Linear classifier

In the field of machine learning, the goal of statistical classification is to use an object's characteristics to identify which class (or group) it belongs to. A **linear classifier** achieves this by making a classification decision based on the value of a linear combination of the characteristics. An object's characteristics are also known as feature values and are typically presented to the machine in a vector called a feature vector. Such classifiers work well for practical problems such as document classification, and more generally for problems with many variables (features), reaching accuracy levels comparable to non-linear classifiers while taking less time to train and use.[1]

1 Definition

If the input feature vector to the classifier is a real vector $\vec{x}$, then the output score is

$$y = f(\vec{w} \cdot \vec{x}) = f\left(\sum_j w_j x_j\right),$$

where $\vec{w}$ is a real vector of weights and f is a function that converts the dot product of the two vectors into the desired output. (In other words, $\vec{w}$ is a one-form or linear functional mapping $\vec{x}$ onto **R**.) The weight vector $\vec{w}$ is learned from a set of labeled training samples. Often f is a simple function that maps all values above a certain threshold to the first class and all other values to the second class. A more complex f might give the probability that an item belongs to a certain class.

For a two-class classification problem, one can visualize the operation of a linear classifier as splitting a high-dimensional input space with a hyperplane: all points on one side of the hyperplane are classified as “yes”, while the others are classified as “no”.

A linear classifier is often used in situations where the speed of classification is an issue, since it is often the fastest classifier, especially when $\vec{x}$ is sparse. Also, linear classifiers often work very well when the number of dimensions in $\vec{x}$ is large, as in document classification, where each element in $\vec{x}$ is typically the number of occurrences of a word in a document (see document-term matrix). In such cases, the classifier should be well-regularized.

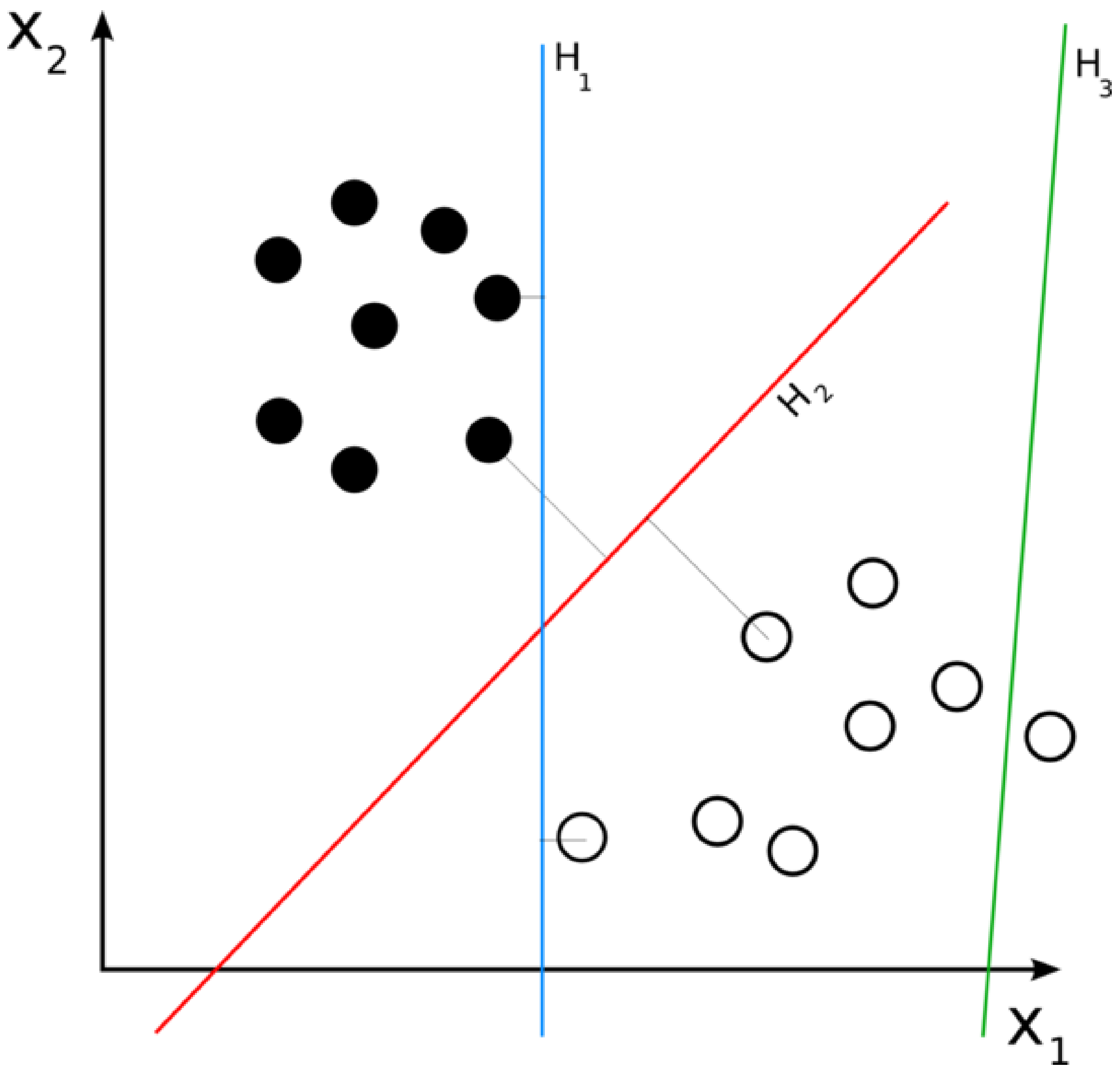
2 Generative models vs. discriminative models

There are two broad classes of methods for determining the parameters of a linear classifier $\vec{w}$. They can be Generative and Discriminative models.[2][3] Methods of the first class model conditional density functions $P(\vec{x}|\text{class})$. Examples of such algorithms include:

- Linear Discriminant Analysis (or Fisher's linear discriminant) (LDA)—assumes Gaussian conditional density models
- Naive Bayes classifier with multinomial or multivariate Bernoulli event models.

The second set of methods includes discriminative models, which attempt to maximize the quality of the output on a training set. Additional terms in the training cost function can easily perform regularization of the final model. Examples of discriminative training of linear classifiers include

- Logistic regression—maximum likelihood estimation of $\vec{w}$ assuming that the observed training set was generated by a binomial model that depends on the output of the classifier.



In this case, the solid and empty dots can be correctly classified by any number of linear classifiers. H1 (blue) classifies them correctly, as does H2 (red). H2 could be considered "better" in the sense that it is also furthest from both groups. H3 (green) fails to correctly classify the dots.

- Perceptron—an algorithm that attempts to fix all errors encountered in the training set

- Support vector machine—an algorithm that maximizes the margin between the decision hyperplane and the examples in the training set.

Note: Despite its name, LDA does not belong to the class of discriminative models in this taxonomy. However, its name makes sense when we compare LDA to the other main linear dimensionality reduction algorithm: principal components analysis (PCA). LDA is a supervised learning algorithm that utilizes the labels of the data, while PCA is an unsupervised learning algorithm that ignores the labels. To summarize, the name is a historical artifact.[4]:117

Discriminative training often yields higher accuracy than modeling the conditional density functions. However, handling missing data is often easier with conditional density models.

All of the linear classifier algorithms listed above can be converted into non-linear algorithms operating on a different input space $\varphi(\vec{x})$, using the kernel trick.

2.1 Discriminative training

Discriminative training of linear classifiers usually proceeds in a supervised way, by means of an optimization algorithm that is given a training set with desired outputs and a loss function that measures the discrepancy between the classifier's outputs and the desired outputs. Thus, the learning algorithm solves an optimization problem of the form[1]

$$\arg\min_{\mathbf{w}} R(\mathbf{w}) + C \sum_{i=1}^{N} L(y_i, \mathbf{w}^\mathsf{T}\mathbf{x}_i)$$

where

- $\mathbf{w}$ is a vector of classifier parameters
- $L(yi,\ \mathbf{w}^\mathsf{T}\mathbf{x}i)$ is a loss function that measures the discrepancy between the classifier's prediction and the true output y_i for the i'th training example,
- $R(\mathbf{w})$ is a regularization function that prevents the parameters from getting too large (causing overfitting), and
- C is a scalar constant (set by the user of the learning algorithm) that controls the balance between the regularization and the loss function

Popular loss functions include the hinge loss (for linear SVMs) and the log loss (for linear logistic regression). If the regularization function R is convex, then the above is a convex problem.[1] Many algorithms exist for solving such problems; popular ones for linear classification include (stochastic) gradient descent, L-BFGS, coordinate descent and Newton methods.

3 See also

- Linear regression
- Winnow (algorithm)
- Quadratic classifier
- Support vector machines
- Perceptron
- Backpropagation

4 Notes

[1] Guo-Xun Yuan; Chia-Hua Ho; Chih-Jen Lin (2012). "Recent Advances of Large-Scale Linear Classification". *Proc. IEEE* **100** (9).

[2] T. Mitchell, Generative and Discriminative Classifiers: Naive Bayes and Logistic Regression. Draft Version, 2005 download

[3] A. Y. Ng and M. I. Jordan. On Discriminative vs. Generative Classifiers: A comparison of logistic regression and Naive Bayes. in NIPS 14, 2002. download

[4] R.O. Duda, P.E. Hart, D.G. Stork, "Pattern Classification", Wiley, (2001). ISBN 0-471-05669-3

See also:

1. Y. Yang, X. Liu, "A re-examination of text categorization", Proc. ACM SIGIR Conference, pp. 42–49, (1999). paper @ citeseer

2. R. Herbrich, "Learning Kernel Classifiers: Theory and Algorithms," MIT Press, (2001). ISBN 0-262-08306-X

5 Text and image sources, contributors, and licenses