MACHINE LEARNING CHEAT SHEET

Foundations · evaluation · probability/statistics · classic ML · deep learning · Transformers/LLMs

1 · MODEL REASONING FRAME

Task/target → data/split → baseline → metric → model/knobs → trade-off → validation. Consider data size/types, error costs, latency and interpretability. State assumptions; prefer a simple, measurable baseline before adding complexity.

2 · FOUNDATIONS & GENERALISATION

Parameter learned from data (weights, split thresholds). **Hyperparameter**: chosen outside training (depth, λ, learning rate), tuned on validation/CV—not test.
Loss / objective per-example error / loss plus constraints or regularisation. Empirical risk minimisation minimises average training loss.
Overfit low train error, high validation error: model learned noise. **Underfit**: both high: insufficient capacity/features/training.
Bias–variance simple models: high bias/low variance; flexible models: lower bias/higher variance. More data mainly reduces variance.
Regularisation prefers simpler solutions by penalising complexity. Stronger $\lambda \Rightarrow$ more bias, less variance; tune at minimum validation error. ↔ **underfit vs overfit**

- **L2 / ridge** $+\lambda\|w\|_2^2$: smoothly shrinks weights; stable with correlated features.
- **L1 / lasso** $+\lambda\|w\|_1$: sparse/feature selection; unstable among correlated features.
- Trees: prune/limit depth/leaves. DL: weight decay, dropout, augmentation, early stopping.

Split correctly train fits; validation selects; test estimates once. Stratify labels; for time use forward splits; for users/groups use group splits. Fit preprocessing inside each fold.
Leakage training sees unavailable future/target/test information; yields deceptively good evaluation. Audit timestamp and unit of prediction.
Cross-validation k fits; mean estimates performance, spread its instability. Nested CV gives honest evaluation after tuning.
Scaling needed for distance/gradient models (kNN, SVM, linear/NN); trees largely invariant. Impute/encode using train only.

3 · METRICS: MATCH THE COST

Regression	MAE robust/linear cost; RMSE punishes large errors; R^2 vs mean baseline.
Classification	Precision $= TP/(TP+FP)$; recall $= TP/(TP+FN)$; F1 balances both. Threshold controls precision ↔ recall.
Ranking / prob.	PR-AUC for rare positives; ROC-AUC ranks broadly; log loss/Brier assess probabilities. Calibrate if probabilities drive decisions.
Imbalance	Stratify; class/sample weights; resample train only; report PR curve + confusion matrix at business threshold. Accuracy can mislead.

4 · PROBABILITY, STATS & A/B

$E[X]$, $Var(X)$ centre and spread; $Var(X) = E[(X - E[X])^2]$. Covariance is scale-dependent; correlation normalises to $[-1, 1]$—neither implies causation.
Conditional probability $P(A|B) = P(A \cap B)/P(B)$. Bayes: posterior $\propto$ likelihood $\times$ prior. Independence: $P(A, B) = P(A)P(B)$.
LLN / CLT sample mean converges to expectation / its sampling distribution tends normal under conditions; $SE(\bar{x}) = s/\sqrt{n}$. More n narrows uncertainty.
Confidence interval procedure whose intervals cover the true value in e.g. 95% of repeated samples—not 95% probability for this fixed interval.
Hypothesis test H_0 vs H_1; p-value is $P(\text{data at least this extreme} \mid H_0)$, not $P(H_0 \mid data)$. α = Type-I/false-positive risk; β = Type-II; power $= 1 - \beta$.
A/B recipe predefine unit, primary metric, MDE, α/power and horizon → randomise → check assignment/guardrails → estimate effect + CI → practical, not only statistical, significance. Avoid peeking/multiple tests or correct them.
Common tests means: t-test (Welch by default); proportions: z/χ^2; paired/repeated: paired test; non-normal robust option: bootstrap/permutation. Dependence demands cluster-aware SEs.
Optimisation gradient points uphill; GD steps $\theta \leftarrow \theta - \eta\nabla L$. Convex: local=global; non-convex NNs rely on useful minima. Too-large η diverges; too-small is slow.

5 · CLASSIC ML: HOW + KNOBS

Linear / logistic regression — linear score; logistic maps it through sigmoid to class probability. Fast, interpretable baseline; linear boundary unless features expand it.
Tune: λ or C$= 1/\lambda$, L1/L2, class weight, decision threshold.
Decision tree — greedy feature splits reduce impurity. Nonlinear/interactions, no scaling; unstable and overfits.
Tune: `max_depth`, `min_samples_leaf/split`, max features, pruning α.
Random forest — bagged trees on bootstrap rows + random feature subsets; averaging reduces variance. Robust/parallel, less interpretable.
Tune: number of trees (stability↑, cost↑), max features, depth/min leaf, bootstrap, class weight.
Gradient boosting / XGBoost — sequential shallow trees fit residual/negative gradients; often strongest tabular model.
Tune: learning rate ↓ needs trees ↑; depth/leaves, subsample/columns, L1/L2, early stopping. ↔ **accuracy vs time/overfit**

9 · RAPID-FIRE DISTINCTIONS

Generative vs discr.	Model $p(x, y)$ or generate x / model $p(y\|x)$ or decision boundary.
Bagging / boosting	Parallel averaging lowers variance / sequential correction lowers bias (and can overfit).
Epoch / batch / step	Full pass / examples per update / one optimiser update.
Calibration / ranking	Probabilities match frequencies / positives rank above negatives.
Data / concept drift	Input $p(x)$ changes / relationship $p(y\|x)$ changes. Monitor both.

Production checklist: reproducible pipeline and baseline; representative split; leakage checks; metric plus business cost; calibration and threshold; slice/fairness tests; latency, memory and cost; monitoring for data/concept drift; rollback plan and retraining trigger.

5 · CLASSIC ML, CONTINUED

SVM — maximum-margin separator; kernels create nonlinear similarity features. Strong medium-sized/high-dimensional data; expensive at large n.
Tune: C: violations vs margin; kernel; RBF γ: locality/complexity. Scale inputs.
k-nearest neighbours — predict from nearby labelled examples. No training, local/nonlinear; slow inference, curse of dimensionality.
Tune: k: small=flexible/noisy, large=smooth/biased; distance, weights. Scale features.
Naive Bayes — Bayes with conditionally independent features. Extremely fast; good text baseline; probabilities often poor.
Tune: Laplace smoothing α; Gaussian variance smoothing / distribution family.
k-means — alternate nearest-centroid assignment and centroid update; minimises within-cluster squares. Spherical, scaled clusters.
Tune: k, initialisation (k-means++), restarts, tolerance. Select with stability/business use, not silhouette alone.
PCA — orthogonal directions of maximum variance via eigendecomposition/SVD. Compression/denoising; components less interpretable.
Tune: number/variance of components; whitening. Centre (usually scale) first.

6 · MODEL CHOICE & DIAGNOSIS

Start	Dummy → regularised linear/logistic → tree ensemble; choose complexity only if validation and constraints justify it.
Train bad; val bad	Underfit: better features, less regularisation, more capacity/training; verify optimiser/data.
Train good; val bad	Overfit: more/cleaner data, stronger regularisation, simpler model, augmentation/early stopping; fix leakage/split first.
Learning curves	Gap that persists suggests variance; both plateau badly suggests bias; more data helps variance, not fundamental bias.
Explain	Linear coefficients (after scale); tree permutation importance/SHAP. Correlation and importance are not causality. Check slices/drift.

7 · DEEP LEARNING

Mechanism layers compose affine transforms + nonlinear activations; backprop uses chain rule to compute gradients; minibatch optimisation updates weights. Capacity comes from width/depth.
Architecture knobs layers/width (capacity ↔ compute/overfit); activation (ReLU/GELU); residual paths aid deep optimisation; batch/layer norm stabilise scale; initialisation preserves signal.
Training knobs learning rate is usually most important; batch size (noise/memory/throughput); epochs/early stopping; scheduler/warmup; dropout; weight decay; augmentation; gradient clipping; seed.
SGD + momentum cheap, noisy, can generalise well; slower/tuning-sensitive. Momentum smooths and accelerates consistent directions.
Adam adaptive per-parameter steps + momentum; fast/easy default, more memory; $\beta_1, \beta_2, \epsilon$.
AdamW decouples weight decay from Adam update; standard Transformer choice. Tune learning rate + weight decay.
RMSProp / Adagrad adaptive moving-square / accumulated-square gradients; useful for some recurrent/sparse settings; Adagrad learning rate can vanish.
Stable recipe normalise → sensible init → AdamW or momentum SGD → LR warmup + decay → monitor train/val → checkpoint/early-stop → clip exploding gradients. Vanishing gradients: residuals/norm/gated units.

8 · TRANSFORMERS & LLMs

Transformer, roughly token IDs → embeddings + position → repeated blocks: masked multi-head self-attention → residual + layer norm → position-wise MLP → residual + norm → vocabulary logits. Decoder-only LLM trains next-token cross-entropy (teacher forcing).
Attention $Q = XW_Q, K = XW_K, V = XW_V$; softmax$(QK^\top/\sqrt{d_k})V$. Each token mixes relevant token values; heads learn different relations. Causal mask blocks future tokens. Cost is $O(n^2)$ in context length.
Model hyperparameters layers, hidden size, heads/head dimension, MLP size, context length, vocabulary, dropout; training: tokens, batch, LR/warmup/schedule, AdamW/weight decay, precision. ↔ **quality vs compute/memory/latency**
Generation knobs temperature ↓ = sharper/deterministic; top-p/top-k truncate tail; max tokens; repetition penalty; beam search favours likely sequences but can be bland. These do not change weights.
Fine-tuning ladder (1) define task + held-out eval; (2) clean/deduplicate prompt–response data; (3) continued pretraining for domain language or **SFT** for behaviour; (4) full FT or **LoRA** low-rank adapters / **QLoRA** quantised base; (5) preference tuning **DPO** or RLHF; (6) safety, regression, slice and human eval. Low LR; mask prompt loss when appropriate.
Distillation train a smaller student to match teacher soft logits/distributions (KL at temperature T), outputs or reasoning traces, often mixed with hard-label loss. Gains speed/cost; loses capacity and can inherit teacher errors. Distillation transfers behaviour; quantisation reduces numeric precision; pruning removes weights.
RAG vs tuning RAG retrieves current, attributable knowledge into context; tuning changes behaviour/style/domain patterns but is a poor database. Combine when needed.