# Intro to Machine Learning and Binary Classification

An aggregation of class notes and Lugosi notes

*Aimee Barciauskas*

*2016-02-08*

## Machine Learning

Machine learning is about constructing automatic rules that help find structure in data. In this course we same a few prototypical problems, learn about mathematical modelling, techniques and limitations.

**We'll discuss:**

- classification
- regression
- clustering
- on-line learning

In particular, we try to cover most of the following topics:

- Basic decision theory, bayes classifiers
- Non-parametric classifiers, consistency, nearest neighbor classifiers
- Empirical risk minimization, VC-theory
- Linear Classifiers, support vector machines, perceptron, classification trees
- Principal Component Analysis
- Least squares regression, Lasso
- Clustering: k-means, single linkage, spectral clustering
- Online-learning, prediction with expert advice, Bandit problems
- Stochastic Gradient Ascent

**Tools for this course**

Probability, linear algebra, optimization, algorithmic complexity

**Evaluation:**

- 50% final exam
- 25% from project (correction from website)
- 25% bi-weekly exercises

**Recommended books:**

- Duda, Hart, and Stork, Pattern Classification, 2nd Edition, Wiley, 2000
- Devroye, Györfi, Lugosi, A Probabilistic Theory of Pattern Recognition, Springer, 1996
- Tom Mitchell, Machine Learning, McGraw Hill, 1997
- V. N. Vapnik, Statistical learning theory, 1998

- C. M. Bishop, Pattern Recognition and Machine Learning, Springer, 2007
- Kearns and Vazirani, Introduction to Computational Learning Theory (MIT Press 1994) J. Shawe-Taylor and N Crisianini, Kernel Methods for Pattern Analysis, Cambridge University Press, * 2004 N. Cesa-Bianchi, and G. Lugosi, Prediction, Learning, and Games. Cambridge University Press, New * York, 2006.
- Steinwart and Christmann, Support Vector Machines. Springer, 2008.
- Kevin Patrick Murphy, Machine Learning: a Probabilistic Perspective, MIT Press, 2012.
- Yoshua Bengio, Ian Goodfellow and Aaron Courville, Deep learning. In preparation for MIT Press, 2014.
- John Hopcroft and Ravi Kannan, Foundations of Data Science. In preparation, 2014. Shai Shalev-Shwartz and Shai Ben-David. Understanding Machine Learning. From Theory to Algorithms, * Cambridge University Press, 2014.

### What is machine learning? (According to wikipedia)

> Machine learning is a subfield of computer science that evolved from the study of pattern recognition and computational learning theory in artificial intelligence. Machine learning explores the study and construction of algorithms that can learn from and make predictions on data.

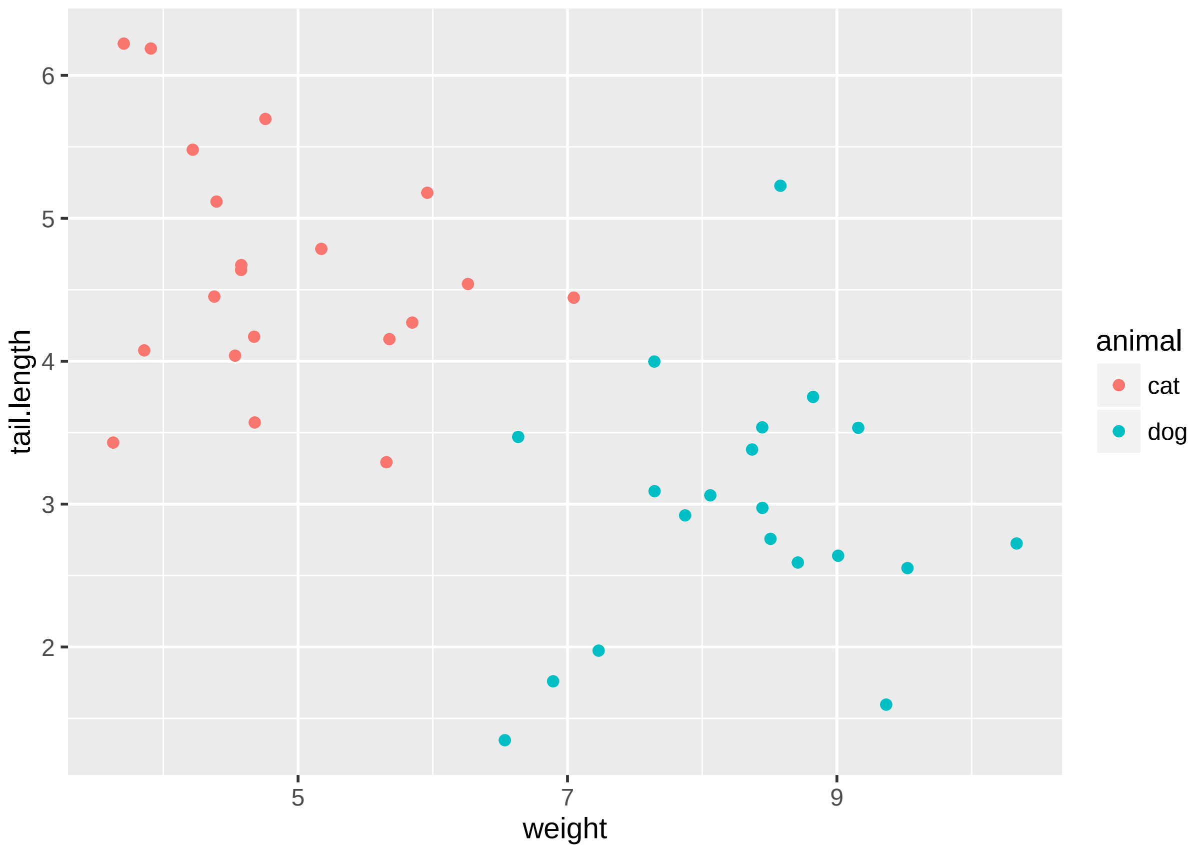
## Binary Classification

### What is a classifier?

A classifer assigns classes discrete values.

$$g : \mathcal{X} \to \{0, 1\}$$

Suppose we are given a training sample of dogs and cats.



Given an *unknown animal*, the problem is to classify whether it is a dog or a cat.

First we need to define what exactly we want to achieve and what we mean by good classifier.

In order tod this we introduce a mathematical framework:

Let $\mathcal{X}$ denote the set of observations. Given an observation $x \in \mathcal{X}$, we want to assign it to one of the two classes, say 0 or 1. Formally a classifier is a function $g : \mathcal{X} \to \{0, 1\}$

The problem can be set up as a statistical hypothesis testing problem.

**How do we measure a good classifier?**

The probability of making a mistake ($R(g) = \mathbb{P}(g(X) \neq Y)$), plus, optionally, the consideration of a loss function.

$$R^l(g) = \mathbb{E}\ell(g(X), Y) = \mathbb{E}\mathbb{I}_{g(X)\neq Y}$$

In this framework one assumes that the observation $X$ is random, and it has different (conditional) distributions depending on which of the two classes it belongs to (i.e. *cat* or *dog*).

Formally, $(X, Y)$ is a pair of random variables taking values in $\mathcal{X} \times \{0, 1\}$. $X$ represents the observation and $Y$ its class (or label).

The joint distribution of $(X, Y)$ may be described in various ways. It is given by the pair $(\mu, \eta)$ where $\mu$ is the distribution of $X$, that is,

$$\mu(A) = \mathbb{P}(X \in A) \text{ for all } A \subset \mathcal{X}$$

and $\eta : \mathcal{X} \to [0, 1]$ is the a posterior probability

$$\eta(x) = \mathbb{P}(Y = 1|X = x)$$

$$\mathbb{E}(X|Z) \rightarrow \text{ a random variable}$$
$$\mathbb{E}(X|Z = z) \rightarrow \text{ a value}$$
$$\mathbb{E}[\mathbb{E}(X|Z = z)] = \mathbb{E}(X)$$
$$\mathbb{P}(X \in A) = \mathbb{E}\mathbb{I}_{X\in A}$$

We may also define the a priori probabilties:

$$q_0 = \mathbb{P}(Y = 0), q_1 = \mathbb{P}(Y = 1)$$

and the class-conditional distributions

$$\mathbb{P}(X \in A|Y = 0); \mathbb{P}(X \in A|Y = 1) \text{ for } A \subset \mathcal{X}$$

Given a classifier $g : \mathcal{X} \rightarrow \{0, 1\}$, we may measure its quality by the probability of error:

$$R(g) = \mathbb{P}(g(X) \neq Y)$$

Note: $R(g) = \mathbb{P}(g(X) = 1|Y = 0)q_0 + \mathbb{P}(g(X) = 0|Y = 1)q_1$ measures the two types of errors in a symmetric way. This may not be natural (or wise) in some applications. More generally, we may have a **loss function:**

$$\ell : \{0, 1\} \times \{1, 0\} \rightarrow \mathbb{R}_+ \text{ measuring the cost of different types of errors}$$

The risk becomes $R(g) = \mathbb{E}\ell(g(X), Y)$. For simplicity, we only consider the probability of error. The first natural question is to determine the classifier that minimizes the probability of error. This can be done easily: define the **Bayes classifier** by:

$$g^* = \begin{cases} 1 \text{ if } \eta(x) > \frac{1}{2} \\ 0 \text{ otherwise} \end{cases}$$

**Theorem:** The Bayes classifier minimizes the probability of error. That is for any classification $g$,

$$R(g) \geq R(g^*)$$

*Proof:* For any fixed $x \in \mathcal{X}$,

$$\begin{aligned} \mathbb{P}(g(X) \neq Y|X = x) &= 1 - \mathbb{P}(g(X) = 1, Y = 1|X = x) - \mathbb{P}(g(X) = 0, Y = 0|X = x) \\ &= 1 - \mathbb{I}_{g(X)=1}\eta(x) - \mathbb{I}_{g(X)=0}(1 - \eta(x)) \end{aligned}$$

Therefore,

$$\begin{aligned} &\mathbb{P}(g(X) \neq Y|X = x) - \mathbb{P}(g^*(X) \neq Y|X = x) \\ &= \eta(x)(\mathbb{I}_{g^*=1} - \mathbb{I}_{g(x)=1}) + \eta(x - 1)(\mathbb{I}_{g^*=0} - \mathbb{I}_{g(x)=0}) \\ &= (2\eta(x) - 1)(\mathbb{I}_{g^*(x)=1} - \mathbb{I}_{g(x)=1} \geq 0 \end{aligned}$$

$R^* = R(g)$ is called the **Bayes Risk**

$R^* = 0$ means that the two classes are *seperable.* $R^* = \frac{1}{2}$ means complete overlap.

The proof shows that:

$$R(g) = 1 - \mathbb{E}[\mathbb{I}_{g(x)=1}\eta(x)] - \mathbb{E}[\mathbb{I}_{g(x)=0}(1-\eta(x))]$$

In particular,

$$R(g) = 1 - \mathbb{E}[\mathbb{I}_{\eta(x)>1}\eta(x)] - \mathbb{E}[\mathbb{I}_{\eta(x)\leq 1}(1-\eta(x))]$$

Thus, the optimal classifier is easy to determine. However, it requires the knowledge of the function $\eta(x)$ which is rarely available. Typically all we have is training data.

$$R(g^*) = \mathbb{E}[min(1-\eta(x), \eta(x))]$$

**Jensen's Inequaltiy:** If $\phi$ is a convex function,

$$\mathbb{E}(\phi(x)) \geq \phi(\mathbb{E}(x))$$

So,

$$R^* \leq min[\mathbb{E}\eta(x), \mathbb{E}(1-\eta(x))] = min(\mathbb{P}(Y=1), \mathbb{P}(Y=0))$$
$$= \frac{1}{2} \text{ if } \mathbb{P}(Y=1) == \mathbb{P}(Y=0)$$

*The maximum of convex function is convex. The minimum of a concave function is concave*

## Plug-in Classifiers

The a posteriori probabilities $\eta(x)$ are usually unknown. If an approximation $\widetilde{\eta(x)}$ is available, it is natural to define the *plug-in classifier:*

$$\widetilde{g} = \begin{cases} 1 \text{ if } \widetilde{\eta(x)} > \frac{1}{2} \\ 0 \text{ otherwise} \end{cases}$$

From the proof of the previous theorem we see that:

$$R(\widetilde{g}) - R^*$$
$$\leq 2\mathbb{E}|\eta(x) - \widetilde{\eta(x)}|\mathbb{I}_{\widetilde{g(X) \neq g^*}(X)}$$
$$\leq 2\mathbb{E}|\eta(x) - \widetilde{\eta(x)}|$$

*Classification is "easier" than regression!*

Note that $\eta(x) = \mathbb{E}(Y|X=x)$ is just the regrssion function.

Training data is modeled as independent pairs of random variables $(X_1, Y_1), ..., (X_n, Y_n)$ drown from the same distribution as $(X, Y)$

A **data-based classifier** is now a function

$$g_n(x) = g_n(x, D_n) \text{ with } D_n = ((X_1, Y_1), ..., (X_n, Y_n))$$

It is random as it depends on the data.

The probability of error

$$R(g_n) = \mathbb{P}(g_n(X) \neq Y | D_n) \textit{ conditional probability!}$$

is therefore a random variable. Clearly, $R(g_n \geq R^*)$.

We would like to construct classifiers (i.e. data-dependent rules $g_n$) such that $R(g_n)$ is "small."

*Problem:* We don't know *anything* about the distribution – though sometimes we may be willing to assume something about it. Can this be achieved? *Another problem:* What does it mean that a random variable is "small"?

Possible meaningful goals include constructing a classifier for which $\mathbb{E}R(g_n)$ is close to $R^*$ for a large class of distributions when $n$ is large.

## Consistency

A sequence of classifiers $\{g_n\}$ is **consistent** (for a certain distribution of $(X, Y)$) if

$$\lim_{n\to\infty} \mathbb{E}R(g_n) = R^*$$

It is **universally consistent** if it is consistent for all possible distributions.

This notion of consistency only concerns the expected value. **Strong consistency** requires that

$$\lim_{n\to\infty} R(g_n) = R^* \text{ with probability 1 .}$$

We will also be interested in statements of the form

$$\mathbb{P}(R(g_n) > \bar{R} + \epsilon) < \delta$$

where $\bar{R}$ is the **best risk in a family** of classifiers. Here, we'd like to keep $\epsilon, \delta$ as small as possible. Other useful bounds we'd like to have are of the form

$$\mathbb{P}(R(g_n) > \hat{R}_n + \epsilon) < \delta$$

where $R(\hat{g_n})$ is an **empirical quantity** that one can compute from the data $D_n$

There are many aspects of machine learning that we *do not* discuss in this course:

- reinforcement learning
- graphical models, Bayesian networks
- neural networks
- deep learning
- feature extraction (e.g. image identification), representation learnging
- learning from dependent data (i.e. time series)
- manifold learning, dimensionality reduction
- density estimation

- multi-class classification
- multitask learning
- independent component analysis
- matrix completion, recommendation systems
- community detection