

# User Guide

## 1. Supervised learning

### ▼ 1.1. Linear Models

- [1.1.1. Ordinary Least Squares](#)
- [1.1.2. Ridge regression and classification](#)
- [1.1.3. Lasso](#)
- [1.1.4. Multi-task Lasso](#)
- [1.1.5. Elastic-Net](#)
- [1.1.6. Multi-task Elastic-Net](#)
- [1.1.7. Least Angle Regression](#)
- [1.1.8. LARS Lasso](#)
- [1.1.9. Orthogonal Matching Pursuit \(OMP\)](#)
- [1.1.10. Bayesian Regression](#)
- [1.1.11. Logistic regression](#)
- [1.1.12. Generalized Linear Regression](#)
- [1.1.13. Stochastic Gradient Descent - SGD](#)
- [1.1.14. Perceptron](#)
- [1.1.15. Passive Aggressive Algorithms](#)
- [1.1.16. Robustness regression: outliers and modeling errors](#)
- [1.1.17. Quantile Regression](#)
- [1.1.18. Polynomial regression: extending linear models with basis functions](#)

### ▼ 1.2. Linear and Quadratic Discriminant Analysis

- [1.2.1. Dimensionality reduction using Linear Discriminant Analysis](#)
- [1.2.2. Mathematical formulation of the LDA and QDA classifiers](#)
- [1.2.3. Mathematical formulation of LDA dimensionality reduction](#)
- [1.2.4. Shrinkage and Covariance Estimator](#)
- [1.2.5. Estimation algorithms](#)

### - 1.3. Kernel ridge regression

### ▼ 1.4. Support Vector Machines

- [1.4.1. Classification](#)
- [1.4.2. Regression](#)
- [1.4.3. Density estimation, novelty detection](#)
- [1.4.4. Complexity](#)
- [1.4.5. Tips on Practical Use](#)
- [1.4.6. Kernel functions](#)
- [1.4.7. Mathematical formulation](#)
- [1.4.8. Implementation details](#)

### ▼ 1.5. Stochastic Gradient Descent

- [1.5.1. Classification](#)
- [1.5.2. Regression](#)
- [1.5.3. Online One-Class SVM](#)
- [1.5.4. Stochastic Gradient Descent for sparse data](#)
- [1.5.5. Complexity](#)
- [1.5.6. Stopping criterion](#)
- [1.5.7. Tips on Practical Use](#)
- [1.5.8. Mathematical formulation](#)
- [1.5.9. Implementation details](#)

### ▼ 1.6. Nearest Neighbors

Toggle Menu [unsupervised Nearest Neighbors](#)

- [1.6.2. Nearest Neighbors Classification](#)
- [1.6.3. Nearest Neighbors Regression](#)
- [1.6.4. Nearest Neighbor Algorithms](#)
- [1.6.5. Nearest Centroid Classifier](#)
- [1.6.6. Nearest Neighbors Transformer](#)
- [1.6.7. Neighborhood Components Analysis](#)
- ▼ [1.7. Gaussian Processes](#)
  - [1.7.1. Gaussian Process Regression \(GPR\)](#)
  - [1.7.2. GPR examples](#)
  - [1.7.3. Gaussian Process Classification \(GPC\)](#)
  - [1.7.4. GPC examples](#)
  - [1.7.5. Kernels for Gaussian Processes](#)
- ▼ [1.8. Cross decomposition](#)
  - [1.8.1. PLSCanonical](#)
  - [1.8.2. PLSSVD](#)
  - [1.8.3. PLSRegression](#)
  - [1.8.4. Canonical Correlation Analysis](#)
- ▼ [1.9. Naive Bayes](#)
  - [1.9.1. Gaussian Naive Bayes](#)
  - [1.9.2. Multinomial Naive Bayes](#)
  - [1.9.3. Complement Naive Bayes](#)
  - [1.9.4. Bernoulli Naive Bayes](#)
  - [1.9.5. Categorical Naive Bayes](#)
  - [1.9.6. Out-of-core naive Bayes model fitting](#)
- ▼ [1.10. Decision Trees](#)
  - [1.10.1. Classification](#)
  - [1.10.2. Regression](#)
  - [1.10.3. Multi-output problems](#)
  - [1.10.4. Complexity](#)
  - [1.10.5. Tips on practical use](#)
  - [1.10.6. Tree algorithms: ID3, C4.5, C5.0 and CART](#)
  - [1.10.7. Mathematical formulation](#)
  - [1.10.8. Minimal Cost-Complexity Pruning](#)
- ▼ [1.11. Ensemble methods](#)
  - [1.11.1. Bagging meta-estimator](#)
  - [1.11.2. Forests of randomized trees](#)
  - [1.11.3. AdaBoost](#)
  - [1.11.4. Gradient Tree Boosting](#)
  - [1.11.5. Histogram-Based Gradient Boosting](#)
  - [1.11.6. Voting Classifier](#)
  - [1.11.7. Voting Regressor](#)
  - [1.11.8. Stacked generalization](#)
- ▼ [1.12. Multiclass and multioutput algorithms](#)
  - [1.12.1. Multiclass classification](#)
  - [1.12.2. Multilabel classification](#)
  - [1.12.3. Multiclass-multioutput classification](#)
  - [1.12.4. Multioutput regression](#)
- ▼ [1.13. Feature selection](#)
  - [1.13.1. Removing features with low variance](#)
  - [1.13.2. Univariate feature selection](#)
  - [1.13.3. Recursive feature elimination](#)

- [1.13.5. Sequential Feature Selection](#)
- [1.13.6. Feature selection as part of a pipeline](#)
- ▼ [1.14. Semi-supervised learning](#)
  - [1.14.1. Self Training](#)
  - [1.14.2. Label Propagation](#)
- [1.15. Isotonic regression](#)
- ▼ [1.16. Probability calibration](#)
  - [1.16.1. Calibration curves](#)
  - [1.16.2. Calibrating a classifier](#)
  - [1.16.3. Usage](#)
- ▼ [1.17. Neural network models \(supervised\)](#)
  - [1.17.1. Multi-layer Perceptron](#)
  - [1.17.2. Classification](#)
  - [1.17.3. Regression](#)
  - [1.17.4. Regularization](#)
  - [1.17.5. Algorithms](#)
  - [1.17.6. Complexity](#)
  - [1.17.7. Mathematical formulation](#)
  - [1.17.8. Tips on Practical Use](#)
  - [1.17.9. More control with warm start](#)

## 2. Unsupervised learning

- ▼ [2.1. Gaussian mixture models](#)
  - [2.1.1. Gaussian Mixture](#)
  - [2.1.2. Variational Bayesian Gaussian Mixture](#)
- ▼ [2.2. Manifold learning](#)
  - [2.2.1. Introduction](#)
  - [2.2.2. Isomap](#)
  - [2.2.3. Locally Linear Embedding](#)
  - [2.2.4. Modified Locally Linear Embedding](#)
  - [2.2.5. Hessian Eigenmapping](#)
  - [2.2.6. Spectral Embedding](#)
  - [2.2.7. Local Tangent Space Alignment](#)
  - [2.2.8. Multi-dimensional Scaling \(MDS\)](#)
  - [2.2.9. t-distributed Stochastic Neighbor Embedding \(t-SNE\)](#)
  - [2.2.10. Tips on practical use](#)
- ▼ [2.3. Clustering](#)
  - [2.3.1. Overview of clustering methods](#)
  - [2.3.2. K-means](#)
  - [2.3.3. Affinity Propagation](#)
  - [2.3.4. Mean Shift](#)
  - [2.3.5. Spectral clustering](#)
  - [2.3.6. Hierarchical clustering](#)
  - [2.3.7. DBSCAN](#)
  - [2.3.8. OPTICS](#)
  - [2.3.9. BIRCH](#)
  - [2.3.10. Clustering performance evaluation](#)
- ▼ [2.4. Biclustering](#)
  - [2.4.1. Spectral Co-Clustering](#)
  - [2.4.2. Spectral Biclustering](#)

- ▼ [2.5. Decomposing signals in components \(matrix factorization problems\)](#)
  - [2.5.1. Principal component analysis \(PCA\)](#)
  - [2.5.2. Kernel Principal Component Analysis \(kPCA\)](#)
  - [2.5.3. Truncated singular value decomposition and latent semantic analysis](#)
  - [2.5.4. Dictionary Learning](#)
  - [2.5.5. Factor Analysis](#)
  - [2.5.6. Independent component analysis \(ICA\)](#)
  - [2.5.7. Non-negative matrix factorization \(NMF or NNMF\)](#)
  - [2.5.8. Latent Dirichlet Allocation \(LDA\)](#)
- ▼ [2.6. Covariance estimation](#)
  - [2.6.1. Empirical covariance](#)
  - [2.6.2. Shrunk Covariance](#)
  - [2.6.3. Sparse inverse covariance](#)
  - [2.6.4. Robust Covariance Estimation](#)
- ▼ [2.7. Novelty and Outlier Detection](#)
  - [2.7.1. Overview of outlier detection methods](#)
  - [2.7.2. Novelty Detection](#)
  - [2.7.3. Outlier Detection](#)
  - [2.7.4. Novelty detection with Local Outlier Factor](#)
- ▼ [2.8. Density Estimation](#)
  - [2.8.1. Density Estimation: Histograms](#)
  - [2.8.2. Kernel Density Estimation](#)
- ▼ [2.9. Neural network models \(unsupervised\)](#)
  - [2.9.1. Restricted Boltzmann machines](#)

### **3. Model selection and evaluation**

- ▼ [3.1. Cross-validation: evaluating estimator performance](#)
  - [3.1.1. Computing cross-validated metrics](#)
  - [3.1.2. Cross validation iterators](#)
  - [3.1.3. A note on shuffling](#)
  - [3.1.4. Cross validation and model selection](#)
  - [3.1.5. Permutation test score](#)
- ▼ [3.2. Tuning the hyper-parameters of an estimator](#)
  - [3.2.1. Exhaustive Grid Search](#)
  - [3.2.2. Randomized Parameter Optimization](#)
  - [3.2.3. Searching for optimal parameters with successive halving](#)
  - [3.2.4. Tips for parameter search](#)
  - [3.2.5. Alternatives to brute force parameter search](#)
- ▼ [3.3. Metrics and scoring: quantifying the quality of predictions](#)
  - [3.3.1. The \*\*scoring\*\* parameter: defining model evaluation rules](#)
  - [3.3.2. Classification metrics](#)
  - [3.3.3. Multilabel ranking metrics](#)
  - [3.3.4. Regression metrics](#)
  - [3.3.5. Clustering metrics](#)
  - [3.3.6. Dummy estimators](#)
- ▼ [3.4. Validation curves: plotting scores to evaluate models](#)
  - [3.4.1. Validation curve](#)
  - [3.4.2. Learning curve](#)

- ▼ [4.1. Partial Dependence and Individual Conditional Expectation plots](#)
  - [4.1.1. Partial dependence plots](#)
  - [4.1.2. Individual conditional expectation \(ICE\)\\_plot](#)
  - [4.1.3. Mathematical Definition](#)
  - [4.1.4. Computation methods](#)
- ▼ [4.2. Permutation feature importance](#)
  - [4.2.1. Outline of the permutation importance algorithm](#)
  - [4.2.2. Relation to impurity-based importance in trees](#)
  - [4.2.3. Misleading values on strongly correlated features](#)

## 5. Visualizations

- ▼ [5.1. Available Plotting Utilities](#)
  - [5.1.1. Functions](#)
  - [5.1.2. Display Objects](#)

## 6. Dataset transformations

- ▼ [6.1. Pipelines and composite estimators](#)
  - [6.1.1. Pipeline: chaining estimators](#)
  - [6.1.2. Transforming target in regression](#)
  - [6.1.3. FeatureUnion: composite feature spaces](#)
  - [6.1.4. ColumnTransformer for heterogeneous data](#)
  - [6.1.5. Visualizing Composite Estimators](#)
- ▼ [6.2. Feature extraction](#)
  - [6.2.1. Loading features from dicts](#)
  - [6.2.2. Feature hashing](#)
  - [6.2.3. Text feature extraction](#)
  - [6.2.4. Image feature extraction](#)
- ▼ [6.3. Preprocessing data](#)
  - [6.3.1. Standardization, or mean removal and variance scaling](#)
  - [6.3.2. Non-linear transformation](#)
  - [6.3.3. Normalization](#)
  - [6.3.4. Encoding categorical features](#)
  - [6.3.5. Discretization](#)
  - [6.3.6. Imputation of missing values](#)
  - [6.3.7. Generating polynomial features](#)
  - [6.3.8. Custom transformers](#)
- ▼ [6.4. Imputation of missing values](#)
  - [6.4.1. Univariate vs. Multivariate Imputation](#)
  - [6.4.2. Univariate feature imputation](#)
  - [6.4.3. Multivariate feature imputation](#)
  - [6.4.4. References](#)
  - [6.4.5. Nearest neighbors imputation](#)
  - [6.4.6. Marking imputed values](#)
- ▼ [6.5. Unsupervised dimensionality reduction](#)
  - [6.5.1. PCA: principal component analysis](#)
  - [6.5.2. Random projections](#)
  - [6.5.3. Feature agglomeration](#)
- ▼ [6.6. Random Projection](#)
  - [6.6.1. The Johnson-Lindenstrauss lemma](#)
  - [Gaussian random projection](#)

- [6.6.3. Sparse random projection](#)

## ▼ [6.7. Kernel Approximation](#)

- [6.7.1. Nystroem Method for Kernel Approximation](#)
- [6.7.2. Radial Basis Function Kernel](#)
- [6.7.3. Additive Chi Squared Kernel](#)
- [6.7.4. Skewed Chi Squared Kernel](#)
- [6.7.5. Polynomial Kernel Approximation via Tensor Sketch](#)
- [6.7.6. Mathematical Details](#)

## ▼ [6.8. Pairwise metrics, Affinities and Kernels](#)

- [6.8.1. Cosine similarity](#)
- [6.8.2. Linear kernel](#)
- [6.8.3. Polynomial kernel](#)
- [6.8.4. Sigmoid kernel](#)
- [6.8.5. RBF kernel](#)
- [6.8.6. Laplacian kernel](#)
- [6.8.7. Chi-squared kernel](#)

## ▼ [6.9. Transforming the prediction target \(y\)](#)

- [6.9.1. Label binarization](#)
- [6.9.2. Label encoding](#)

# 7. Dataset loading utilities

## ▼ [7.1. Toy datasets](#)

- [7.1.1. Boston house prices dataset](#)
- [7.1.2. Iris plants dataset](#)
- [7.1.3. Diabetes dataset](#)
- [7.1.4. Optical recognition of handwritten digits dataset](#)
- [7.1.5. Linnerrud dataset](#)
- [7.1.6. Wine recognition dataset](#)
- [7.1.7. Breast cancer wisconsin \(diagnostic\) dataset](#)

## ▼ [7.2. Real world datasets](#)

- [7.2.1. The Olivetti faces dataset](#)
- [7.2.2. The 20 newsgroups text dataset](#)
- [7.2.3. The Labeled Faces in the Wild face recognition dataset](#)
- [7.2.4. Forest covertypes](#)
- [7.2.5. RCV1 dataset](#)
- [7.2.6. Kddcup 99 dataset](#)
- [7.2.7. California Housing dataset](#)

## ▼ [7.3. Generated datasets](#)

- [7.3.1. Generators for classification and clustering](#)
- [7.3.2. Generators for regression](#)
- [7.3.3. Generators for manifold learning](#)
- [7.3.4. Generators for decomposition](#)

## ▼ [7.4. Loading other datasets](#)

- [7.4.1. Sample images](#)
- [7.4.2. Datasets in svmlight / libsvm format](#)
- [7.4.3. Downloading datasets from the openml.org repository](#)
- [7.4.4. Loading from external datasets](#)

# 8. Computing with scikit-learn

## ▼ [8.1. Strategies to scale computationally: bigger data](#)

- [8.1.1. Scaling with instances using out-of-core learning](#)

## ▼ [8.2. Computational Performance](#)

- [8.2.1. Prediction Latency](#)
- [8.2.2. Prediction Throughput](#)
- [8.2.3. Tips and Tricks](#)

## ▼ [8.3. Parallelism, resource management, and configuration](#)

- [8.3.1. Parallelism](#)
- [8.3.2. Configuration switches](#)

## [9. Model persistence](#)

### ▼ [9.1. Python specific serialization](#)

- [9.1.1. Security & maintainability limitations](#)

### - [9.2. Interoperable formats](#)

## [10. Common pitfalls and recommended practices](#)

### - [10.1. Inconsistent preprocessing](#)

### ▼ [10.2. Data leakage](#)

- [10.2.1. Data leakage during pre-processing](#)
- [10.2.2. How to avoid data leakage](#)

### ▼ [10.3. Controlling randomness](#)

- [10.3.1. Using `None` or `RandomState` instances, and repeated calls to `fit` and `split`](#)
- [10.3.2. Common pitfalls and subtleties](#)
- [10.3.3. General recommendations](#)