Cross-Validation (Cross-Val)
Cross-val tests model on new data. Split data into folds, train on most, test on one, repeat. Avg performance is reliable.
Pros: Better perf estimate than single split; Detects overfitting.
Cons: More time/compute for many folds/large data; Tricky for time-series.

Convex Optimization (Convex-Opt)
Convex opt finds global best in bowl-shaped space. Used in SVMs, regression.
Pros: Guarantees global min; Efficient solvers.
Cons: Not all probs convex, need approx; Heavy for large probs.

Gradient Descent (GD)
GD updates params opposite grad of loss to min errors.
Pros: Simple to implement; Good for convex.
Cons: Slow on large data (full set/step); Stuck in local min for non-convex.

Stochastic Gradient Descent (SGD)
SGD like GD but updates w/one random point, faster/noisier.
Pros: Faster on large data; Escapes local min via noise.
Cons: Noisy, erratic; Needs LR scheduling.

Mini-batch Gradient Descent
Mini-batch GD updates w/small batches, balances GD/SGD.
Pros: Faster than GD, less noisy than SGD; GPU-efficient.
Cons: Batch size tuning needed; Can stuck in local min.

Data Augmentation
Data aug mods existing ex (rotate, noise) for robust models.
Pros: More data w/o collect; Better gen, esp images.
Cons: May add unreal data; Compute-heavy in train.

Lagrangian
Lagrangian combines obj func w/constraints via mults for opt pts.
Pros: Solves eq/ineq constraints; Base for SVMs.
Cons: Complex math; Needs KKT checks.

Dual Lagrangian
Dual reformulates primal, often easier, esp kernels.
Pros: Simplifies computation in many cases; Enables kernel trick for non-linear problems.
Cons: May increase complexity for some formulations; Requires careful handling of dual variables.

K-Nearest Neighbors (KNN)
KNN classifies a new data point based on the majority label of its 'k' closest neighbors in the training data, using distance metrics like Euclidean.
Pros: Simple and intuitive, no training phase needed; Works well for non-linear data.
Cons: Slow for large datasets (computes distances at prediction time); Sensitive to irrelevant features and noise.

Naive Bayes
Naive Bayes is a probabilistic classifier that applies Bayes' theorem, assuming features are independent, to predict class probabilities.
Pros: Fast and efficient, especially for high-dimensional data like text; Performs well even with the 'naive' independence assumption.
Cons: Assumption of feature independence often unrealistic; Struggles with zero-probability issues (use smoothing).

Linear Discriminant Analysis (LDA)
LDA projects data onto a lower-dimensional space to maximize class separability, assuming Gaussian distributions and equal covariances.
Pros: Good for dimensionality reduction while preserving class info; Computationally efficient.
Cons: Assumes normality and equal covariances, which may not hold; Linear boundaries only.

Logistic Regression
Logistic Regression models the probability of binary outcomes using a sigmoid function on a linear combination of features.
Pros: Interpretable coefficients show feature importance; Handles binary and multi-class (via one-vs-rest).
Cons: Assumes linear decision boundaries; Sensitive to multicollinearity.

Support Vector Machines (SVM)
SVM finds the hyperplane that best separates classes with the maximum margin, using support vectors.
Pros: Effective in high-dimensional spaces; Robust to overfitting with proper regularization.
Cons: Computationally intensive for large datasets; Sensitive to choice of kernel and parameters.

Kernel SVM
Kernel SVM extends SVM to non-linear data by mapping to higher dimensions via kernels (e.g., RBF) without explicit transformation.
Pros: Handles complex, non-linear boundaries; Versatile with different kernels.
Cons: More computationally expensive; Risk of overfitting if kernel not chosen well.

Linear Regression
Linear Regression fits a line to data by minimizing squared errors, predicting outputs as a linear combination of inputs.
Pros: Simple and interpretable; Fast to train.
Cons: Assumes linearity; poor for complex relationships; Sensitive to outliers.

Ridge Regression
Ridge Regression adds L2 regularization to linear regression to shrink coefficients and handle multicollinearity.
Pros: Reduces overfitting and stabilizes estimates; Good for correlated features.
Cons: Includes all features (no selection); Bias introduced by regularization.

Lasso Regression
Lasso Regression uses L1 regularization, which can set some coefficients to zero for feature selection.
Pros: Performs automatic feature selection; Handles multicollinearity.
Cons: Can be unstable with highly correlated features; Bias like Ridge.

Kernel Ridge
Kernel Ridge combines Ridge regression with kernels for non-linear fitting.
Pros: Captures non-linear patterns; Regularization prevents overfitting.
Cons: Computationally heavy for large data; Kernel tuning required.

Support Vector Regression (SVR)
SVR adapts SVM for regression, finding a function that deviates from actual values by at most epsilon.
Pros: Robust to outliers; Effective in high dimensions.
Cons: Sensitive to parameter choice (C, epsilon); Slow for large datasets.

Kernel SVR
Kernel SVR uses kernels for non-linear regression in SVR.
Pros: Handles complex non-linear data; Flexible with kernels.
Cons: Increased complexity and compute; Overfitting risk.

Polynomial Regression
Polynomial Regression fits higher-degree polynomials to capture non-linear trends.
Pros: Simple extension of linear regression; Good for curved relationships.
Cons: Prone to overfitting with high degrees; Extrapolation can be poor.

K-Means
K-Means partitions data into k clusters by minimizing within-cluster variance, assigning points to nearest centroids.
Pros: Simple and scalable; Fast convergence.
Cons: Needs k specified; sensitive to initialization; Assumes spherical clusters.

Gaussian Mixture Model (GMM)
GMM models data as a mixture of Gaussian distributions, using EM to estimate parameters.
Pros: Handles elliptical clusters and soft assignments; Probabilistic outputs.
Cons: Slower than K-Means; sensitive to init; Assumes Gaussian components.

Perceptron
Perceptron is a single-layer neural network for linear classification, updating weights on errors.
Pros: Basic building block of NNs; Converges for linearly separable data.
Cons: Only linear; no hidden layers; Doesn't handle XOR-like problems.

Multi-Layer Perceptron (MLP)
MLP adds hidden layers to Perceptron for non-linear learning via backpropagation.
Pros: Universal approximator for functions; Handles complex data.
Cons: Prone to overfitting; needs regularization; Black-box; hard to interpret.

Convolutional Neural Networks (CNN)
CNN uses convolutional layers for feature extraction, ideal for grid data like images.
Pros: Excellent for spatial hierarchies (e.g., images); Parameter sharing reduces compute.
Cons: Requires large data and GPU; Overfits without augmentation.

KNN and Naive Bayes
Similarities: Both simple classifiers for beginners; Non-parametric (KNN) or probabilistic (NB).
Differences: KNN instance-based (lazy); NB model-based (eager); KNN slow predict; NB fast but assumes independence.

Naive Bayes and LDA
Similarities: Probabilistic, assume Gaussian-like distributions; Good for text/multi-class.
Differences: NB independent features; LDA shared covariances; LDA for dim reduction; NB pure classification.

LDA and Logistic Regression
Similarities: Linear decision boundaries; Used for classification.
Differences: LDA generative (models distributions); Logistic discriminative (probabilities); LDA assumes normality; Logistic no distribution assumption.

Logistic Regression and SVM
Similarities: Linear classifiers; can be regularized; Binary/multi-class capable.
Differences: Logistic probs via sigmoid; SVM margins via hyperplane; SVM better for small data; Logistic interpretable.

SVM and Kernel SVM
Similarities: Maximize margins for separation; Use support vectors.
Differences: SVM linear; Kernel non-linear via mapping; Kernel more flexible but slower.

Linear Regression and Logistic Regression
Similarities: Linear models; optimized via gradients; Interpretable.
Differences: Linear for continuous; Logistic for binary probs; Linear MSE loss; Logistic cross-entropy.

Ridge Regression and Lasso Regression
Similarities: Regularized linear regression; Handle multicollinearity/overfitting.
Differences: Ridge L2 (shrinks); Lasso L1 (selects); Lasso for sparse; Ridge keeps all.

Kernel Ridge and Kernel SVR
Similarities: Kernel-based for non-linear regression; Regularized.
Differences: Kernel Ridge least-squares; SVR epsilon-tube; SVR robust to outliers; Kernel Ridge variance-focused.

SVR and Kernel SVR
Similarities: SVM-based regression; Margin/epsilon concept.
Differences: SVR linear; Kernel non-linear; Kernel captures curves better but more compute.

Polynomial Regression and Kernel Ridge
Similarities: Non-linear extensions of linear; Fit curves.
Differences: Polynomial explicit degrees; Kernel implicit via functions; Polynomial overfitting risk; Kernel regularized.

K-Means and GMM
Similarities: Cluster data unsupervised; Iterative (centroids/EM).
Differences: K-Means hard assign, spherical; GMM soft, probabilistic, elliptical; GMM more flexible but slower.

Perceptron and MLP
Similarities: Neural networks; weight updates; Building blocks.
Differences: Perceptron single-layer linear; MLP multi-layer non-linear; MLP backprop; Perceptron simple rule.

MLP and CNN
Similarities: Deep NNs with hidden layers; Backprop training.
Differences: MLP fully connected; CNN convolutional for spatial; CNN better for images; MLP general.

Sequential Minimal Optimization (SMO)
SMO solves the SVM dual by updating two Lagrange multipliers at a time while keeping constraints satisfied.
Pros: Efficient for large SVM problems; Avoids large QP solvers; Works well with kernels.
Cons: More complex to implement than simple GD; Speed depends on heuristics for picking pairs.

RANSAC
RANSAC repeatedly samples minimal subsets, fits a model, and counts inliers to find a robust fit under many outliers.
Pros: Very robust to outliers; Simple concept; Works well for geometric vision tasks.
Cons: Needs many iterations if inlier ratio low; Requires thresholds and max-iter tuning.

Expectation Maximization (EM)
EM maximizes a latent-variable likelihood by alternating: E-step (compute posteriors/expectations) and M-step (maximize expected complete log-likelihood).
Pros: Handles missing/latent variables naturally; Closed-form updates for models like GMM.
Cons: Converges only to local maxima; Can be slow; Sensitive to initialization.

Dimensionality Reduction
Dimensionality reduction maps high-dim data to lower-dim space while preserving structure (variance, distances, or class info).
Pros: Reduces storage and computation; Helps visualization and denoising.
Cons: May discard useful information; Choice of method and target dim is non-trivial.

Feature Selection
Feature selection chooses a subset of input features (filter, wrapper, embedded methods) instead of transforming them.
Pros: Improves interpretability; Can reduce overfitting and training time.
Cons: Search can be expensive; Risk of discarding informative but weak features.

Linear Dimensionality Reduction
Linear DR finds projections $z = W^\top x$ that keep most variance or class separation (e.g., PCA, LDA).
Pros: Simple and fast; Often has eigenvalue/eigenvector closed forms.
Cons: Only captures linear structure; Fails on curved manifolds (non-linear relations).

Singular Value Decomposition (SVD)
SVD: $X = U\Sigma V^\top$, with orthogonal U, V and singular values in Σ.
Pros: Basis of PCA and low-rank approximations; Optimal rank-k approximation in Frobenius norm.
Cons: Expensive on very large matrices; Often needs truncated or randomized SVD.

Principal Component Analysis (PCA)
PCA finds directions of maximum variance (eigenvectors of covariance, or top right-singular vectors of X).
Pros: Unsupervised linear DR; Decorrelates features; Often improves downstream methods.
Cons: Components are linear and not label-aware; Sensitive to scaling and outliers.

Kernel PCA
Kernel PCA applies PCA in an implicit feature space using a kernel matrix instead of the covariance of raw features.
Pros: Captures non-linear structure; Works with same kernels as Kernel SVM.
Cons: Needs storing and eigendecomposing $N \times N$ kernel matrix; Less interpretable than standard PCA.

Whitening
Whitening transforms data so that it has zero mean and identity covariance (decorrelated, unit variance). Often done after PCA.
Pros: Removes linear correlations; Useful preprocessing for some models and ICA.
Cons: Can amplify noise in low-variance directions; Requires good covariance estimate.

Looking at Learning Curves
Learning curves plot train and validation error vs. training set size or epochs.
Pros: Helps diagnose high-bias vs. high-variance; Guides whether to get more data or change model complexity.
Cons: Requires repeated training; Interpretation can be ambiguous with noisy curves.

PCA, Kernel PCA, Whitening
Similarities: All linear transforms in some space; Used for preprocessing and dimensionality reduction.
Differences: PCA linear in input space; Kernel PCA non-linear via kernels; Whitening rescales to identity covariance (often after PCA) instead of just keeping top-variance directions.