Chapter 1
Introduction

1.1 Types of machine learning

- Supervised learning
  - Classification
  - Regression
- Unsupervised learning
  - Discovering clusters
  - Discovering latent factors
  - Discovering graph structure
  - Matrix completion

1.2 Three elements of a machine learning model

Model = Representation + Evaluation + Optimization[1]

1.2.1 Representation

In supervised learning, a model must be represented as a conditional probability distribution $P(y|\boldsymbol{x})$(usually we call it classifier) or a decision function $f(x)$. The set of classifiers(or decision functions) is called the hypothesis space of the model. Choosing a representation for a model is tantamount to choosing the hypothesis space that it can possibly learn.

1.2.2 Evaluation

In the hypothesis space, an evaluation function (also called objective function or risk function) is needed to distinguish good classifiers(or decision functions) from bad ones.

1.2.2.1 Loss function and risk function

Definition 1.1. In order to measure how well a function fits the training data, a **loss function** $L : Y \times Y \to R \geq 0$ is defined. For training example (x_i, y_i), the loss of predicting the value $\widehat{y}$ is $L(y_i, \widehat{y})$.

The following is some common loss functions:

1. 0-1 loss function
   $$L(Y, f(X)) = \mathbb{I}(Y, f(X)) = \begin{cases} 1, & Y = f(X) \\ 0, & Y \neq f(X) \end{cases}$$
2. Quadratic loss function $L(Y, f(X)) = (Y - f(X))^2$
3. Absolute loss function $L(Y, f(X)) = |Y - f(X)|$
4. Logarithmic loss function
   $L(Y, P(Y|X)) = -\log P(Y|X)$

Definition 1.2. The risk of function f is defined as the expected loss of f:

$$R_{\exp}(f) = E\left[L(Y, f(X))\right] = \int L(y, f(x))\, P(x, y)\mathrm{d}x\mathrm{d}y \tag{1.1}$$

which is also called expected loss or **risk function**.

Definition 1.3. The risk function $R_{\exp}(f)$ can be estimated from the training data as

$$R_{\text{emp}}(f) = \frac{1}{N} \sum_{i=1}^{N} L(y_i, f(x_i)) \tag{1.2}$$

which is also called empirical loss or **empirical risk**.

You can define your own loss function, but if you're a novice, you're probably better off using one from the literature. There are conditions that loss functions should meet[2]:

1. They should approximate the actual loss you're trying to minimize. As was said in the other answer, the standard loss functions for classification is zero-one-loss (misclassification rate) and the ones used for training classifiers are approximations of that loss.
2. The loss function should work with your intended optimization algorithm. That's why zero-one-loss is not used directly: it doesn't work with gradient-based optimization methods since it doesn't have a well-defined gradient (or even a subgradient, like the hinge loss for SVMs has).
   The main algorithm that optimizes the zero-one-loss directly is the old perceptron algorithm(chapter §**??**).

[1] Domingos, P. A few useful things to know about machine learning. Commun. ACM. 55(10):7887 (2012).

[2] http://t.cn/zTrDxLO

1.2.2.2 ERM and SRM

Definition 1.4. ERM(Empirical risk minimization)

$$\min_{f\in\mathcal{F}} R_{\text{emp}}(f) = \min_{f\in\mathcal{F}} \frac{1}{N}\sum_{i=1}^{N} L(y_i, f(x_i)) \tag{1.3}$$

Definition 1.5. Structural risk

$$R_{\text{smp}}(f) = \frac{1}{N}\sum_{i=1}^{N} L(y_i, f(x_i)) + \lambda J(f) \tag{1.4}$$

Definition 1.6. SRM(Structural risk minimization)

$$\min_{f\in\mathcal{F}} R_{\text{srm}}(f) = \min_{f\in\mathcal{F}} \frac{1}{N}\sum_{i=1}^{N} L(y_i, f(x_i)) + \lambda J(f) \tag{1.5}$$

1.2.3 Optimization

Finally, we need a **training algorithm**(also called **learning algorithm**) to search among the classifiers in the the hypothesis space for the highest-scoring one. The choice of optimization technique is key to the **efficiency** of the model.

1.3 Some basic concepts

1.3.1 Parametric vs non-parametric models

1.3.2 A simple non-parametric classifier: K-nearest neighbours

1.3.2.1 Representation

$$y = f(\boldsymbol{x}) = \arg\min_{c} \sum_{\boldsymbol{x}_i\in N_k(\boldsymbol{x})} \mathbb{I}(y_i = c) \tag{1.6}$$

where $N_k(\boldsymbol{x})$ is the set of k points that are closest to point $\boldsymbol{x}$.

Usually use **k-d tree** to accelerate the process of finding k nearest points.

1.3.2.2 Evaluation

No training is needed.

1.3.2.3 Optimization

No training is needed.

1.3.3 Overfitting

1.3.4 Cross validation

Definition 1.7. Cross validation, sometimes called *rotation estimation*, is a *model validation* technique for assessing how the results of a statistical analysis will generalize to an independent data set[3].

Common types of cross-validation:

1. K-fold cross-validation. In k-fold cross-validation, the original sample is randomly partitioned into k equal size subsamples. Of the k subsamples, a single subsample is retained as the validation data for testing the model, and the remaining k 1 subsamples are used as training data.
2. 2-fold cross-validation. Also, called simple cross-validation or holdout method. This is the simplest variation of k-fold cross-validation, k=2.
3. Leave-one-out cross-validation(*LOOCV*). k=M, the number of original samples.

1.3.5 Model selection

When we have a variety of models of different complexity (e.g., linear or logistic regression models with different degree polynomials, or KNN classifiers with different values ofK), how should we pick the right one? A natural approach is to compute the **misclassification rate** on the training set for each method.

[3] http://en.wikipedia.org/wiki/Cross-validation_(statistics)