Machine Learning Methods

Prepared by: *Shayan Ali Akbar*
Contact: *sakbar (at) purdue.edu*

Classification Methods

	Discriminative / Generative	Parametric / Non-parametric	Model Space	Search Function (convex/nc) (smooth/ns)	Score Function	Overfitting	Functional Form / Prediction process	Learning Procedure
Naïve Bayes Classifier	**Generative**: Probabilities (cpds and prior) are estimated and decision boundary not explicit	**Parametric**: Number of probabilities to estimate are fixed	Probabilities (cpds and prior)	Max the likelihood (c: max of likelihood) (s: likelihood is differentiable)	Likelihood	Smoothing with Laplace correction or Dirichlet	Test X = [13, 1] (1) P(class=1 | X1=13, X2=1) = P(class=1|X1=13) x P(class=1|X2=1) (2) P(class=0 | X1=13, X2=1) = P(class=0|X1=13) x P(class=0|X2=1) (3) If P(.) in (1) > P(.) in (2) then predict that class = 1, otherwise 0.	Find conditional and prior probability distributions. If Y is the class label and X1, X2, X3 features the find: P(Y=y|X1=x1), P(Y=y|X2=x2), P(Y=y|X3=x3) P(Y=y) These can be found by counting the number of occurrences in the training examples. For example, P(Y=1|X=2) can be found by counting the number of Y=1 examples in which X=2 and dividing it by total number of times X=2 appears.
Decision Trees	**Discriminative**: Thresholds define boundary	**Non-parametric**: Number of nodes and thresholds depend on data	All possible trees	Combinatorial greedy search as function is discrete	Feature split functions like Information gain, Gini gain, Chi-squared loss	Pre-prune using stat test, Post-prune using held out data	Test X = [13, 1] Traverse the tree using the values of X and comparing them against the node values. Once hitting the leaf node do a majority voting to decide the label.	Recursive algorithm. Start with root node and keep building the tree for a certain depth or until the leaf node contains single class label. To select which node to pick, find the information or gini gain or chi square distance. The node with highest gain or least distance should be picked. Do a split of all the examples at the input of this selected node. Split can be done based on a threshold if feature/node values are continuous. Threshold should be so that both the example sets are maximally separated.
Bagged Trees	**Discriminative:** Thresholds define boundary	**Non-parametric:** Data dependent nodes and thresholds	All possible sets of trees	Combinatorial greedy search	Feature split functions like Information gain, Gini gain, Chi-squared loss	Pre-prune using stat test, Post-prune using held out data	Apply model of each tree to test set. Use majority predication or average prediction	For a given number of trees, obtain a bootstrap sample by drawing N instances with replacement from the entire dataset and learn a tree model from sampled data
Random Forests	**Discriminative:** Thresholds define boundary	**Non-parametric:** Data dependent nodes and thresholds	All possible sets of trees	Combinatorial greedy search	Feature split functions like Information gain, Gini gain, Chi-squared loss	Pre-prune using stat test, Post-prune using held out data	Apply model of each tree to test set. Use majority predication or average prediction	Same as Bagged Trees. Only that not all features are used to learn each tree. Reduces correlation between the models. For each tree split, a random sample of k features is drawn first, and only those features are considered when selecting the best feature to split on
Boosted Trees	**Discriminative:** Thresholds define boundary	**Non-parametric:** Data dependent nodes and thresholds	All possible sets of trees	Combinatorial greedy search	Feature split functions like Information gain, Gini gain, Chi-squared loss	Pre-prune using stat test, Post-prune using held out data	Apply model of each tree to test set. Use weighted prediction. Weights decided at learning phase.	For given number of trees learn model from the data. Then calculate the error of model and up-weight the examples that are incorrectly classified to form data for next step. Then normalize weights to sum to 1. Set weight for this tree. This tree weight is used when making prediction.
Perceptron	**Discriminative**: Weights define boundary	**Parametric**: Number of weights are fixed	All possible weight values	Refinement of weights	0/1 loss or MSE	Regularize (penalize large weights)	Functional form: $sign(\sum w_i x_i + b)$ w b are parameters, x is test input The sign declares the label	If $y(j)(\sum w_i x_i(j) + b) \leq 0$ Update weight as $w + \alpha y(j) x(j)$ Iterate over examples j
Artificial Neural Network	**Discriminative**: Weights define boundary	**Parametric**: Number of weights are fixed (topology fixed)	All possible weight values	Backpropagation to find weights	0/1 loss or MSE	Regularize (penalize large weights)	Forward propagation through the network, the output (last) layer produces the probabilities of each label	Give each input at input (first) layer, forward propagate through network, get the probabilities as output, get the errors at output layer, backpropagate errors and adjust weights
Nearest Neighbor	**Discriminative**: Voronoi edges define boundary	**Non-parametric**: Number of Voronoi cells change with data	All possible tessellations of feature space. Choose k, distance function, voting procedure	Implicit search	0/1 loss or MSE	k-NN with majority voting instead of 1-NN	Find k nearest neighbors of the given input and then do majority voting to label this input	Form the voronoi tessellation given the data points
Logistic Regression	**Generative**: Probability distribution that maximizes likelihood is estimated and not explicit boundary	**Parametric**: Number of weights are fixed	All possible weight values	Max the likelihood (c: max of likelihood) (s: likelihood is differentiable)	Likelihood	Regularize (penalize large weights)	Functional form: $\mathrm{P}(\mathrm{y}=1\|\mathrm{x}) = \frac{1}{e^{-(\mathrm{w}^{\mathrm{T}}x++b))}}$ Is the probability of label 1	Start with zero weights w, make predictions $\hat{y}_i$ with the current w using function form, find gradient for each parameter in w as $\Delta_j = \sum (y_i - \hat{y}_i) x_{ij} - \lambda w_j$ Update $w_j = w_j + \alpha \Delta_j$
Support Vector Machine	**Discriminative**: Weights define boundary	**Parametric**: Number of weights are fixed	All possible weight values	Min weights and hinge loss to maximize the margin	Margin and hinge loss	Use slack variables to relax constraint	$sign(\sum w_i x_i + b)$	Same as LR, but gradient calculated as: $\Delta_j = \frac{1}{N}\sum(\lambda w_j - \Delta_{ji})$, $\Delta_{ji} = y_i x_i$ if $y_i \hat{y}_i < 1$ Update $w_j = w_j - \alpha \Delta_j$

Clustering Methods

	Hard / Soft	Partition/ Hierarchical/ Probability model	Model Space	Search Function	Score Function	Knowledge representation	Clustering Procedure
K-means	**Hard**: A point is only in one cluster	**Partition**	All possible partitions of the examples into k groups	Iterative refinement correspond to greedy hill-climbing	Minimize within-cluster sum of squared error	K clusters are defined by canonical members (e.g., centroids)	Start with k randomly chosen centroids, assign instances to closest centroid, and recompute cluster centroids. Repeat until no changes in assignments
Mixture Models	**Soft**: A point can be in multiple clusters with certain probabilities	**Probability model**	parameters = mixture coefficient and component parameters	Expectation maximization. iteratively find parameters that maximize likelihood and predicts cluster memberships	Likelihood	parameters = mixture coefficient and component parameters	For each data point: Select component i randomly based on component weights Generate data point by sampling randomly from component I
Hierarchical	**In between:** Have clustering for multiple distances	**Hierarchical**	All possible dendrograms (i.e., hierarchies of partitions from 1 to n)	Local greedy search	Local across-cluster distance (e.g., single link)	Dendrogram represents a hierarchy of clusterings	Agglomerative: merge clusters successively Divisive: divided clusters successively

Disclaimers:

(1) This is merely a cheat sheet and is not meant to give a detailed description of the individual methods. I leave that matter for more scholarly folks than me.

(2) There may be outright errors on this sheet. Please email me if you happen to find some. I will be happy to make the update.