

- 1. Supervised learning
  - 1.1. Generalized Linear Models
    - 1.1.1. Ordinary Least Squares
      - 1.1.1.1. Ordinary Least Squares Complexity
    - 1.1.2. Ridge Regression
      - 1.1.2.1. Ridge Complexity
      - 1.1.2.2. Setting the regularization parameter: generalized Cross-Validation
    - 1.1.3. Lasso
      - 1.1.3.1. Setting regularization parameter
        - 1.1.3.1.1. Using cross-validation
        - 1.1.3.1.2. Information-criteria based model selection
        - 1.1.3.1.3. Comparison with the regularization parameter of SVM
      - 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
        - 1.1.8.1. Mathematical formulation
      - 1.1.9. Orthogonal Matching Pursuit (OMP)
      - 1.1.10. Bayesian Regression
        - 1.1.10.1. Bayesian Ridge Regression
        - 1.1.10.2. Automatic Relevance Determination - ARD
      - 1.1.11. Logistic regression
      - 1.1.12. Stochastic Gradient Descent - SGD
      - 1.1.13. Perceptron
      - 1.1.14. Passive Aggressive Algorithms
      - 1.1.15. Robustness regression: outliers and modeling errors
        - 1.1.15.1. Different scenario and useful concepts
        - 1.1.15.2. RANSAC: Random Sample Consensus
          - 1.1.15.2.1. Details of the algorithm
      - 1.1.15.3. Theil-Sen estimator: generalized-median-based estimator
        - 1.1.15.3.1. Theoretical considerations
      - 1.1.15.4. Huber Regression
      - 1.1.15.5. Notes
    - 1.1.16. 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
    - 1.2.5. Estimation algorithms
  - 1.3. Kernel ridge regression
  - 1.4. Support Vector Machines
    - 1.4.1. Classification
      - 1.4.1.1. Multi-class classification
      - 1.4.1.2. Scores and probabilities
      - 1.4.1.3. Unbalanced problems
    - 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
      - 1.4.6.1. Custom Kernels
        - 1.4.6.1.1. Using Python functions as kernels
        - 1.4.6.1.2. Using the Gram matrix
        - 1.4.6.1.3. Parameters of the RBF Kernel

- 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
  - 1.10.7.1. Classification criteria
  - 1.10.7.2. Regression criteria
- 1.11. Ensemble methods
  - 1.11.1. Bagging meta-estimator
  - 1.11.2. Forests of randomized trees
    - 1.11.2.1. Random Forests
    - 1.11.2.2. Extremely Randomized Trees
    - 1.11.2.3. Parameters
    - 1.11.2.4. Parallelization
    - 1.11.2.5. Feature importance evaluation
    - 1.11.2.6. Totally Random Trees Embedding
  - 1.11.3. AdaBoost
    - 1.11.3.1. Usage
  - 1.11.4. Gradient Tree Boosting
    - 1.11.4.1. Classification
    - 1.11.4.2. Regression
    - 1.11.4.3. Fitting additional weak-learners
    - 1.11.4.4. Controlling the tree size
    - 1.11.4.5. Mathematical formulation
    - 1.11.4.5.1. Loss Functions
  - 1.11.4.6. Regularization
    - 1.11.4.6.1. Shrinkage
    - 1.11.4.6.2. Subsampling
  - 1.11.4.7. Interpretation
    - 1.11.4.7.1. Feature importance
    - 1.11.4.7.2. Partial dependence
  - 1.11.5. Voting Classifier
    - 1.11.5.1. Majority Class Labels (Majority/Hard Voting)
      - 1.11.5.1.1. Usage
    - 1.11.5.2. Weighted Average Probabilities (Soft Voting)
    - 1.11.5.3. Using the VotingClassifier with GridSearch
      - 1.11.5.3.1. Usage
  - 1.12. Multiclass and multilabel algorithms
    - 1.12.1. Multilabel classification format
    - 1.12.2. One-Vs-The-Rest
      - 1.12.2.1. Multiclass learning
      - 1.12.2.2. Multilabel learning
    - 1.12.3. One-Vs-One
      - 1.12.3.1. Multiclass learning
  - 1.12.4. Error-Correcting Output-Codes
    - 1.12.4.1. Multiclass learning
  - 1.12.5. Multioutput regression
  - 1.12.6. Multioutput classification
  - 1.12.7. Classifier Chain
  - 1.12.8. Regressor Chain
  - 1.13. Feature selection
    - 1.13.1. Removing features with low variance
    - 1.13.2. Univariate feature selection
    - 1.13.3. Recursive feature elimination
    - 1.13.4. Feature selection using SelectFromModel

- 1.4.7. Mathematical formulation
  - 1.4.7.1. SVC
  - 1.4.7.2. NuSVC
  - 1.4.7.3. SVR
- 1.4.8. Implementation details
- 1.5. Stochastic Gradient Descent
  - 1.5.1. Classification
  - 1.5.2. Regression
  - 1.5.3. Stochastic Gradient Descent for sparse data
  - 1.5.4. Complexity
  - 1.5.5. Stopping criterion
  - 1.5.6. Tips on Practical Use
  - 1.5.7. Mathematical formulation
    - 1.5.7.1. SGD
  - 1.5.8. Implementation details
- 1.6. Nearest Neighbors
  - 1.6.1. Unsupervised Nearest Neighbors
    - 1.6.1.1. Finding the Nearest Neighbors
    - 1.6.1.2. KDTree and BallTree Classes
  - 1.6.2. Nearest Neighbors Classification
    - 1.6.3. Nearest Neighbors Regression
    - 1.6.4. Nearest Neighbor Algorithms
      - 1.6.4.1. Brute Force
      - 1.6.4.2. K-D Tree
      - 1.6.4.3. Ball Tree
      - 1.6.4.4. Choice of Nearest Neighbors Algorithm
      - 1.6.4.5. Effect of leaf\_size
    - 1.6.5. Nearest Centroid Classifier
      - 1.6.5.1. Nearest Shrunken Centroid
  - 1.7. Gaussian Processes
    - 1.7.1. Gaussian Process Regression (GPR)
    - 1.7.2. GPR examples
      - 1.7.2.1. GPR with noise-level estimation
      - 1.7.2.2. Comparison of GPR and Kernel Ridge Regression
      - 1.7.2.3. GPR on Mauna Loa CO2 data
    - 1.7.3. Gaussian Process Classification (GPC)
    - 1.7.4. GPC examples
      - 1.7.4.1. Probabilistic predictions with GPC
      - 1.7.4.2. Illustration of GPC on the XOR dataset
      - 1.7.4.3. Gaussian process classification (GPC) on iris dataset
    - 1.7.5. Kernels for Gaussian Processes
      - 1.7.5.1. Gaussian Process Kernel API
      - 1.7.5.2. Basic kernels
      - 1.7.5.3. Kernel operators
      - 1.7.5.4. Radial-basis function (RBF) kernel
      - 1.7.5.5. Matérn kernel
      - 1.7.5.6. Rational quadratic kernel
      - 1.7.5.7. Exp-Sine-Squared kernel
      - 1.7.5.8. Dot-Product kernel
      - 1.7.5.9. References
  - 1.8. Cross decomposition
    - 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. Out-of-core naive Bayes model fitting
  - 1.10. Decision Trees
    - 1.13.4.1. L1-based feature selection
    - 1.13.4.2. Tree-based feature selection
    - 1.13.5. Feature selection as part of a pipeline
  - 1.14. Semi-Supervised
    - 1.14.1. Label Propagation
    - 1.15. Isotonic regression
    - 1.16. Probability calibration
    - 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
        - 2.1.1.1. Pros and cons of class GaussianMixture
          - 2.1.1.1.1. Pros
          - 2.1.1.1.2. Cons
        - 2.1.1.2. Selecting the number of components in a classical Gaussian Mixture Model
        - 2.1.1.3. Estimation algorithm Expectation-maximization
      - 2.1.2. Variational Bayesian Gaussian Mixture
        - 2.1.2.1. Estimation algorithm: variational inference
        - 2.1.2.2. Pros and cons of variational inference with BayesianGaussianMixture
          - 2.1.2.2.1. Pros
          - 2.1.2.2.2. Cons
        - 2.1.2.3. The Dirichlet Process
      - 2.2. Manifold learning
        - 2.2.1. Introduction
        - 2.2.2. Isomap
          - 2.2.2.1. Complexity
        - 2.2.3. Locally Linear Embedding
          - 2.2.3.1. Complexity
        - 2.2.4. Modified Locally Linear Embedding
          - 2.2.4.1. Complexity
        - 2.2.5. Hessian Eigenmapping
          - 2.2.5.1. Complexity
        - 2.2.6. Spectral Embedding
          - 2.2.6.1. Complexity
        - 2.2.7. Local Tangent Space Alignment
          - 2.2.7.1. Complexity
        - 2.2.8. Multi-dimensional Scaling (MDS)
          - 2.2.8.1. Metric MDS
          - 2.2.8.2. Nonmetric MDS
        - 2.2.9. t-distributed Stochastic Neighbor Embedding (t-SNE)
          - 2.2.9.1. Optimizing t-SNE
          - 2.2.9.2. Barnes-Hut t-SNE
      - 2.2.10. Tips on practical use

- 2.3. Clustering
  - 2.3.1. Overview of clustering methods
  - 2.3.2. K-means
    - 2.3.2.1. Mini Batch K-Means
  - 2.3.3. Affinity Propagation
  - 2.3.4. Mean Shift
  - 2.3.5. Spectral clustering
    - 2.3.5.1. Different label assignment strategies
    - 2.3.5.2. Spectral Clustering Graphs
  - 2.3.6. Hierarchical clustering
    - 2.3.6.1. Different linkage type: Ward, complete, average, and single linkage
    - 2.3.6.2. Adding connectivity constraints
    - 2.3.6.3. Varying the metric
  - 2.3.7. DBSCAN
  - 2.3.8. Birch
  - 2.3.9. Clustering performance evaluation
    - 2.3.9.1. Adjusted Rand index
      - 2.3.9.1.1. Advantages
      - 2.3.9.1.2. Drawbacks
      - 2.3.9.1.3. Mathematical formulation
  - 2.3.9.2. Mutual Information based scores
    - 2.3.9.2.1. Advantages
    - 2.3.9.2.2. Drawbacks
    - 2.3.9.2.3. Mathematical formulation
  - 2.3.9.3. Homogeneity, completeness and V-measure
    - 2.3.9.3.1. Advantages
    - 2.3.9.3.2. Drawbacks
    - 2.3.9.3.3. Mathematical formulation
  - 2.3.9.4. Fowlkes-Mallows scores
    - 2.3.9.4.1. Advantages
    - 2.3.9.4.2. Drawbacks
  - 2.3.9.5. Silhouette Coefficient
    - 2.3.9.5.1. Advantages
    - 2.3.9.5.2. Drawbacks
  - 2.3.9.6. Calinski-Harabaz Index
    - 2.3.9.6.1. Advantages
    - 2.3.9.6.2. Drawbacks
  - 2.3.9.7. Davies-Bouldin Index
    - 2.3.9.7.1. Advantages
    - 2.3.9.7.2. Drawbacks
  - 2.3.9.8. Contingency Matrix
    - 2.3.9.8.1. Advantages
    - 2.3.9.8.2. Drawbacks
- 2.4. Biclustering
  - 2.4.1. Spectral Co-Clustering
    - 2.4.1.1. Mathematical formulation
  - 2.4.2. Spectral Biclustering
    - 2.4.2.1. Mathematical formulation
  - 2.4.3. Biclustering evaluation
- 2.5. Decomposing signals in components (matrix factorization problems)
  - 2.5.1. Principal component analysis (PCA)
    - 2.5.1.1. Exact PCA and probabilistic interpretation
    - 2.5.1.2. Incremental PCA
    - 2.5.1.3. PCA using randomized SVD
    - 2.5.1.4. Kernel PCA

- 2.5.1.5. Sparse principal components analysis (SparsePCA and MiniBatchSparsePCA)
- 2.5.2. Truncated singular value decomposition and latent semantic analysis
- 2.5.3. Dictionary learning
  - 2.5.3.1. Sparse coding with a precomputed dictionary
  - 2.5.3.2. Generic dictionary learning
  - 2.5.3.3. Mini-batch dictionary learning
- 2.5.4. Factor Analysis
- 2.5.5. Independent component analysis (ICA)
- 2.5.6. Non-negative matrix factorization (NMF or NNMF)
  - 2.5.6.1. NMF with the Frobenius norm
  - 2.5.6.2. NMF with a beta-divergence
- 2.5.7. Latent Dirichlet Allocation (LDA)
- 2.6. Covariance estimation
  - 2.6.1. Empirical covariance
  - 2.6.2. Shrunk Covariance
    - 2.6.2.1. Basic shrinkage
    - 2.6.2.2. Ledoit-Wolf shrinkage
    - 2.6.2.3. Oracle Approximating Shrinkage
- 2.6.3. Sparse inverse covariance
- 2.6.4. Robust Covariance Estimation
  - 2.6.4.1. Minimum Covariance Determinant
- 2.7. Novelty and Outlier Detection
  - 2.7.1. Overview of outlier detection methods
  - 2.7.2. Novelty Detection
  - 2.7.3. Outlier Detection
    - 2.7.3.1. Fitting an elliptic envelope
    - 2.7.3.2. Isolation Forest
    - 2.7.3.3. Local Outlier Factor
- 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
    - 2.9.1.1. Graphical model and parametrization
    - 2.9.1.2. Bernoulli Restricted Boltzmann machines
    - 2.9.1.3. Stochastic Maximum Likelihood learning
- 3. Model selection and evaluation
  - 3.1. Cross-validation: evaluating estimator performance
    - 3.1.1. Computing cross-validated metrics
      - 3.1.1.1. The cross\_validate function and multiple metric evaluation
      - 3.1.1.2. Obtaining predictions by cross-validation
    - 3.1.2. Cross validation iterators
      - 3.1.2.1. Cross-validation iterators for i.i.d. data
        - 3.1.2.1.1. K-fold
          - 3.1.2.1.2. Repeated K-Fold
          - 3.1.2.1.3. Leave One Out (LOO)
          - 3.1.2.1.4. Leave P Out (LPO)
          - 3.1.2.1.5. Random permutations cross-validation a.k.a. Shuffle & Split
      - 3.1.2.2. Cross-validation iterators with stratification based on class labels.
        - 3.1.2.2.1. Stratified k-fold
        - 3.1.2.2.2. Stratified Shuffle Split
      - 3.1.2.3. Cross-validation iterators for grouped data.
        - 3.1.2.3.1. Group k-fold
        - 3.1.2.3.2. Leave One Group Out
        - 3.1.2.3.3. Leave P Groups Out

3.1.2.3.4. Group Shuffle Split

3.1.2.4. Predefined Fold-Splits / Validation-Sets

3.1.2.5. Cross validation of time series data

3.1.2.5.1. Time Series Split

3.1.3. A note on shuffling

3.1.4. Cross validation and model selection

3.2. Tuning the hyper-parameters of an estimator

3.2.1. Exhaustive Grid Search

3.2.2. Randomized Parameter Optimization

3.2.3. Tips for parameter search

3.2.3.1. Specifying an objective metric

3.2.3.2. Specifying multiple metrics for evaluation

3.2.3.3. Composite estimators and parameter spaces

3.2.3.4. Model selection: development and evaluation

3.2.3.5. Parallelism

3.2.3.6. Robustness to failure

3.2.4. Alternatives to brute force parameter search

3.2.4.1. Model specific cross-validation

3.2.4.1.1. sklearn.linear\_model.ElasticNetCV

3.2.4.1.2. sklearn.linear\_model.LarsCV

3.2.4.1.3. sklearn.linear\_model.LassoCV

3.2.4.1.3.1. Examples using sklearn.linear\_model.LassoCV

3.2.4.1.4. sklearn.linear\_model.LassoLarsCV

3.2.4.1.4.1. Examples using sklearn.linear\_model.LassoLarsCV

3.2.4.1.5. sklearn.linear\_model.LogisticRegressionCV

3.2.4.1.6. sklearn.linear\_model.MultiTaskElasticNetCV

3.2.4.1.7. sklearn.linear\_model.MultiTaskLassoCV

3.2.4.1.8. sklearn.linear\_model.OrthogonalMatchingPursuitCV

3.2.4.1.8.1. Examples using sklearn.linear\_model.OrthogonalMatchingPursuitCV

3.2.4.1.9. sklearn.linear\_model.RidgeCV

3.2.4.1.9.1. Examples using sklearn.linear\_model.RidgeCV

3.2.4.1.10. sklearn.linear\_model.RidgeClassifierCV

3.2.4.2. Information Criterion

3.2.4.2.1. sklearn.linear\_model.LassoLarsIC

3.2.4.2.1.1. Examples using sklearn.linear\_model.LassoLarsIC

3.2.4.3. Out of Bag Estimates

3.2.4.3.1. sklearn.ensemble.RandomForestClassifier

3.2.4.3.1.1. Examples using sklearn.ensemble.RandomForestClassifier

3.2.4.3.2. sklearn.ensemble.RandomForestRegressor

3.2.4.3.2.1. Examples using sklearn.ensemble.RandomForestRegressor

3.2.4.3.3. sklearn.ensemble.ExtraTreesClassifier

3.2.4.3.3.1. Examples using sklearn.ensemble.ExtraTreesClassifier

3.2.4.3.4. sklearn.ensemble.ExtraTreesRegressor

3.2.4.3.4.1. Examples using sklearn.ensemble.ExtraTreesRegressor

3.2.4.3.5. sklearn.ensemble.GradientBoostingClassifier

3.2.4.3.5.1. Examples using sklearn.ensemble.GradientBoostingClassifier

3.2.4.3.6. sklearn.ensemble.GradientBoostingRegressor

3.2.4.3.6.1. Examples using sklearn.ensemble.GradientBoostingRegressor

3.3. Model evaluation: quantifying the quality of predictions

3.3.1. The scoring parameter: defining model evaluation rules

3.3.1.1. Common cases: predefined values

3.3.1.2. Defining your scoring strategy from metric functions

3.3.1.3. Implementing your own scoring object

3.3.1.4. Using multiple metric evaluation

3.3.2. Classification metrics

3.3.2.1. From binary to multiclass and multilabel

3.3.2.2. Accuracy score

3.3.2.3. Balanced accuracy score

3.3.2.4. Cohen's kappa

3.3.2.5. Confusion matrix

3.3.2.6. Classification report

3.3.2.7. Hamming loss

3.3.2.8. Jaccard similarity coefficient score

3.3.2.9. Precision, recall and F-measures

3.3.2.9.1. Binary classification

3.3.2.9.2. Multiclass and multilabel classification

3.3.2.10. Hinge loss

3.3.2.11. Log loss

3.3.2.12. Matthews correlation coefficient

3.3.2.13. Receiver operating characteristic (ROC)

3.3.2.14. Zero one loss

3.3.2.15. Brier score loss

3.3.3. Multilabel ranking metrics

3.3.3.1. Coverage error

3.3.3.2. Label ranking average precision

3.3.3.3. Ranking loss

3.3.4. Regression metrics

3.3.4.1. Explained variance score

3.3.4.2. Mean absolute error

3.3.4.3. Mean squared error

3.3.4.4. Mean squared logarithmic error

3.3.4.5. Median absolute error

3.3.4.6. R<sup>2</sup> score, the coefficient of determination

3.3.5. Clustering metrics

3.3.6. Dummy estimators

3.4. Model persistence

3.4.1. Persistence example

3.4.2. Security & maintainability limitations

3.5. Validation curves: plotting scores to evaluate models

3.5.1. Validation curve

3.5.2. Learning curve

4. Dataset transformations

4.1. Pipelines and composite estimators

4.1.1. Pipeline: chaining estimators

4.1.1.1. Usage

4.1.1.2. Notes

4.1.1.3. Caching transformers: avoid repeated computation

4.1.2. Transforming target in regression

4.1.3. FeatureUnion: composite feature spaces

4.1.3.1. Usage

4.1.4. ColumnTransformer for heterogeneous data

4.2. Feature extraction

4.2.1. Loading features from dicts

4.2.2. Feature hashing

4.2.2.1. Implementation details

4.2.3. Text feature extraction

4.2.3.1. The Bag of Words representation

4.2.3.2. Sparsity

4.2.3.3. Common Vectorizer usage

4.2.3.3.1. Using stop words

- 4.2.3.4. Tf-idf term weighting
- 4.2.3.5. Decoding text files
- 4.2.3.6. Applications and examples
- 4.2.3.7. Limitations of the Bag of Words representation
- 4.2.3.8. Vectorizing a large text corpus with the hashing trick
- 4.2.3.9. Performing out-of-core scaling with HashingVectorizer
- 4.2.3.10. Customizing the vectorizer classes
- 4.2.4. Image feature extraction
- 4.2.4.1. Patch extraction
- 4.2.4.2. Connectivity graph of an image
- 4.3. Preprocessing data
- 4.3.1. Standardization, or mean removal and variance scaling
- 4.3.1.1. Scaling features to a range
- 4.3.1.2. Scaling sparse data
- 4.3.1.3. Scaling data with outliers
- 4.3.1.4. Centering kernel matrices
- 4.3.2. Non-linear transformation
- 4.3.2.1. Mapping to a Uniform distribution
- 4.3.2.2. Mapping to a Gaussian distribution
- 4.3.3. Normalization
- 4.3.4. Encoding categorical features
- 4.3.5. Discretization
- 4.3.5.1. K-bins discretization
- 4.3.5.2. Feature binarization
- 4.3.6. Imputation of missing values
- 4.3.7. Generating polynomial features
- 4.3.8. Custom transformers
- 4.4. Imputation of missing values
- 4.4.1. Marking imputed values
- 4.5. Unsupervised dimensionality reduction
- 4.5.1. PCA: principal component analysis
- 4.5.2. Random projections
- 4.5.3. Feature agglomeration
- 4.6. Random Projection
- 4.6.1. The Johnson-Lindenstrauss lemma
- 4.6.2. Gaussian random projection
- 4.6.3. Sparse random projection
- 4.7. Kernel Approximation
- 4.7.1. Nystroem Method for Kernel Approximation
- 4.7.2. Radial Basis Function Kernel
- 4.7.3. Additive Chi Squared Kernel
- 4.7.4. Skewed Chi Squared Kernel
- 4.7.5. Mathematical Details
- 4.8. Pairwise metrics, Affinities and Kernels
- 4.8.1. Cosine similarity
- 4.8.2. Linear kernel
- 4.8.3. Polynomial kernel
- 4.8.4. Sigmoid kernel
- 4.8.5. RBF kernel
- 4.8.6. Laplacian kernel
- 4.8.7. Chi-squared kernel
- 4.9. Transforming the prediction target (y)
- 4.9.1. Label binarization
- 4.9.2. Label encoding
- 5. Dataset loading utilities
- 5.1. General dataset API

- 6.3. Parallelism, resource management, and configuration
- 6.3.1. Parallel and distributed computing
- 6.3.2. Configuration switches
- 6.3.2.1. Python runtime
- 6.3.2.2. Environment variables

- 5.2. Toy datasets
- 5.2.1. Boston house prices dataset
- 5.2.2. Iris plants dataset
- 5.2.3. Diabetes dataset
- 5.2.4. Optical recognition of handwritten digits dataset
- 5.2.5. Linnerrud dataset
- 5.2.6. Wine recognition dataset
- 5.2.7. Breast cancer wisconsin (diagnostic) dataset
- 5.3. Real world datasets
- 5.3.1. The Olivetti faces dataset
- 5.3.2. The 20 newsgroups text dataset
- 5.3.2.1. Usage
- 5.3.2.2. Converting text to vectors
- 5.3.2.3. Filtering text for more realistic training
- 5.3.3. The Labeled Faces in the Wild face recognition dataset
- 5.3.3.1. Usage
- 5.3.3.2. Examples
- 5.3.4. Forest covertypes
- 5.3.5. RCVI dataset
- 5.3.6. Kddcup 99 dataset
- 5.3.7. California Housing dataset
- 5.4. Generated datasets
- 5.4.1. Generators for classification and clustering
- 5.4.1.1. Single label
- 5.4.1.2. Multilabel
- 5.4.1.3. Biclustering
- 5.4.2. Generators for regression
- 5.4.3. Generators for manifold learning
- 5.4.4. Generators for decomposition
- 5.5. Loading other datasets
- 5.5.1. Sample images
- 5.5.2. Datasets in svmlight / libsvm format
- 5.5.3. Downloading datasets from the openml.org repository
- 5.5.3.1. Dataset Versions
- 5.5.4. Loading from external datasets
- 6. Computing with scikit-learn
- 6.1. Strategies to scale computationally: bigger data
- 6.1.1. Scaling with instances using out-of-core learning
- 6.1.1.1. Streaming instances
- 6.1.1.2. Extracting features
- 6.1.1.3. Incremental learning
- 6.1.1.4. Examples
- 6.1.1.5. Notes
- 6.2. Computational Performance
- 6.2.1. Prediction Latency
- 6.2.1.1. Bulk versus Atomic mode
- 6.2.1.2. Configuring Scikit-learn for reduced validation overhead
- 6.2.1.3. Influence of the Number of Features
- 6.2.1.4. Influence of the Input Data Representation
- 6.2.1.5. Influence of the Model Complexity
- 6.2.1.6. Feature Extraction Latency
- 6.2.2. Prediction Throughput
- 6.2.3. Tips and Tricks
- 6.2.3.1. Linear algebra libraries
- 6.2.3.2. Limiting Working Memory
- 6.2.3.3. Model Compression
- 6.2.3.4. Model Reshaping
- 6.2.3.5. Links