Machine Learning Methods

Weige Huang
Department of Economics,Temple University
(Not finished)

Contents

1 Methods for Regression **7**
- 1.1 Linear Regression Models and Least Squares . . . 7
- 1.2 Subset Selection . . . 7
  - 1.2.1 Best-Subset Selection . . . 7
  - 1.2.2 Forward- and Backward-Stepwise Selection . . . 7
  - 1.2.3 Forward-Stagewise Regression . . . 8
    - 1.2.3.1 Incremental Forward Stagewise Regression . . . 8
- 1.3 Shrinkage Methods . . . 8
  - 1.3.1 Ridge Regression . . . 8
  - 1.3.2 Lasso . . . 8
    - 1.3.2.1 Quantile Lasso . . . 9
    - 1.3.2.2 Lag/Ordered Lasso . . . 9
    - 1.3.2.3 The Group Lasso . . . 9
    - 1.3.2.4 Fused Lasso . . . 9
    - 1.3.2.5 Adaptive Lasso . . . 10
    - 1.3.2.6 Relaxed Lasso . . . 10
    - 1.3.2.7 Square-root Lasso . . . 10
  - 1.3.3 Elastic Net . . . 10
  - 1.3.4 Least Angle Regression . . . 11
  - 1.3.5 Dantzig Selector . . . 11
- 1.4 Methods Using Derived Input Directions . . . 11
  - 1.4.1 Principal Components Regression . . . 11
  - 1.4.2 Partial Least Squares . . . 12

2 Methods for Classification **13**
- 2.1 Linear Discriminant Analysis . . . 13
- 2.2 Quadratic Discriminant Analysis . . . 13
- 2.3 Diagonal Linear Discriminant Analysis . . . 13
- 2.4 Reduced rank linear discriminant analysis . . . 13
- 2.5 Generalized Linear Discriminant Analysis . . . 14
  - 2.5.1 Flexible Discriminant Analysis . . . 14
  - 2.5.2 Regularized Discriminant Analysis . . . 14
  - 2.5.3 Mixture Discriminant Analysis . . . 15
- 2.6 Logistic Regression . . . 15
  - 2.6.1 Regularized Logistic Regression . . . 15
  - 2.6.2 Nonparametric Logistic Regression . . . 16
  - 2.6.3 Additive Logistic Regression . . . 16
- 2.7 Naive Bayes Classifier . . . 16

3 Prototype Methods and Nearest-Neighbors **17**
- 3.1 K-means Clustering . . . 17
- 3.2 Learning Vector Quantization (LVQ) . . . 17
- 3.3 Gaussian Mixture Models (GMM) . . . 18
- 3.4 K-Nearest-Neighbor Classifiers . . . 18
- 3.5 Adaptive Nearest-Neighbor Methods . . . 19
- 3.6 Global Dimension Reduction for Nearest-Neighbors . . . 19

4 Decision Tree Learning **20**
4.1 Regression Trees . . . 20
4.2 Classification Trees . . . 20
4.3 Boosting Trees . . . 20
4.4 Bagging . . . 20
4.5 MARS . . . 20
4.6 HME . . . 20
4.7 Bumping . . . 20
4.8 Rotation Forest . . . 21

5 Support Vector Machines **22**
5.1 Optimal Separating Hyperplanes . . . 22
5.2 Support Vector Classifier . . . 22
5.3 Support Vector Machines . . . 22

6 Neural Networks and Deep Learning **23**
6.1 Projection pursuit regression . . . 23
6.2 Neural Networks . . . 23
6.3 Neural Gas . . . 23
6.4 Bayesian neural networks . . . 23

7 Graphical Models **25**
7.1 Undirected Graphical Models . . . 25
7.1.1 Markov Networks (Markov Random Field) . . . 25
7.1.2 Factor Graph . . . 25
7.1.3 Tree Decomposition . . . 25
7.1.4 Conditional Random Field . . . 25
7.1.5 Restricted Boltzmann Machine . . . 25
7.2 Directed Graphical Models . . . 25
7.2.1 Bayesian Network . . . 25
7.3 Mixed Graph . . . 26
7.3.1 Ancestral Graph . . . 26
7.3.2 Chain Graph . . . 26

8 Kernel Methods **27**

9 Unsupervised Learning **28**
9.1 Market Basket Analysis . . . 28
9.2 Cluster Analysis . . . 28
9.2.1 Proximity Matrices . . . 29
9.2.2 Dissimilarities Based on Attributes . . . 29
9.2.3 Object Dissimilarity . . . 29
9.2.4 Clustering Algorithms . . . 29
9.2.4.1 Combinatorial Algorithms . . . 29
9.2.4.2 Gaussian Mixtures as Soft K-means Clustering . . . 29
9.2.4.3 K-means Clustering . . . 29
9.2.4.4 Vector Quantization . . . 30
9.2.4.5 K-medoids . . . 30
9.2.4.6 Fuzzy clustering . . . 31

9.2.5 Hierarchical Clustering . . . 31
9.2.5.1 Agglomerative Clustering . . . 32
9.2.5.2 Divisive Clustering . . . 32
9.3 Self-Organizing Maps . . . 32
9.4 Principal Components, Curves and Surfaces . . . 32
9.4.1 Principal Components Analysis . . . 32
9.4.2 Principal Curves and Surfaces . . . 33
9.4.3 Spectral Clustering . . . 33
9.4.4 Kernel Principal Components Analysis . . . 34
9.4.5 Sparse Principal Components Analysis . . . 34
9.4.6 Non-negative Sparse Principal Component Analysis(NSPCA) . . . 35
9.4.7 Non-negative Matrix Factorization . . . 35
9.4.7.1 Archetypal Analysis . . . 35
9.5 Factor Analysis . . . 36
9.5.1 Exploratory Factor Analysis (EFA) . . . 37
9.5.2 Confirmatory Factor Analysis (CFA) . . . 37
9.6 Independent Component Analysis . . . 38
9.6.1 Product Density Independent Component Analysis . . . 38
9.6.2 Fast Independent Component Analysis . . . 38
9.6.3 Kernal Independent Component Analysis . . . 38
9.7 Exploratory Projection Pursuit . . . 38
9.8 Multidimensional Scaling . . . 39
9.8.1 Classical Multidimensional Scaling . . . 40
9.8.2 Metric Multidimensional Scaling . . . 40
9.8.3 Non-metric Multidimensional Scaling . . . 40
9.8.4 Generalized Multidimensional Scaling . . . 40
9.8.5 Local Multidimensional Scaling . . . 40
9.9 Nonlinear Dimension Reduction . . . 40
9.9.1 Isometric Feature Mapping . . . 41
9.9.2 Local linear Embedding . . . 41
9.9.3 Local Multidimensional Scaling . . . 41
9.10 The Google PageRank Algorithm . . . 41

10 Ensemble Methods **42**
10.1 Bayes Optimal Classifier . . . 42
10.2 Bootstrap Aggregating (Bagging) . . . 42
10.3 Random Forest . . . 42
10.4 Rotation Forest . . . 43
10.5 Boosting . . . 43
10.5.1 Boosting Trees . . . 43
10.5.2 Gradient Boosting . . . 43
10.5.3 Stochastic Gradient Boosting . . . 44
10.5.4 Collaborative Multiview Boosting . . . 44
10.6 Forward Stagewise Additive Modeling . . . 44
10.7 Learning Ensembles . . . 44
10.8 Bayesian Parameter Averaging . . . 44
10.9 Bucket of Models . . . 44
10.10Model Averaging . . . 44

10.10.1 Bayesian Model Averaging . . . 45
10.11 Stacking . . . 45

11 Other Statistical Analysis Methods and Concepts **46**
11.1 Piecewise Polynomials . . . 46
11.2 Kernels . . . 46
11.2.1 Kernel Density Estimation . . . 46
11.2.2 Kernel Density Classification . . . 46
11.2.3 Radial Basis Functions and Kernels . . . 46
11.3 Splines . . . 46
11.3.1 B-spline . . . 46
11.3.1.1 Cardinal B-spline . . . 47
11.3.1.2 Penalized B-spline . . . 47
11.3.2 Cubic Spline . . . 47
11.3.3 Natural Cubic Spline . . . 47
11.3.4 Smoothing Spline . . . 47
11.3.5 Regression Spline . . . 48
11.3.6 Multidimensional Splines . . . 48
11.4 Reproducing Kernel Hilbert Spaces . . . 48
11.5 Wavelet Smoothing . . . 48
11.5.1 Adaptive Wavelet Filtering . . . 48
11.6 Bumping . . . 48
11.7 Local Regression . . . 49
11.7.1 Local Constant Regression . . . 49
11.7.2 Local Linear Regression . . . 49
11.7.3 Local Polynomial Regression . . . 49
11.7.4 Local Regression in R^p . . . 49
11.7.5 Structured Local Regression Models in R^p . . . 49
11.7.5.1 Varying Coefficient Models . . . 49
11.7.6 Local Likelihood . . . 49
11.8 Generalized Addictive Models . . . 49
11.9 Bump Hunting . . . 50
11.10 Multivariate Adaptive Regression Splines (MARS) . . . 50
11.11 Hierarchical Mixtures of Experts(HME) . . . 50
11.12 Canonical Correlation Analysis . . . 51
11.13 Mixture Models for Density Estimation and Classification . . . 51

12 Algorithms **52**
12.1 Piecewise-Linear Path Algorithm . . . 52
12.2 Pathwise Coordinate Optimization . . . 52
12.3 Expectation Maximization algorithm (EM) . . . 52
12.4 Markov Chain Monte Carlo . . . 52
12.5 Gradient Tree Boosting Algorithm . . . 52
12.6 Backfitting Algorithm for Additive Models . . . 52
12.7 Local Scoring Algorithms . . . 52
12.8 AdaBoost . . . 53
12.9 Gradient Descent . . . 53
12.10 Steepest Descent . . . 53
12.11 Rosenblatt's perceptron learning algorithm . . . 53

12.12Forward Stagewise Boosting 53

13 Summary of the characteristics of learning methods **54**

14 Others **55**
14.1 Relative importance of predictor variables 55
14.2 Very useful links 55

Note:
The Elements of Statistical Learning (ESL), second edition Hastie *et al.* (2009)
Statistical Learning with Sparcity (SLS) Hastie *et al.* (2015)
Computer Age Statistical Inference: Algorithms, Evidence and Data Science (CASI) Efron & Hastie (2016)
Deep Learning (DL) Goodfellow *et al.* (2016)
Machine learning with R second edition Lantz (2013)
Machine learning with R cookbook Yu-Wei (2015)

Machine learning: supervised (having dependent variable y)/unsupervised learning (no dependent varible y)

Key words: Curse of dimensionarity, misspecifying,

1 Methods for Regression

Hastie *et al.* (2009) ESL page 43

1.1 Linear Regression Models and Least Squares

Hastie *et al.* (2009) ESL page 43

SIMPLE LINEAR CORRELATION AND REGRESSION
https://ww2.coastal.edu/kingw/statistics/R-tutorials/simplelinear.html

1.2 Subset Selection

Hastie *et al.* (2009) ESL page 57

1.2.1 Best-Subset Selection

Hastie *et al.* (2009) ESL page 57

R package: 'bestglm'
https://cran.r-project.org/web/packages/bestglm/vignettes/bestglm.pdf

1.2.2 Forward- and Backward-Stepwise Selection

Hastie *et al.* (2009) ESL page 58
R Package 'lars'
https://cran.r-project.org/web/packages/lars/lars.pdf page 4

Rather than search through all possible subsets (which becomes infeasible for p much larger than 40), we can seek a good path through them. Forwardstepwise selection starts with the

intercept, and then sequentially adds into the model the predictor that most improves the fit. With many candidate predictors, this might seem like a lot of computation; however, clever updating algorithms can exploit the QR decomposition for the current fit to rapidly establish the next candidate.
Backward-stepwise selection starts with the full model, and sequentially deletes the predictor that has the least impact on the fit. The candidate for dropping is the variable with the smallest Z-score.

Multiple (Linear) Regression
http://www.statmethods.net/stats/regression.html

Stepwise model selection
http://rstudio-pubs-static.s3.amazonaws.com/2899_a9129debf6bd47d2a0501de9c0dc583d.html

1.2.3 Forward-Stagewise Regression

Hastie *et al.* (2009) p60, 342
R Package 'lars'
https://cran.r-project.org/web/packages/lars/lars.pdf page 4

Forward-stagewise regression (FS) is even more constrained than forwardstepwise regression. It starts like forward-stepwise regression, with an intercept equal to $\bar{y}$, and centered predictors with coefficients initially all 0. At each step the algorithm identifies the variable most correlated with the current residual. It then computes the simple linear regression coefficient of the residual on this chosen variable, and then adds it to the current coefficient for that variable. This is continued till none of the variables have correlation with the residuals—i.e. the least-squares fit when $N > p$.

1.2.3.1 Incremental Forward Stagewise Regression

Hastie *et al.* (2009) p86

1.3 Shrinkage Methods

1.3.1 Ridge Regression

Hastie *et al.* (2009) p61
R package: 'glmnet'

How to use Ridge Regression and Lasso in R
http://ricardoscr.github.io/how-to-use-ridge-and-lasso-in-r.html

1.3.2 Lasso

Hastie *et al.* (2009) p68

R Package 'lars'
https://cran.r-project.org/web/packages/lars/lars.pdf page 4

1.3.2.1 Quantile Lasso

Package 'rqPen'
https://cran.r-project.org/web/packages/rqPen/rqPen.pdf

1.3.2.2 Lag/Ordered Lasso

R Package 'orderedLasso'
https://cran.r-project.org/web/packages/orderedLasso/orderedLasso.pdf

1.3.2.3 The Group Lasso

Hastie *et al.* (2015) p58

Package 'grplasso'
https://cran.r-project.org/web/packages/grplasso/grplasso.pdf

Package 'gglasso'
https://cran.r-project.org/web/packages/gglasso/gglasso.pdf

In other applications, features may be structurally grouped. Examples include the dummy variables that are used to code a multilevel categorical predictor, or sets of coefficients in a multiple regression problem. In such settings, it is natural to select or omit all the coefficients within a group together. The group lasso and the overlap group lasso achieve these effects by using sums of (un-squared) l_2 penalties.
Consider a linear regression model involving J groups of covariates,

$$\underset{\theta_0\in\mathbb{R},\theta_j\in\mathbb{R}^{p_j}}{\text{minimize}}\Big\{\ \frac{1}{2}\sum_{i=1}^{N}(y_i-\theta_0-\sum_{j=1}^{J}z_{ij}^T\theta_j)^2+\lambda\sum_{j=1}^{J}\|\theta_j\|_2\Big\}$$

1.3.2.4 Fused Lasso

Hastie *et al.* (2015) p77

Package 'genlasso'
https://cran.r-project.org/web/packages/genlasso/genlasso.pdf

Another kind of structural grouping arises from an underlying index set such as time; our parameters might each have an associated time stamp. We might then ask for time-neighboring coefficients to be the same or similar. The fused lasso is a method naturally tailored to such situations.

$$\underset{(\beta_0,\beta)\in\mathbb{R}\times\mathbb{R}^p}{\text{minimize}}\Big\{\ \frac{1}{2}\sum_{i=1}^{N}(y_i-\beta_0-\sum_{j=1}^{p}x_{ij}\beta_j)^2+\lambda_1\sum_{j=1}^{p}|\beta_j|+\lambda_2\sum_{j=1}^{p}|\beta_j-\beta_{j-1}|\Big\}$$

1.3.2.5 Adaptive Lasso

Hastie *et al.* (2009) p92
R Package 'parcor'
https://cran.r-project.org/web/packages/parcor/parcor.pdf

R package: 'glmnet'
Adaptive Lasso: What it is and how to implement in R
http://ricardoscr.github.io/how-to-adaptive-lasso.html

Zou, H. (2006). The adaptive lasso and its oracle properties, Journal of the American Statistical Association 101: 1418–1429.

The adaptive lasso yields consistent estimates of the parameters while retaining the attractive convexity property of the lasso.

$$\underset{(\beta_0,\beta)\in\mathbb{R}\times\mathbb{R}^p}{\text{minimize}}\left\{\frac{1}{2}\sum_{i=1}^{N}(y_i-\beta_0-x_i^T\beta)^2+\lambda\sum_{j=1}^{p}\hat{\omega}_j|\beta_j|\right\}$$

With ω we are performing a different regularization for each coefficient, i.e., this vector adjusts the penalty differently for each coefficient. The Adaptive Weights vector is defined as:$\hat{\omega}_j = \frac{1}{(\hat{\beta}_j{}^{initial})^\gamma}$. The authors suggest γ with the possible values of 0.5, 1 and 2.

1.3.2.6 Relaxed Lasso

Hastie *et al.* (2015) p12, Hastie *et al.* (2009) p91 last paragraph

R package: 'relaxo'
https://cran.r-project.org/web/packages/relaxo/relaxo.pdf

Meinshausen, N. (2007), Relaxed Lasso, Computational Statistics and Data Analysis pp. 374–393.

Two-stage process:
1. run the lasso and select the variates
2. run the ols using selected variates from lasso.

1.3.2.7 Square-root Lasso

Belloni *et al.* (2011)

1.3.3 Elastic Net

Hastie *et al.* (2015) p55

Wikipedia https://en.wikipedia.org/wiki/Elastic_net_regularization

Package 'glmnet'
ftp://debian.ustc.edu.cn/CRAN/web/packages/glmnet/glmnet.pdf

Glmnet Vignette
https://web.stanford.edu/~hastie/glmnet/glmnet_alpha.html

Such generalized penalties arise in a wide variety of settings. For instance,in microarray studies, we often find groups of correlated features, such as genes that operate in the same biological pathway. Empirically, the lasso sometimes does not perform well with highly correlated variables. By combining a squared l_2-penalty with the l_1-penalty, we obtain the elastic net, another penalized method that deals better with such correlated groups, and tends to select the correlated features (or not) together.

$$\underset{(\beta_0,\beta)\in\mathbb{R}\times\mathbb{R}^p}{\text{minimize}}\Big\{\frac{1}{2}\sum_{i=1}^{N}(y_i-\beta_0-x_i^T\beta)^2+\lambda[\frac{1}{2}(1-\alpha)\|\beta\|_2^2+\alpha\|\beta\|_1]\Big\}$$

1.3.4 Least Angle Regression

Hastie *et al.* (2009) ESL page 73

Wikipedia https://en.wikipedia.org/wiki/Least-angle_regression

R Package 'lars'
https://cran.r-project.org/web/packages/lars/lars.pdf page 4

1.3.5 Dantzig Selector

Hastie *et al.* (2009) ESL page 89
Candes & Tao (2007) proposed the following criterion:

$$min_\beta\|\beta\|_1 \quad \text{subject to} \quad \|X^T(y-X\beta)\|_\infty \le s \tag{1.1}$$

They call the solution the Dantzig selector. It can written equivalently as

$$min_\beta\|X^T(y-X\beta)\|_\infty \quad \text{subject to } |\beta\|_1 \le t \tag{1.2}$$

1.4 Methods Using Derived Input Directions

1.4.1 Principal Components Regression

Hastie *et al.* (2009) p79

Unsupervised Principal Components Regression
Check in Unsupervised Learning

Supervised Principal Components Regression (High-Dimensional Regression)
Hastie *et al.* (2009) ESL page 674

Performing Principal Components Regression (PCR) in R
https://www.r-bloggers.com/performing-principal-components-regression-pcr-in-r/

1.4.2 Partial Least Squares

Hastie *et al.* (2009) ESL page 80

R Package 'pls'
https://cran.r-project.org/web/packages/pls/pls.pdf
Partial Least Squares Regression in R
https://www.r-bloggers.com/partial-least-squares-regression-in-r/

2 Methods for Classification

Hastie *et al.* (2009) ESL page 101

2.1 Linear Discriminant Analysis

Hastie *et al.* (2009) ESL page 106

Discriminant Function Analysis
http://www.statmethods.net/advstats/discriminant.html

Computing and visualizing LDA in R
https://www.r-bloggers.com/computing-and-visualizing-lda-in-r/

2.2 Quadratic Discriminant Analysis

Hastie *et al.* (2009) ESL page 110
Quadratic discriminant function does not assume homogeneity of variance-covariance matrices.

Discriminant Analysis in R
https://rstudio-pubs-static.s3.amazonaws.com/35817_2552e05f1d4e4db8ba87b334101a43da.html

Classification: Linear Discriminant Analysis
https://rpubs.com/ryankelly/LDA-QDA

2.3 Diagonal Linear Discriminant Analysis

2.4 Reduced rank linear discriminant analysis

Hastie *et al.* (2009) p113

So far we have discussed LDA as a restricted Gaussian classifier. Part of its popularity is due to an additional restriction that allows us to view informative low-dimensional projections of the data. The K centroids in p-dimensional input space lie in an affine subspace of dimension$\leq K-1$, and if p is much larger than K, this will be a considerable drop in dimension. Moreover, in locating the closest centroid, we can ignore distances orthogonal to this subspace, since they will contribute equally to each class. Thus we might just as well project the X^* onto this centroid-spanning subspace H_{K-1}, and make distance comparisons there. Thus there is a fundamental dimension reduction in LDA, namely, that we need only consider the data in a subspace of dimension at most $K-1$.

Package 'SPCALDA'
A new reduced-rank LDA method which works for high dimensional multi-class data.
https://cran.r-project.org/web/packages/SPCALDA/SPCALDA.pdf

Linear discriminant analysis (LDA) and the related Fisher's linear discriminant are methods used in statistics, pattern recognition and machine learning to find a linear combination of features which characterizes or separates two or more classes of objects or events. The resulting combination may be used as a linear classifier, or, more commonly, for dimensionality reduction before later classification.

Discriminant Analysis in R
https://rstudio-pubs-static.s3.amazonaws.com/35817_2552e05f1d4e4db8ba87b334101a43da.html

2.5 Generalized Linear Discriminant Analysis

Hastie *et al.* (2009) p438

In the remainder of this chapter we describe a class of techniques that attend to all these issues by generalizing the LDA model. This is achieved largely by three different ideas.

The first idea is to recast the LDA problem as a linear regression problem. Many techniques exist for generalizing linear regression to more flexible, nonparametric forms of regression. This in turn leads to more flexible forms of discriminant analysis, which we call **Flexible Discriminant Analysis (FDA)**. In most cases of interest, the regression procedures can be seen to identify an enlarged set of predictors via basis expansions. FDA amounts to LDA in this enlarged space, the same paradigm used in SVMs.

In the case of too many predictors, such as the pixels of a digitized image, we do not want to expand the set: it is already too large. The second idea is to fit an LDA model, but penalize its coefficients to be smooth or otherwise coherent in the spatial domain, that is, as an image. We call this procedure **penalized discriminant analysis** or **PDA**. With FDA itself, the expanded basis set is often so large that regularization is also required (again as in SVMs). Both of these can be achieved via a suitably regularized regression in the context of the FDA model. The third idea is to model each class by a mixture of two or more Gaussians with different centroids, but with every component Gaussian, both within and between classes, sharing the same covariance matrix. This allows for more complex decision boundaries, and allows for subspace reduction as in LDA. We call this extension **mixture discriminant analysis** or **MDA**. All three of these generalizations use a common framework by exploiting their connection with LDA.

2.5.1 Flexible Discriminant Analysis

Hastie *et al.* (2009) p440

Package 'mda'
Mixture and flexible discriminant analysis, multivariate adaptive regression splines (MARS), BRUTO, ...
https://cran.r-project.org/web/packages/mda/mda.pdf page 7

2.5.2 Regularized Discriminant Analysis

Hastie *et al.* (2009) p112, 656, 446
Penalized Discriminant Analysis Hastie *et al.* (2009) p446

Friedman (1989) proposed a compromise between LDA and QDA, which allows one to shrink the separate covariances of QDA toward a common covariance as in LDA. These methods are very similar in flavor to ridge regression.

R package 'rda'
Shrunken Centroids Regularized Discriminant Analysis for the classification purpose in high dimensional data.
https://cran.r-project.org/web/packages/rda/rda.pdf

Package 'rrlda'
This package offers methods to perform robust regularized linear discriminant analysis.
https://cran.r-project.org/web/packages/rrlda/rrlda.pdf

Package 'sparsediscrim'
A collection of sparse and regularized discriminant analysis methods intended for small-sample, high-dimensional data sets. The package features the High-Dimensional Regularized Discriminant Analysis classifier.
https://cran.r-project.org/web/packages/sparsediscrim/sparsediscrim.pdf

Package 'mda'
https://cran.r-project.org/web/packages/mda/mda.pdf page 9

R Package 'penalizedLDA'
Perform penalized linear discriminant analysis using L1 or fused lasso penalties.
https://cran.r-project.org/web/packages/penalizedLDA/penalizedLDA.pdf

2.5.3 Mixture Discriminant Analysis

Hastie *et al.* (2009) ESL page 449

Linear discriminant analysis can be viewed as a prototype classifier. Each class is represented by its centroid, and we classify to the closest using an appropriate metric. In many situations a single prototype is not sufficient to represent inhomogeneous classes, and mixture models are more appropriate.

Package 'mda'
https://cran.r-project.org/web/packages/mda/mda.pdf page 14

2.6 Logistic Regression

Hastie *et al.* (2009) ESL page 119
LOGISTIC REGRESSION
https://ww2.coastal.edu/kingw/statistics/R-tutorials/logistic.html

2.6.1 Regularized Logistic Regression

Hastie *et al.* (2009) ESL page 125, 661, 657

L_1 and L_2 Regularized Logistic Regression
Package 'LiblineaR'
https://cran.r-project.org/web/packages/LiblineaR/LiblineaR.pdf p4

2.6.2 Nonparametric Logistic Regression

Hastie *et al.* (2009) p161

2.6.3 Additive Logistic Regression

Hastie *et al.* (2009) p299

2.7 Naive Bayes Classifier

Hastie *et al.* (2009) p210, p108
It is especially appropriate when the dimension p of the feature space is high, making density estimation unattractive.

3 Prototype Methods and Nearest-Neighbors

model-free methods for classification and pattern recognition, highly unstructured, not useful for understanding the nature of the relationship between the features and class outcome.
used in regression: low-dimensional. not work well in high-dimensional regression.

3.1 K-means Clustering

Check in Unsupervised Learning

3.2 Learning Vector Quantization (LVQ)

Hastie *et al.* (2009) ESL page 462

Package 'class'
https://cran.r-project.org/web/packages/class/class.pdf

LVQ can be a source of great help in **classifying text documents**.

The Learning Vector Quantization algorithm belongs to the field of Artificial Neural Networks and Neural Computation. More broadly to the field of Computational Intelligence. The Learning Vector Quantization algorithm is a supervised neural network that uses a competitive (winner-take-all) learning strategy. It is related to other supervised neural networks such as the Perceptron and the Back-propagation algorithm. It is related to other competitive learning neural networks such as the the Self-Organizing Map algorithm that is a similar algorithm for unsupervised learning with the addition of connections between the neurons. Additionally, LVQ is a baseline technique that was defined with a few variants LVQ1, LVQ2, LVQ2.1, LVQ3, OLVQ1, and OLVQ3 as well as many third-party extensions and refinements too numerous to list.

Strategy
The information processing objective of the algorithm is to prepare a set of codebook (or prototype) vectors in the domain of the observed input data samples and to use these vectors to classify unseen examples. An initially random pool of vectors is prepared which are then exposed to training samples. A winner-take-all strategy is employed where one or more of the most similar vectors to a given input pattern are selected and adjusted to be closer to the input vector, and in some cases, further away from the winner for runners up. The repetition of this process results in the distribution of codebook vectors in the input space which approximate the underlying distribution of samples from the test dataset.

Heuristics

Learning Vector Quantization was designed for classification problems that have existing data sets that can be used to supervise the learning by the system. The algorithm does not support regression problems.

- LVQ is non-parametric, meaning that it does not rely on assumptions about that structure of the function that it is approximating.
- Real-values in input vectors should be normalized such that $x \in [0, 1]$.
- Euclidean distance is commonly used to measure the distance between real-valued vectors, although other distance measures may be used (such as dot product), and data specific distance measures may be required for non-scalar attributes.
- There should be sufficient training iterations to expose all the training data to the model multiple times.
- The learning rate is typically linearly decayed over the training period from an initial value to close to zero.
- The more complex the class distribution, the more codebook vectors that will be required, some problems may need thousands.
- Multiple passes of the LVQ training algorithm are suggested for more robust usage, where the first pass has a large learning rate to prepare the codebook vectors and the second pass has a low learning rate and runs for a long time (perhaps 10-times more iterations).

http://www.cleveralgorithms.com/nature-inspired/neural/lvq.html

3.3 Gaussian Mixture Models (GMM)

Hastie *et al.* (2009) ESL page 463

R package 'mixtools': normalmixEM

It's important to understand that no "labels" have been assigned here actually. Unlike k-means which assigns each data point to a cluster (defined as a "hard-label"), mixture models provide what are called "soft-labels". The end-user decides on what "threshold" to use to assign data into the components.

Using Mixture Models for Clustering
http://tinyheero.github.io/2015/10/13/mixture-model.html

mixture-examples.R
https://www.stat.cmu.edu/~cshalizi/uADA/12/lectures/mixture-examples.R

Fitting a Mixture Model Using the Expectation-Maximization Algorithm in R
http://tinyheero.github.io/2016/01/03/gmm-em.html

3.4 K-Nearest-Neighbor Classifiers

Hastie *et al.* (2009) ESL page 14, 463
One of the most comprehensible non-parametric methods is k-nearest-neighbors: find the points which are most similar to you, and do what, on average, they do. There are two big drawbacks to it: first, you're defining "similar" entirely in terms of the inputs, not the response; second, k is constant everywhere, when some points just might have more very-similar neighbors than

others. Trees get around both problems: leaves correspond to regions of the input space (a neighborhood), but one where the responses are similar, as well as the inputs being nearby; and their size can vary arbitrarily. Prediction trees are adaptive nearest-neighbor methods.

R package 'class', function: knn

Best way to learn kNN Algorithm using R Programming
https://www.analyticsvidhya.com/blog/2015/08/learning-concept-knn-algorithms-programming/

3.5 Adaptive Nearest-Neighbor Methods

Hastie *et al.* (2009) ESL page 475

No R package

When nearest-neighbor classification is carried out in a high-dimensional feature space, the nearest neighbors of a point can be very far away, causing bias and degrading the performance of the rule.

Implicit in nearest-neighbor classification is the assumption that the class probabilities are roughly constant in the neighborhood, and hence simple averaging gives a good estimate for the class posterior.

In high dimensional space, the neighborhood represented by the few nearest samples may not be local.

Adaptive Nearest-Neighbor Methods
https://onlinecourses.science.psu.edu/stat857/node/185

Discriminant Adaptive Nearest Neighbor Classification
https://web.stanford.edu/~hastie/Papers/dann_IEEE.pdf

3.6 Global Dimension Reduction for Nearest-Neighbors

Hastie *et al.* (2009) ESL page 479

No R package

Discriminant Adaptive Nearest Neighbor Classification
https://web.stanford.edu/~hastie/Papers/dann_IEEE.pdf

4 Decision Tree Learning

Wikipedia https://en.wikipedia.org/wiki/Decision_tree_learning

4.1 Regression Trees

Hastie *et al.* (2009) ESL page 307

4.2 Classification Trees

Hastie *et al.* (2009) p308

CART(Classification and Regression Trees) modeling, conditional inference trees, and random forests
http://www.statmethods.net/advstats/cart.html

A Brief Tour of the Trees and Forests
https://www.r-bloggers.com/a-brief-tour-of-the-trees-and-forests/

Classification & Regression Trees
http://www.di.fc.ul.pt/~jpn/r/tree/tree.html

4.3 Boosting Trees

Check in Ensemble Methods

4.4 Bagging

Check in Ensemble Methods

4.5 MARS

Other Statistical Analysis Methods and Concepts

4.6 HME

Other Statistical Analysis Methods and Concepts

4.7 Bumping

Other Statistical Analysis Methods and Concepts

4.8 Rotation Forest

Check in Ensemble Methods

5 Support Vector Machines

5.1 Optimal Separating Hyperplanes

Hastie *et al.* (2009) ESL page 132
Support vector machines: The linearly separable case.

5.2 Support Vector Classifier

Hastie *et al.* (2009) ESL page 417

Optimal separating hyperplane: classes are separated by a linear boundary.

Support vector classifier: Classes may not be separable by a linear boundary.

5.3 Support Vector Machines

Hastie *et al.* (2009) ESL page 423
R package 'e1071'
https://cran.r-project.org/web/packages/e1071/e1071.pdf

Support vector machine produces nonlinear boundaries by constructing a linear boundary in a large, transformed version of the feature space.

Support Vector Regression with R
http://www.svm-tutorial.com/2014/10/support-vector-regression-r/

The SVM as a penalization method, ESL page 426
Package 'penalizedSVM'
https://cran.r-project.org/web/packages/penalizedSVM/penalizedSVM.pdf

6 Neural Networks and Deep Learning

Deep learning has been characterized as a buzzword, or a rebranding of neural networks.

6.1 Projection pursuit regression

Hastie *et al.* (2009) ESL page 389
Model overview:

$$Y = \beta_0 + \sum_{j=1}^{r} f_j(\beta_j' x) + \epsilon$$

Thus this model takes the form of the basic additive model but with the additional β_j component; making it fit $\beta_j' x$ rather than the actual inputs x. The vector $\beta_j' X$ is the projection of X onto the unit vector β_j, where the directions β_j are chosen to optimize model fit.

6.2 Neural Networks

R package: 'neuralnet'

Neural Networks with R: A Simple Example http://gekkoquant.com/2012/05/26/neural-networks-with-

Fitting a neural network in R: 'neuralnet' package https://www.r-bloggers.com/fitting-a-neural-network

Standardization: At the outset it is best to standardize all inputs to have mean zero and standard deviation one.

Number of Hidden Units and Layers: Usually, if at all necessary, one hidden layer is enough for a vast numbers of applications. As far as the number of neurons is concerned, it should be between the input layer size and the output layer size, usually 2/3 of the input size.

Multiple minima: The final solution obtained is quite dependent on the choice of starting points. One must at least try a number of random starting configurations, and choose the solution giving lowest (penalized) error. Probably a better approach is to use the average predictions over the collection of networks as the final prediction (Ripley, 1996). This is preferable to averaging the weights, since the nonlinearity of the model implies that this averaged solution could be quite poor. Another approach is via **bagging**, which averages the predictions of networks training from randomly perturbed versions of the training data. This is described in Section 8.7.

Overfitting: A validation dataset is useful for determining when to stop, since we expect the validation error to start increasing.A more explicit method for regularization is **weight decay**, which is analogous to ridge regression used for linear models (Section 3.4.1).

6.3 Neural Gas

6.4 Bayesian neural networks

Hastie *et al.* (2009) ESL page 409

Bayesian network in R: Introduction https://www.r-bloggers.com/bayesian-network-in-r-introduction/

7 Graphical Models

Wikipidia https://en.wikipedia.org/wiki/Graphical_model#Types_of_graphical_models

7.1 Undirected Graphical Models

Hastie *et al.* (2009) ESL page 625

7.1.1 Markov Networks (Markov Random Field)

Wikipedia https://en.wikipedia.org/wiki/Markov_random_field

XMRF: an R package to fit Markov Networks to high-throughput genetics data
https://bmcsystbiol.biomedcentral.com/articles/10.1186/s12918-016-0313-0

7.1.2 Factor Graph

Wikipedia https://en.wikipedia.org/wiki/Factor_graph

7.1.3 Tree Decomposition

In machine learning, tree decompositions are also called **junction trees, clique trees, or join trees**.

Wikipedia https://en.wikipedia.org/wiki/Tree_decomposition

7.1.4 Conditional Random Field

Wikipedia https://en.wikipedia.org/wiki/Conditional_random_field

7.1.5 Restricted Boltzmann Machine

Wikipedia https://en.wikipedia.org/wiki/Restricted_Boltzmann_machine

7.2 Directed Graphical Models

7.2.1 Bayesian Network

Wikipedia https://en.wikipedia.org/wiki/Bayesian_network

Bayesian network in R: Introduction
https://www.r-bloggers.com/bayesian-network-in-r-introduction/

Learning Bayesian Networks with the bnlearn R Package
https://arxiv.org/pdf/0908.3817.pdf

7.3 Mixed Graph

Wikipedia https://en.wikipedia.org/wiki/Mixed_graph

7.3.1 Ancestral Graph

Wikipedia https://en.wikipedia.org/wiki/Ancestral_graph

7.3.2 Chain Graph

A chain graph is a graph which may have both directed and undirected edges, but without any directed cycles (i.e. if we start at any vertex and move along the graph respecting the directions of any arrows, we cannot return to the vertex we started from if we have passed an arrow). Both directed acyclic graphs and undirected graphs are special cases of chain graphs, which can therefore provide a way of unifying and generalizing Bayesian and Markov networks.

8 Kernel Methods

Wikipedia https://en.wikipedia.org/wiki/Kernel_method

R package 'kernlab'
Kernel-based machine learning methods for classification, regression, clustering, novelty detection, quantile regression and dimensionality reduction. Among other methods 'kernlab' includes Support Vector Machines, Spectral Clustering, Kernel PCA, Gaussian Processes and a QP solver. https://cran.r-project.org/web/packages/kernlab/kernlab.pdf

9 Unsupervised Learning

9.1 Market Basket Analysis

Hastie *et al.* (2009) ESL page 488

Package 'arules'
Provides the infrastructure for representing, manipulating and analyzing transaction data and patterns (frequent itemsets and association rules). Also provides interfaces to C implementations of the association mining algorithms Apriori and Eclat by C. Borgelt.
https://cran.r-project.org/web/packages/arules/arules.pdf
Package 'arulesViz'
Extends package arules with various visualization techniques for association rules and itemsets. The package also includes several interactive visualizations for rule exploration.
https://cran.r-project.org/web/packages/arulesViz/arulesViz.pdf

Market Basket Analysis with R
http://www.salemmarafi.com/code/market-basket-analysis-with-r/

9.2 Cluster Analysis

Hastie *et al.* (2009) ESL page 501
Algorithms: Clustering, combinatorial, k-means,vector quantization, k-medoids

Package 'cluster'
https://cran.r-project.org/web/packages/cluster/cluster.pdf

Cluster analysis, also called data segmentation, has a variety of goals. All relate to grouping or segmenting a collection of objects into subsets or "clusters," such that those within each cluster are more closely related to one another than objects assigned to different clusters. An object can be described by a set of measurements, or by its relation to other objects. In addition, the goal is sometimes to arrange the clusters into a natural hierarchy. This involves successively grouping the clusters themselves so that at each level of the hierarchy, clusters within the same group are more similar to each other than those in different groups.Cluster analysis is also used to form descriptive statistics to ascertain whether or not the data consists of a set distinct subgroups, each group representing objects with substantially different properties. This latter goal requires an assessment of the degree of difference between the objects assigned to the respective clusters.

Cluster Analysis
http://www.statmethods.net/advstats/cluster.html

Cluster Analysis with R
https://rstudio-pubs-static.s3.amazonaws.com/33876_1d7794d9a86647ca90c4f182df93f0e8.html

K-means clustering requires us to specify the number of clusters, and finding the optimal number of clusters can often be hard.
K Means Clustering in R

https://www.r-bloggers.com/k-means-clustering-in-r/

Cluster Analysis and R - Berkeley Statistics
https://www.stat.berkeley.edu/~s133/Cluster2a.html

9.2.1 Proximity Matrices

9.2.2 Dissimilarities Based on Attributes

9.2.3 Object Dissimilarity

9.2.4 Clustering Algorithms

The goal of cluster analysis is to partition the observations into groups ("clusters") so that the pairwise dissimilarities between those assigned to the same cluster tend to be smaller than those in different clusters. Clustering algorithms fall into three distinct types: combinatorial algorithms, mixture modeling, and mode seeking.

Combinatorial algorithms work directly on the observed data with no direct reference to an underlying probability model. **Mixture modeling** supposes that the data is an i.i.d sample from some population described by a probability density function. This density function is characterized by a parameterized model taken to be a mixture of component density functions; each component density describes one of the clusters. This model is then fit to the data by maximum likelihood or corresponding Bayesian approaches. **Mode seekers ("bump hunters")** take a nonparametric perspective, attempting to directly estimate distinct modes of the probability density function. Observations "closest" to each respective mode then define the individual clusters. Mixture

9.2.4.1 Combinatorial Algorithms

9.2.4.2 Gaussian Mixtures as Soft K-means Clustering

9.2.4.3 K-means Clustering

Hastie *et al.* (2009) ESL page 460

Wikipedia https://en.wikipedia.org/wiki/K-means_clustering

k-means clustering is a method of vector quantization, originally from signal processing, that is popular for cluster analysis in data mining. k-means clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster. This results in a partitioning of the data space into Voronoi cells.

The problem is computationally difficult (NP-hard); however, there are efficient heuristic algorithms that are commonly employed and converge quickly to a local optimum. These are usually similar to the expectation-maximization algorithm for mixtures of Gaussian distributions via an iterative refinement approach employed by both algorithms. Additionally, they both use cluster centers to model the data; however, k-means clustering tends to find clusters of comparable spatial extent, while the expectation-maximization mechanism allows clusters to have different shapes.

The algorithm has a loose relationship to the k-nearest neighbor classifier, a popular machine learning technique for classification that is often confused with k-means because of the k in the name. One can apply the 1-nearest neighbor classifier on the cluster centers obtained by k-means to classify new data into the existing clusters. This is known as nearest centroid classifier or Rocchio algorithm.

K-means clustering is a method for finding clusters and cluster centers in a set of unlabeled data. One chooses the desired number of cluster centers, say R, and the K-means procedure iteratively moves the centers to minimize the total within cluster variance.

R package 'stats', function: kmeans

The general idea of a clustering algorithm is to partition a given dataset into distinct, exclusive clusters so that the data points in each group are quite similar to each other. K-means clustering is a method for finding clusters and cluster centers in a set of unlabeled data. One chooses the desired number of cluster centers, say R, and the K-means procedure iteratively moves the centers to minimize the total within cluster variance.

The most common partitioning method is the K-means cluster analysis. Conceptually, the K-means algorithm:

1. Selects K centroids (K rows chosen at random)
2. Assigns each data point to its closest centroid
3. Recalculates the centroids as the average of all data points in a cluster (i.e., the centroids are p-length mean vectors, where p is the number of variables)
4. Assigns data points to their closest centroids
5. Continues steps 3 and 4 until the observations are not reassigned or the maximum number of iterations (R uses 10 as a default) is reached.

Implementation details for this approach can vary.

All variables must be continuous and the approach can be severely affected by outliers. They also perform poorly in the presence of non-convex (e.g., U-shaped) clusters.

K-means Clustering (from "R in Action")
https://www.r-statistics.com/2013/08/k-means-clustering-from-r-in-action/

Example of K-Means Clustering with R
https://rpubs.com/FelipeRego/K-Means-Clustering

9.2.4.4 Vector Quantization

Hastie *et al.* (2009) p514

Wikipedia https://en.wikipedia.org/wiki/Vector_quantization

9.2.4.5 K-medoids

Hastie *et al.* (2009) p515
Wikipedia https://en.wikipedia.org/wiki/K-medoids

The k-medoids algorithm is a clustering algorithm related to the k-means algorithm and the medoidshift algorithm. Both the k-means and k-medoids algorithms are partitional (breaking

the dataset up into groups) and both attempt to minimize the distance between points labeled to be in a cluster and a point designated as the center of that cluster. In contrast to the k-means algorithm, k-medoids chooses datapoints as centers (medoids or exemplars) and works with a generalization of the Manhattan Norm to define distance between datapoints instead of l_2. This method was proposed in 1987 for the work with l_1 norm and other distances.

k-medoid is a classical partitioning technique of clustering that clusters the data set of n objects into k clusters known a priori. A useful tool for determining k is the silhouette.

It is more robust to noise and outliers as compared to k-means because it minimizes a sum of pairwise dissimilarities instead of a sum of squared Euclidean distances.

A medoid can be defined as the object of a cluster whose average dissimilarity to all the objects in the cluster is minimal. i.e. it is a most centrally located point in the cluster.

9.2.4.6 Fuzzy clustering

Wikipedia https://en.wikipedia.org/wiki/Fuzzy_clustering#Fuzzy_c-means_clustering

Fuzzy clustering (also referred to as soft clustering) is a form of clustering in which each data point can belong to more than one cluster.

Clustering or cluster analysis involves assigning data points to clusters (also called buckets, bins, or classes), or homogeneous classes, such that items in the same class or cluster are as similar as possible, while items belonging to different classes are as dissimilar as possible. Clusters are identified via similarity measures. These similarity measures include distance, connectivity, and intensity. Different similarity measures may be chosen based on the data or the application.

9.2.5 Hierarchical Clustering

Hastie *et al.* (2009) p520

Wikipedia https://en.wikipedia.org/wiki/Hierarchical_clustering

R package 'stats': hclust

In data mining and statistics, hierarchical clustering (also called hierarchical cluster analysis or HCA) is a method of cluster analysis which seeks to build a hierarchy of clusters. Strategies for hierarchical clustering generally fall into two types: Rokach & Maimon (2005)

Agglomerative: This is a "bottom up" approach: each observation starts in its own cluster, and pairs of clusters are merged as one moves up the hierarchy.

Divisive: This is a "top down" approach: all observations start in one cluster, and splits are performed recursively as one moves down the hierarchy. In general, the merges and splits are determined in a greedy manner. The results of hierarchical clustering are usually presented in a dendrogram.

Hierarchical Clustering in R
https://www.r-bloggers.com/hierarchical-clustering-in-r-2/

Hierarchical Cluster Analysis
http://www.r-tutor.com/gpu-computing/clustering/hierarchical-cluster-analysis

9.2.5.1 Agglomerative Clustering

Hastie *et al.* (2009) p523

9.2.5.2 Divisive Clustering

Hastie *et al.* (2009) p526

Kaufman & Rousseeuw (2008) The basic principle of divisive clustering was published as the DIANA (DIvisive ANAlysis Clustering) algorithm. Initially, all data is in the same cluster, and the largest cluster is split until every object is separate. Because there exist $O(2^n)$ ways of splitting each cluster, heuristics are needed. DIANA chooses the object with the maximum average dissimilarity and then moves all objects to this cluster that are more similar to the new cluster than to the remainder. An obvious alternate choice is k-means clustering with $k = 2$,[13] but any other clustering algorithm can be used that always produces at least two clusters.

9.3 Self-Organizing Maps

Hastie *et al.* (2009) ESL page 528

Wikipedia https://en.wikipedia.org/wiki/Self-organizing_map

R package 'kohonen'

Self-Organizing Maps (SOMs) are an unsupervised data visualization technique that can be used to visualize high-dimensional data sets in lower (typically 2) dimensional representations.

Self-Organizing Maps for Customer Segmentation using R
https://www.r-bloggers.com/self-organising-maps-for-customer-segmentation-using-r/

9.4 Principal Components, Curves and Surfaces

Hastie *et al.* (2009) ESL page 534

9.4.1 Principal Components Analysis

R package: 'prcomp'

Principal component analysis (PCA) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components. The number of principal components is less than or equal to the smaller of (number of original variables or number of observations). This transformation is defined in such a way that the first principal component has the largest possible variance (that is, accounts for as much of the variability in the data as possible), and each succeeding component in turn has the highest variance possible under the constraint that it is orthogonal to the preceding components. The resulting vectors are an uncorrelated orthogonal basis set. PCA is sensitive to the relative scaling of the original variables.

Since skewness and the magnitude of the variables influence the resulting PCs, it is good practice to apply skewness transformation, center and scale the variables prior to the application of PCA. In the example above, we applied a log transformation to the variables but we could have been more general and applied a Box and Cox transformation.

Principal component analysis in R
http://www.sthda.com/english/wiki/principal-component-analysis-in-r-prcomp-vs-princomp-r-software-a

Principal Components and Factor Analysis
http://www.statmethods.net/advstats/factor.html

Computing and visualizing PCA in R
https://tgmstat.wordpress.com/2013/11/28/computing-and-visualizing-pca-in-r/

Principal Component Analysis
http://www.di.fc.ul.pt/~jpn/r/pca/pca.html#kernel-pca

9.4.2 Principal Curves and Surfaces

Hastie *et al.* (2009) p541

R package: 'analogue', prcurve
https://cran.r-project.org/web/packages/princurve/princurve.pdf

R package: 'vegan', prc

A principal curve is a non-parametric generalization of the principal component and is a curve that passes through the middle of a cloud of data points for a certain definition of 'middle'.

Principal curves example (Elements of Statistical Learning)
https://www.r-bloggers.com/principal-curves-example-elements-of-statistical-learning/

9.4.3 Spectral Clustering

Hastie *et al.* (2009) ESL page 544
R package 'kernlab': specc

Wikipedia https://en.wikipedia.org/wiki/Spectral_clustering

Spectral clustering is nice because it gives you as much flexibility as you want to define how pairs of data points are similar or dissimilar. K-means only works well for data that are grouped in elliptically shaped, whereas spectral clustering can theoretically work well for any group. For example, the data in this image is easily clustered by spectral, but would not be by k-means. The flexibility of spectral clustering can also be a burden in that there are an infinite ways to group points.

The basic idea (and all the flexibility) behind spectral clustering is that you define the similarity between any two data points however you want, and put them in a matrix. So if you have 100 data points, you will end up with a 100x100 matrix, where the rth row and cth

column is the similarity between the rth data point and the cth data point. You can define "similarity" any way you want. Popular methods are Euclidean distance, a kernel function of the Euclidean distance , or a k nearest neighbors approach.

Once you have the similarity matrix, you need to create a normalized/unnormalized Laplacian matrix, then calculate the eigenvectors and eigenvalues of the the Laplacian. Finally, use the k-means algorithm on the eigenvalues corresponding to the k smallest eigenvectors. This will give you k clusters (something else you need to specify).

Spectral clustering is a class of techniques that perform cluster division using eigenvectors of the similarity matrix. The division is such that points in the same cluster should be highly similar and points in different clusters should have highly dissimilar. Thus, spectral clustering is a non-parametric method of clustering. One of the advantages of this clustering mechanism is that it is not affected by outliers or noise and performs fast!

In application to image segmentation, spectral clustering is known as segmentation-based object categorization.

Spectral Clustering
http://www.di.fc.ul.pt/~jpn/r/spectralclustering/spectralclustering.html

Hands-on Spectral clustering in R
https://www.linkedin.com/pulse/hands-on-spectral-clustering-r-madhur-modi

Unsupervised Image Segmentation with Spectral Clustering with R
https://www.r-bloggers.com/unsupervised-image-segmentation-with-spectral-clustering-with-r/

9.4.4 Kernel Principal Components Analysis

Hastie *et al.* (2009) p547

Wikipedia
https://en.wikipedia.org/wiki/Kernel_principal_component_analysis

Principal Component Analysis (last part)
http://www.di.fc.ul.pt/~jpn/r/pca/pca.html#kernel-pca

LECTURE :KERNEL PCA
http://www.cs.haifa.ac.il/~rita/uml_course/lectures/KPCA.pdf

9.4.5 Sparse Principal Components Analysis

Hastie *et al.* (2009) ESL page 550
R package" 'elasticnet' or 'nsprcomp'

Wikipedia https://en.wikipedia.org/wiki/Sparse_PCA

Principal component analysis(PCA) is one of the classical methods in multivariate statistics. In addition, it is now widely used as a way to implement data-processing and dimension-

reduction. Besides statistics, there are numerous applications about PCA in engineering, biology, and so on. There are two main optimal properties of PCA, which are guaranteeing minimal information loss and uncorrelated principal components. That's why PCA becomes so successful nowadays.

Although the classical(traditional) PCA method is theoretically sound, how to explain the result(the derived PCs) is still a headache. For example, you know, the loadings are typically non-zero(each PC is a linear combination of all variables), but the question is that, if we choose the first PC, we can consider that the first PC depends on all of the original variables. Obviously it is not good for us. Another questions is that we usually need to require the weights which should be non-negative based on our pre-study intuition.

Because of these drawbacks above, there are several extensions of the traditional PCA methods proposed. **Sparse principal component analysis(SPCA)** and **non-negative sparse principal component analysis(NSPCA)** probably are good solutions. However, the obvious question is the computational complexity. As a matter of fact, no matter the sparseness constraint or sparseness constraint will lead the optimization problem become a NP hard problem. So far, researchers have not found a general algorithm which can solve these problem perfectly, although there are numerous methodologies(via relaxing several constraints or ended up with a local optimal solution) which have been developed. So before we choose the eventual method, we should weight the advantages and disadvantages.

By the way, all of the PCA, SPCA and NSPCA can be implented via R. Function princomp(for Q-mode PCA use function prcomp) performs a principal components analysis in R. The function nsprcomp in package nsprcomp performs a constrained principal component analysis(both SPCA and NSPCA). If the value of the argument nneg is TRUE, the loadings should be non-negative. It is worth mentioning that set a appropriate argument is very important.

PCA or SPCA or NSPCA?
https://www.r-bloggers.com/pca-or-spca-or-nspca/

Sparse PCA example in R: part 1
https://crude2refined.wordpress.com/2013/07/30/sparse-pca-example-in-r-part-1/

9.4.6 Non-negative Sparse Principal Component Analysis(NSPCA)

9.4.7 Non-negative Matrix Factorization

Hastie *et al.* (2009) p553

9.4.7.1 Archetypal Analysis

Hastie *et al.* (2009) ESL page 554

Package 'archetypes'
https://cran.r-project.org/web/packages/archetypes/archetypes.pdf

Archetypal analysis in the statistics is an unsupervised learning method similar to the cluster analysis and introduced by Cutler & Breiman (1994). Rather than "typical" observations (cluster centers), it seeks extremal points in the multidimensional data, the "archetypes". The

archetypes are convex combinations of observations chosen so that observations can be approximated by convex combinations of the archetypes.

Cluster analysis focuses on groupings within the cloud of individual respondents. Archetypal analysis, on the other hand, searches the periphery for concentrations of more extreme individuals. Cluster analysis describes its segments using the "average" member as the prototype. Archetypal analysis uses extreme exemplars to describe the frontiers.

This method, due to Cutler and Breiman (1994), approximates data points by prototypes that are themselves linear combinations of data points. In this sense it has a similar flavor to K-means clustering. However, rather than approximating each data point by a single nearby prototype, archetypal analysis approximates each data point by a convex combination of a collection of prototypes. The use of a convex combination forces the prototypes to lie on the convex hull of the data cloud. In this sense, the prototypes are "pure,", or "archetypal."

9.5 Factor Analysis

Hastie *et al.* (2009) ESL page 558

Wikipedia https://en.wikipedia.org/wiki/Factor_analysis

Factor analysis is a statistical method used to describe variability among observed, correlated variables in terms of a potentially lower number of unobserved variables called factors. For example, it is possible that variations in six observed variables mainly reflect the variations in two unobserved (underlying) variables. Factor analysis searches for such joint variations in response to unobserved latent variables. The observed variables are modelled as linear combinations of the potential factors, plus "error" terms. Factor analysis aims to find independent latent variables. Followers of factor analytic methods believe that the information gained about the interdependencies between observed variables can be used later to reduce the set of variables in a dataset. Factor analysis is not used to any significant degree in physics, biology and chemistry but is used very heavily in psychometrics personality theories, marketing, product management, operations research. Users of factor analysis believe that it helps to deal with data sets where there are large numbers of observed variables that are thought to reflect a smaller number of underlying/latent variables.

Factor analysis is a classical technique developed in the statistical literature that aims to identify these latent sources. Factor analysis models are typically wed to Gaussian distributions, which has to some extent hindered their usefulness. More recently, **independent component analysis** has emerged as a strong competitor to factor analysis, and as we will see, relies on the non-Gaussian nature of the underlying sources for its success.

Suppose we have a set of p observable random variables, $x_1, \ldots, x_p$ with means $\mu_1, \ldots, \mu_p$.

Suppose for some unknown constants l_{ij} and k unobserved random variables F_j(called "common factors" because they influence all the observed random variables), where $i \in 1, \ldots, p$ and $j \in 1, \ldots, k$, where $k < p$, we have

$x_i - \mu_i = l_{i1}F_1 + \cdots + l_{ik}F_k + \varepsilon_i$.

Here, the ε_i are unobserved stochastic error terms with zero mean and finite variance, which may not be the same for all i.

In matrix terms, we have

$x - \mu = LF + \varepsilon$.

If we have n observations, then we will have the dimensions $x_{p\times n}$, $L_{p\times k}$, and $F_{k\times n}$. Each column of x and F denote values for one particular observation, and matrix L does not vary across observations.

Also we will impose the following assumptions on F:

F and ε are independent. $\mathrm{E}(F) = 0$. $\mathrm{Cov}(F) = I$ (to make sure that the factors are uncorrelated).

Any solution of the above set of equations following the constraints for F is defined as the "factors", and L as the "loading matrix".

Suppose $\mathrm{Cov}(x - \mu) = \Sigma$. Then note that from the conditions just imposed on F, we have

$\mathrm{Cov}(x - \mu) = \mathrm{Cov}(LF + \varepsilon)$,

or

$\Sigma = L\mathrm{Cov}(F)L^T + \mathrm{Cov}(\varepsilon)$,

or

$\Sigma = LL^T + \Psi$.

Note that for any orthogonal matrix Q, if we set $L = LQ$ and $F = Q^T F$, the criteria for being factors and factor loadings still hold. Hence a set of factors and factor loadings is unique only up to orthogonal transformation.

Factor Analysis
https://web.stanford.edu/class/psych253/tutorials/FactorAnalysis.html

Factor Analysis
http://www.di.fc.ul.pt/~jpn/r/factoranalysis/factoranalysis.html

9.5.1 Exploratory Factor Analysis (EFA)

Wikipedia https://en.wikipedia.org/wiki/Exploratory_factor_analysis

In multivariate statistics, exploratory factor analysis (EFA) is a statistical method used to uncover the underlying structure of a relatively large set of variables. EFA is a technique within factor analysis whose overarching goal is to identify the underlying relationships between measured variables. It is commonly used by researchers when developing a scale (a scale is a collection of questions used to measure a particular research topic) and serves to identify a set of latent constructs underlying a battery of measured variables. It should be used when the researcher has no a priori hypothesis about factors or patterns of measured variables.

Principal Components and Factor Analysis
http://www.statmethods.net/advstats/factor.html

EXPLORATORY FACTOR ANALYSIS IN R
http://connor-johnson.com/2014/03/30/exploratory-factor-analysis-in-r/

9.5.2 Confirmatory Factor Analysis (CFA)

Wikipedia https://en.wikipedia.org/wiki/Confirmatory_factor_analysis

In statistics, confirmatory factor analysis (CFA) is a special form of factor analysis, most commonly used in social research. It is used to test whether measures of a construct are consistent with a researcher's understanding of the nature of that construct (or factor). As such, the objective of confirmatory factor analysis is to test whether the data fit a hypothesized measurement model. This hypothesized model is based on theory and/or previous analytic research.

All together now – Confirmatory Factor Analysis in R
https://www.r-bloggers.com/all-together-now-confirmatory-factor-analysis-in-r/

confirmatory factor analysis
http://lavaan.ugent.be/tutorial/cfa.html

Confirmatory Factor Analysis Using the SEM Package in R
https://www.methodsconsultants.com/tutorial/confirmatory-factor-analysis-using-the-sem-package-in-r/

9.6 Independent Component Analysis

Hastie *et al.* (2009) ESL page 560

A direct approach to Independent Component Analysis:

9.6.1 Product Density Independent Component Analysis

Hastie *et al.* (2009) ESL page 567
R package: 'ProDenICA'
https://cran.r-project.org/web/packages/ProDenICA/ProDenICA.pdf

ProDenICA: Product Density Independent Component Analysis
https://rdrr.io/cran/ProDenICA/man/ProDenICA.html

9.6.2 Fast Independent Component Analysis

Hastie *et al.* (2009) ESL page 569
R package: 'FastICA'
https://cran.r-project.org/web/packages/fastICA/fastICA.pdf

9.6.3 Kernal Independent Component Analysis

Hastie *et al.* (2009) ESL page 569
R package: 'KernelICA'

9.7 Exploratory Projection Pursuit

Hastie *et al.* (2009) ESL page 565
R package 'xgobi'

https://cran.r-project.org/web/packages/xgobi/xgobi.pdf

Wikipedia https://en.wikipedia.org/wiki/Projection_pursuit

Friedman & Tukey (1974) proposed exploratory projection pursuit, a graphical exploration technique for visualizing high-dimensional data. Their view was that most low (one- or two-dimensional) projections of highdimensional data look Gaussian. Interesting structure, such as clusters or long tails, would be revealed by non-Gaussian projections. They proposed a number of projection indices for optimization, each focusing on a different departure from Gaussianity. Since their initial proposal, a variety of improvements have been suggested (Huber, 1985; Friedman (1987)), and a variety of indices, including entropy, are implemented in the interactive graphics package Xgobi (Swayne et al., 1991, now called GGobi).

Friedman (1987) A new projection pursuit algorithm for exploring multivariate data is presented that has both statistical and computational advantages over previous methods. A number of practical issues concerning its application are addressed. A connection to multivariate density estimation is established, and its properties are investigated through simulation studies and application to real data. The goal of exploratory projection pursuit is to use the data to find low- (one-, two-, or three-) dimensional projections that provide the most revealing views of the full-dimensional data. With these views the human gift for pattern recognition can be applied to help discover effects that may not have been anticipated in advance. Since linear effects are directly captured by the covariance structure of the variable pairs (which are straightforward to estimate) the emphasis here is on the discovery of nonlinear effects such as clustering or other general nonlinear associations among the variables. Although arbitrary nonlinear effects are impossible to parameterize in full generality, they are easily recognized when presented in a low-dimensional visual representation of the data density. Projection pursuit assigns a numerical index to every projection that is a functional of the projected data density. The intent of this index is to capture the degree of nonlinear structuring present in the projected distribution. The pursuit consists of maximizing this index with respect to the parameters defining the projection. Since it is unlikely that there is only one interesting view of a multivariate data set, this procedure is iterated to find further revealing projections. After each maximizing projection has been found, a transformation is applied to the data that removes the structure present in the solution projection while preserving the multivariate structure that is not captured by it. The projection pursuit algorithm is then applied to these transformed data to find additional views that may yield further insight. This projection pursuit algorithm has potential advantages over other dimensionality reduction methods that are commonly used for data exploration. It focuses directly on the "interestingness" of a projection rather than indirectly through the interpoint distances. This allows it to be unaffected by the scale and (linear) correlational structure of the data, helping it to overcome the "curse of dimensionality" that tends to plague methods based on multidimensional scaling, parametric mapping, cluster analysis, and principal components.

9.8 Multidimensional Scaling

Hastie *et al.* (2009) ESL page 570

Wikipedia https://en.wikipedia.org/wiki/Multidimensional_scaling

Multidimensional Scaling with R (from "Mastering Data Analysis with R")
https://www.r-statistics.com/2016/01/multidimensional-scaling-with-r-from-mastering-data-analysis-wit

Multidimensional Scaling
http://www.statmethods.net/advstats/mds.html

Multidimensional Scaling (MDS) with R
https://www.r-bloggers.com/multidimensional-scaling-mds-with-r/

9.8.1 Classical Multidimensional Scaling

It is also known as Principal Coordinates Analysis (PCoA), Torgerson Scaling or Torgerson–Gower scaling.

9.8.2 Metric Multidimensional Scaling

7 Functions to do Metric Multidimensional Scaling in R
https://www.r-bloggers.com/7-functions-to-do-metric-multidimensional-scaling-in-r/

9.8.3 Non-metric Multidimensional Scaling

NMDS TUTORIAL IN R
https://jonlefcheck.net/2012/10/24/nmds-tutorial-in-r/

Non-metric Multidimensional Scaling
http://ecology.msu.montana.edu/labdsv/R/labs/lab9/lab9.html

9.8.4 Generalized Multidimensional Scaling

9.8.5 Local Multidimensional Scaling

Hastie *et al.* (2009) ESL page 572

Local Multidimensional Scaling performs multidimensional scaling in local regions, and then uses convex optimization to fit all the pieces together.

9.9 Nonlinear Dimension Reduction

Hastie *et al.* (2009) ESL page 572

note: no R package yet
Wikipedia https://en.wikipedia.org/wiki/Nonlinear_dimensionality_reduction

Several methods have been recently proposed for nonlinear dimension reduction, similar in spirit to principal surfaces. The idea is that the data lie close to an intrinsically low-dimensional

nonlinear manifold embedded in a high-dimensional space. These methods can be thought of as "flattening" the manifold, and hence reducing the data to a set of low-dimensional coordinates that represent their relative positions in the manifold. They are useful for problems where signal-to-noise ratio is very high (e.g., physical systems), and are probably not as useful for observational data with lower signal-to-noise ratios.

9.9.1 Isometric Feature Mapping

9.9.2 Local linear Embedding

9.9.3 Local Multidimensional Scaling

9.10 The Google PageRank Algorithm

Hastie *et al.* (2009) ESL page 576

R package 'igraph': page.rank

Wikipedia https://en.wikipedia.org/wiki/PageRank

PageRank is an algorithm used by Google Search to rank websites in their search engine results. PageRank was named after Larry Page, one of the founders of Google. PageRank is a way of measuring the importance of website pages. According to Google:

PageRank works by counting the number and quality of links to a page to determine a rough estimate of how important the website is. The underlying assumption is that more important websites are likely to receive more links from other websites.

Finding the essential R packages using the pagerank algorithm
http://blog.revolutionanalytics.com/2014/12/a-reproducible-r-example-finding-the-most-popular-package
html

10 Ensemble Methods

In statistics and machine learning, ensemble methods use multiple learning algorithms to obtain better predictive performance than could be obtained from any of the constituent learning algorithms alone. Unlike a statistical ensemble in statistical mechanics, which is usually infinite, a machine learning ensemble refers only to a concrete finite set of alternative models, but typically allows for much more flexible structure to exist among those alternatives.

10.1 Bayes Optimal Classifier

10.2 Bootstrap Aggregating (Bagging)

R Package 'adabag'

Hastie *et al.* (2009) ESL page 282

Breiman, L. (1996a). Bagging predictors, Machine Learning 26: 123–140.
Proposed by Leo Breiman in 1994 to improve the classification by combining classifications of randomly generated training sets.
Given a standard training set D of size n, bagging generates m new training sets D_i, each of size n', by sampling from D uniformly and with replacement. The m models are fitted using the above m bootstrap samples and combined by averaging the output (for regression) or voting (for classification).

$$\hat{f}_{bag}(x) = \frac{1}{B}\sum_{b=1}^{B} \hat{f}^{*b}(x)$$

For more, check Wikipedia https://en.wikipedia.org/wiki/Bootstrap_aggregating

example using adabag packagehttps://artax.karlin.mff.cuni.cz/r-help/library/adabag/html/bagging.html

without using adabag: Bagging/Bootstrap Aggregation with R :Practical walkthroughs on machine learning, data exploration and finding insight.http://amunategui.github.io/bagging-in-R/

10.3 Random Forest

Hastie *et al.* (2009) ESL page 587
Lantz (2013) p369
Yu-Wei (2015) p274

Wikipedia https://en.wikipedia.org/wiki/Random_forest

The algorithm starts by building out trees similar to the way a normal decision tree algorithm works. However, every time a split has to made, it uses only a small random subset of features to make the split instead of the full set of features (usually $\sqrt{p}$), where p is the number of predictors). It builds multiple trees using the same process, and then takes the average of all the trees to arrive at the final model. This works by reducing the amount of correlation

between trees, and thus helping reduce the variance of the final tree.

Predicting wine quality using Random Forests
https://www.r-bloggers.com/predicting-wine-quality-using-random-forests/

10.4 Rotation Forest

in which every decision tree is trained by first applying principal component analysis (PCA) on a random subset of the input features. Rodríguez *et al.* (2006)

Package 'rotationForest'
https://cran.r-project.org/web/packages/rotationForest/rotationForest.pdf

10.5 Boosting

Hastie *et al.* (2009) p337

R Package 'adabag'
R Package 'gbm': Generalized Boosted Regression Models

Wikipedia https://en.wikipedia.org/wiki/Boosting_(machine_learning)

A short example for Adaboost
https://qizeresearch.wordpress.com/2013/12/05/short-example-for-adaboost/

An Attempt to Understand Boosting Algorithms (very good)
https://www.r-bloggers.com/an-attempt-to-understand-boosting-algorithms/

10.5.1 Boosting Trees

Hastie *et al.* (2009) p353

10.5.2 Gradient Boosting

Hastie *et al.* (2009) p358
R Package 'gbm': Generalized Boosted Regression Models

Wikipedia https://en.wikipedia.org/wiki/Gradient_boosting

Gradient Boosting: Analysis of LendingClub's Data
https://www.r-bloggers.com/gradient-boosting-analysis-of-lendingclubs-data/

Gradient Boosting Machines
https://github.com/ledell/useR-machine-learning-tutorial/blob/master/gradient-boosting-machines.

Rmd

Shrinkage: Empirically it has been found that using small learning rates yields dramatic improvements in model's generalization ability over gradient boosting without shrinking. However, it comes at the price of increasing computational time both during training and querying: lower learning rate requires more iterations.

10.5.3 Stochastic Gradient Boosting

Hastie *et al.* (2009) p365

Gradient boosting constructs additive regression models by sequentially fitting a simple parameterized function (base learner) to current "pseudo'-residuals by least squares at each iteration. The pseudo-residuals are the gradient of the loss functional being minimized, with respect to the model values at each training data point evaluated at the current step. It is shown that both the approximation accuracy and execution speed of gradient boosting can be substantially improved by incorporating randomization into the procedure. Specifically, at each iteration a subsample of the training data is drawn at random (without replacement) from the full training data set. This randomly selected subsample is then used in place of the full sample to fit the base learner and compute the model update for the current iteration. This randomized approach also increases robustness against overcapacity of the base learner.

10.5.4 Collaborative Multiview Boosting

10.6 Forward Stagewise Additive Modeling

Hastie *et al.* (2009) ESL page 342

10.7 Learning Ensembles

Hastie *et al.* (2009) ESL page 616

10.8 Bayesian Parameter Averaging

Bayesian methods for nonparametric regression can also be viewed as ensemble methods.

10.9 Bucket of Models

10.10 Model Averaging

Hastie *et al.* (2009) ESL page 288

Model Averaging:
Madigan, D. and Raftery, A. (1994). Model selection and accounting for model uncertainty using Occam's window, Journal of the American Statistical Association 89: 1535–46.

10.10.1 Bayesian Model Averaging

Bayesian Model Averaging: A Tutorial
http://www.stat.colostate.edu/~jah/papers/statsci.pdf

Package 'BMA'
https://cran.r-project.org/web/packages/BMA/BMA.pdf

10.11 Stacking

Hastie *et al.* (2009) ESL page 288

Wolpert, D. (1992). Stacked generalization, Neural Networks 5: 241–259.
Breiman, L. (1996b). Stacked regressions, Machine Learning 24: 51–64.
Leblanc, M. and Tibshirani, R. (1996). Combining estimates in regression and classification, Journal of the American Statistical Association 91: 1641–1650.

Suppose ζ is some quantity of interest, for example, a prediction $f(x)$ at some fixed feature value x. The posterior distribution of ζ is

$$Pr(\zeta|Z) = \sum_{m=1}^{M} Pr(\zeta|M_m, Z)Pr(M_m|Z)$$

with posterior mean

$$E(\zeta|Z) = \sum_{m=1}^{M} E(\zeta|M_m, Z)Pr(M_m|Z)$$

$$\hat{w}^{st} = \underset{w}{argmin} \sum_{i=1}^{N} [y_i - \sum_{m=1}^{M} w_m \hat{f}_m^{-i}(x_i)]$$

Committee methods (unweighted average), BIC criterion (weigted average, Hastie *et al.* (2009) ESL 7.7 page 233)
Stacking avoids giving unfairly high weight to models with higher complexity.

11 Other Statistical Analysis Methods and Concepts

11.1 Piecewise Polynomials

Hastie *et al.* (2009) p141

11.2 Kernels

11.2.1 Kernel Density Estimation

Hastie *et al.* (2009) p208

11.2.2 Kernel Density Classification

Hastie *et al.* (2009) p210

11.2.3 Radial Basis Functions and Kernels

Hastie *et al.* (2009) p212

Wikipedia https://en.wikipedia.org/wiki/Radial_basis_function

11.3 Splines

Hastie *et al.* (2009) p141-167, 191-213

11.3.1 B-spline

Hastie *et al.* (2009) p186
Wikipedia https://en.wikipedia.org/wiki/B-spline

For any given set of knots, the B-spline is unique, hence the name, B being short for Basis. The usefulness of B-splines lies in the fact that any spline function of order "n" on a given set of knots can be expressed as a linear combination of B-splines:

$$S_{n,t}(x) = \sum_i \alpha_i B_{i,n}(x). \tag{11.1}$$

This follows from the fact that all pieces have the same continuity properties, within their individual range of support, at the knots.
Expressions for the polynomial pieces can be derived by means of the Cox-de Boor recursion formula

$$B_{i,0}(x) := \begin{cases} 1 & \text{if} \quad t_i \leq x < t_{i+1} \\ 0 & \text{otherwise} \end{cases} \tag{11.2}$$

$$B_{i,k}(x) := \frac{x - t_i}{t_{i+k} - t_i} B_{i,k-1}(x) + \frac{t_{i+k+1} - x}{t_{i+k+1} - t_{i+1}} B_{i+1,k-1}(x). \tag{11.3}$$

11.3.1.1 Cardinal B-spline

A cardinal B-spline has a constant separation, "h", between knots. The cardinal B-splines for a given degree "n" are just shifted copies of each other. They can be obtained from the simpler definition.

$$B_{i,n,t}(x) = \frac{x - t_i}{h} n[0, \ldots, n](. - t_i)_+^{n-1} \tag{11.4}$$

The "placeholder" notation is used to indicate that the nth divided difference of the function $(t-x)_+^{n-1}$ of the two variables "t" and "x" is to be taken by fixing "x" and considering $(t-x)_+^{n-1}$ as a function of "t" alone.

A cardinal B-spline has uniform spaced knots, therefore interpolation between the knots equals convolution with a smoothing kernel.

11.3.1.2 Penalized B-spline

The term P-spline stands for "penalized B-spline". It refers to using the B-spline representation where the coefficients are determined partly by the data to be fitted, and partly by an additional penalty function that aims to impose smoothness to avoid overfitting.

11.3.2 Cubic Spline

Hastie *et al.* (2009) p143
A simple example of a cubic spline is $S(t) = |t|^3$ as

$$S(t) = \begin{cases} t^3 & t \geq 0 \\ -t^3 & t < 0 \end{cases} \tag{11.5}$$

and $S'(0) = 0$ $S''(0) = 0$.

An example of using a cubic spline to create a bell shaped curve is the Irwin-Hall distribution polynomials:

$$f_X(x) = \begin{cases} \frac{1}{4}(x+2)^3 & -2 \leq x \leq -1 \\ \frac{1}{4}\left(3|x|^3 - 6x^2 + 4\right) & -1 \leq x \leq 1 \\ \frac{1}{4}(2-x)^3 & 1 \leq x \leq 2 \end{cases} \tag{11.6}$$

11.3.3 Natural Cubic Spline

Hastie *et al.* (2009) p145

A natural cubic spline adds additional constraints, namely that the function is linear beyond the boundary knots. This frees up four degrees of freedom (two constraints each in both boundary regions), which can be spent more profitably by sprinkling more knots in the interior region.

11.3.4 Smoothing Spline

Hastie *et al.* (2009) p151

Let $(x_i, Y_i); x_1 < x_2 < \cdots < x_n, i \in \mathbb{Z}$ be a sequence of observations, modeled by the relation $Y_i = f(x_i)$. The smoothing spline estimate $\hat{f}$ of the function is defined to be the minimizer (over the class of twice differentiable functions) of

$$\sum_{i=1}^{n}(Y_i - \hat{f}(x_i))^2 + \lambda \int_{x_1}^{x_n} \hat{f}''(x)^2 \, dx. \tag{11.7}$$

11.3.5 Regression Spline

Hastie *et al.* (2009) p144

11.3.6 Multidimensional Splines

Hastie *et al.* (2009) p162

11.4 Reproducing Kernel Hilbert Spaces

Hastie *et al.* (2009) p167

Wikipedia https://en.wikipedia.org/wiki/Reproducing_kernel_Hilbert_space

11.5 Wavelet Smoothing

Hastie *et al.* (2009) p174

Wikipedia https://en.wikipedia.org/wiki/Wavelet

R Package 'wavelets'
https://cran.r-project.org/web/packages/wavelets/wavelets.pdf

R package 'wavethresh'
https://cran.r-project.org/web/packages/wavethresh/wavethresh.pdf

11.5.1 Adaptive Wavelet Filtering

Hastie *et al.* (2009) p179

11.6 Bumping

Hastie *et al.* (2009) ESL 290

Tibshirani, R. and Knight, K. (1999). Model search and inference by bootstrap "bumping, Journal of Computational and Graphical Statistics 8: 671–686.

A technique for finding a better single model
Boostrap → models → best fit.

11.7 Local Regression

11.7.1 Local Constant Regression

11.7.2 Local Linear Regression

Hastie *et al.* (2009) p194

11.7.3 Local Polynomial Regression

Hastie *et al.* (2009) p197

11.7.4 Local Regression in R^p

Hastie *et al.* (2009) p200

11.7.5 Structured Local Regression Models in R^p

Hastie *et al.* (2009) p201

11.7.5.1 Varying Coefficient Models

Hastie *et al.* (2009) p203

11.7.6 Local Likelihood

Hastie *et al.* (2009) p205

11.8 Generalized Addictive Models

Hastie *et al.* (2009) ELS page 295

Package 'gam'

$$E(Y|X_1, X_2, ..., X_P) = \alpha + f_1(X1) + f_2(X_2) + ... + f_p(X_p)$$

where f_j's are unspecified smooth ("nonparametric") functions.

Fit each function using a scatterplot smoother (e.g., a cubic smoothing spline or kernel smoother).

11.9 Bump Hunting

Hastie *et al.* (2009) ESL page 317

PRIM: Patient Rule Induction Method

Bump Hunting in high-dimentional data, by Jerome Friedman and Nicholas Fisher http://statweb.stanford.edu/~jhf/ftp/prim.pdf

11.10 Multivariate Adaptive Regression Splines (MARS)

Hastie *et al.* (2009) p321, 162

Package 'mda'
Mixture and flexible discriminant analysis, multivariate adaptive regression splines (MARS), BRUTO, ...
https://cran.r-project.org/web/packages/mda/mda.pdf page 12

Package 'earth'
Build regression models using the techniques in Friedman's papers "Fast MARS" and "Multivariate Adaptive Regression Splines". (The term "MARS" is trademarked and thus not used in the name of the package.)
https://cran.r-project.org/web/packages/earth/earth.pdf

Wikipedia https://en.wikipedia.org/wiki/Multivariate_adaptive_regression_splines

MARS builds models of the form

$$\hat{f}(x) = \sum_{i=1}^{k} c_i B_i(x) \tag{11.8}$$

The model is a weighted sum of basis functions $B_i(x)$. Each c_i is a constant coefficient. For example, each line in the formula for ozone above is one basis function multiplied by its coefficient.
Each basis function $B_i(x)$ takes one of the following three forms:
1) a constant 1. There is just one such term, the intercept.
2) a "hinge" function. A hinge function has the form $\max(0, x - const)$ or $\max(0, const - x)$. MARS automatically selects variables and values of those variables for knots of the hinge functions.
3) a product of two or more hinge functions. These basis functions can model interaction between two or more variables.

11.11 Hierarchical Mixtures of Experts(HME)

Hastie *et al.* (2009) ESL page 329

Hierarchical mixtures of experts and EM algorithm https://www.cs.toronto.edu/~hinton/absps/hme.pdf

Package https://cran.r-project.org/web/packages/mixtools/mixtools.pdf

11.12 Canonical Correlation Analysis

Wipipedia https://en.wikipedia.org/wiki/Canonical_correlation

11.13 Mixture Models for Density Estimation and Classification

Hastie *et al.* (2009) p214

12 Algorithms

12.1 Piecewise-Linear Path Algorithm

Hastie *et al.* (2009) ESL page 89

12.2 Pathwise Coordinate Optimization

Hastie *et al.* (2009) ESL page 92

12.3 Expectation Maximization algorithm (EM)

Hastie *et al.* (2009) p272

EM algorithm 1
http://rstudio-pubs-static.s3.amazonaws.com/1001_3177e85f5e4840be840c84452780db52.html

EM algorithm 2
http://rstudio-pubs-static.s3.amazonaws.com/154174_78c021bc71ab42f8add0b2966938a3b8.html

12.4 Markov Chain Monte Carlo

Hastie *et al.* (2009) p279

12.5 Gradient Tree Boosting Algorithm

Hastie *et al.* (2009) p361

Gradient Boosting Algorithm
https://www.analyticsvidhya.com/blog/2015/09/complete-guide-boosting-methods/

An Attempt to Understand Boosting Algorithm(s)
https://www.r-bloggers.com/an-attempt-to-understand-boosting-algorithms/

XGBoost, AdaBoost, Gentle Boost

12.6 Backfitting Algorithm for Additive Models

Hastie *et al.* (2009) p298

12.7 Local Scoring Algorithms

Hastie *et al.* (2009) p300

12.8 AdaBoost

Hastie *et al.* (2009) p339

12.9 Gradient Descent

Gradient descent in r
https://www.r-bloggers.com/gradient-descent-in-r/

12.10 Steepest Descent

Hastie *et al.* (2009) p358

12.11 Rosenblatt's perceptron learning algorithm

Hastie *et al.* (2009) ESL page 130

12.12 Forward Stagewise Boosting

Hastie *et al.* (2009) p342

13 Summary of the characteristics of learning methods

Hastie *et al.* (2009) ESL page 351

14 Others

14.1 Relative importance of predictor variables

Hastie *et al.* (2009) ESL page 367

14.2 Very useful links

R package: caret http://topepo.github.io/caret/index.html

Tuning Machine Learning Models Using the Caret R Package http://machinelearningmastery.com/tuning-machine-learning-models-using-the-caret-r-package/

Appendix

References

Belloni, Alexandre, Chernozhukov, Victor, & Wang, Lie. 2011. Square-root lasso: pivotal recovery of sparse signals via conic programming. *Biometrika*, **98**(4), 791–806.

Candes, Emmanuel, & Tao, Terence. 2007. The Dantzig Selector: Statistical Estimation When p Is Much Larger than n. *The Annals of Statistics*, **35**(6), 2313–2351.

Cutler, Adele, & Breiman, Leo. 1994. Archetypal Analysis. *Technometrics*, **36**(4), 338–347.

Efron, Bradley, & Hastie, Trevor. 2016. *Computer Age Statistical Inference: Algorithms, Evidence, and Data Science*. 1st edn. New York, NY, USA: Cambridge University Press.

Friedman, Jerome H. 1987. Exploratory projection pursuit. *Journal of the American statistical association*, **82**(397), 249–266.

Friedman, Jerome H. 1989. Regularized Discriminant Analysis. *Journal of the American Statistical Association*, **84**(405), 165–175.

Friedman, Jerome H, & Tukey, John W. 1974. A projection pursuit algorithm for exploratory data analysis. *IEEE Transactions on computers*, **100**(9), 881–890.

Goodfellow, Ian, Bengio, Yoshua, & Courville, Aaron. 2016. *Deep Learning*. MIT Press. http://www.deeplearningbook.org.

Hastie, Trevor, Tibshirani, Robert, & Friedman, Jerome. 2009. *The Elements of Statistical Learning*. Springer Series in Statistics. New York, NY, USA: Springer New York Inc.

Hastie, Trevor, Tibshirani, Robert, & Wainwright, Martin. 2015. *Statistical Learning with Sparsity: The Lasso and Generalizations*. Chapman & Hall/CRC.

Kaufman, Leonard, & Rousseeuw, Peter J. 2008. *Finding Groups in Data*. John Wiley & Sons, Inc.

Lantz, Brett. 2013. *Machine learning with R*. Packt Publishing Ltd.

Rodríguez, Juan J., Kuncheva, Ludmila I., & Alonso, Carlos J. 2006. Rotation forest: A new classifier ensemble method. *IEEE TRANS. PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, **28**(10), 1619–1630.

Rokach, Lior, & Maimon, Oded. 2005. *Clustering Methods*. Boston, MA: Springer US. Pages 321–352.

Yu-Wei, Chiu David Chiu. 2015. *Machine learning with R cookbook*. Packt Publishing Ltd.