune): **+** aligns to task; simple – needs quality data; overfitting/catastrophic forgetting
LoRA/PEFT: **+** cheap adaptation; small deltas – still needs eval; may not match full fine-tune
RLHF / preference optimization: **+** improves helpfulness/safety – reward hacking; high complexity; evaluation hard
RAG: **+** injects fresh knowledge; reduces hallucination – retrieval errors; latency; prompt/grounding complexity

Generative models

VAE: **+** latent structure; likelihood-based – blurry samples; posterior collapse risk
GAN: **+** sharp samples – instability; mode collapse; hard to evaluate
Autoregressive: **+** good likelihood; controllable – slow generation; exposure bias
Diffusion: **+** stable training; SOTA quality – slow sampling; large compute (mitigations exist)
Normalizing flows: **+** exact likelihood; invertible – architectural constraints; memory/compute

Reinforcement learning

Q-learning / DQN: **+** sample efficient (off-policy) – unstable; overestimation; needs replay/target nets
Policy gradient: **+** works with continuous actions – high variance; needs baselines
Actor-critic: **+** lower variance; efficient – more moving parts; instability
PPO: **+** robust default; stable updates – can be sample hungry; sensitive to reward shaping

- **Core concepts:** MDP, reward shaping, discount γ, exploration (ϵ-greedy/Thompson), off-policy vs on-policy, credit assignment.

Time series & forecasting

ARIMA/SARIMA: **+** interpretable; strong for linear seasonal – needs stationarity; limited nonlinear patterns
ETS / state-space: **+** handles trend/seasonality – model selection; less flexible
Prophet: **+** easy defaults; holidays – not best for complex series; can mislead without tuning
RNN/Transformer forecasting: **+** captures nonlinear + covariates – needs data; leakage risks; compute
Anomaly detection TS: **+** detect drift/spikes – false positives; thresholding; seasonality handling

Training tricks & stability

- **Mixed precision:** faster; watch underflow/grad scaling.
- **Gradient clipping:** stabilizes RNN/Transformers; can hide LR issues.
- **Classical tricks:** data augmentation, mixup/cutmix, label smoothing, EMA weights.
- **Monitoring:** loss curves, gradient norms, activation stats, overfit gap, calibration drift.

Choosing a method (quick heuristics)

- **Tabular:** start with GBDT; consider MLP w/ embeddings if lots of categorical + interactions.
- **Text:** pretrained Transformers + fine-tune/PEFT; add RAG if knowledge changes.
- **Vision:** pretrained CNN/ViT; augment; consider distillation for latency.
- **Graphs:** start with features + GBDT; then GNN if relational signal matters.
- **Small data:** linear/trees + strong regularization; transfer learning; uncertainty/GP if feasible.