Purdue University, CS/ECE662
Pattern Recognition and Decision Making Processes

Lecture Notes

Prof. Mireille Boutin

Spring 2022

Contents

1 Foundations

1.1 What is Pattern Recognition?

In the signal processing community, pattern recognition is typically defined as the process of deciding between mutually exclusive alternatives called “classes" or “patterns" (e.g. in [5]). For example, detection is a type of pattern recognition where one is trying to determine whether a given something (e.g., a plane, a disease, a defect) is present or not. In this cases, there are two classes: present and not present.

In statistics, pattern recognition is viewed as the process of predicting a random categorical outcome, and the classes are known as “hypotheses" (e.g. in [7]). More specifically, we are given data which is viewed as a random sample x of a random variable X, called the input variable (also “predictor" or, more classically, “independent variable."). The class corresponding to the random sample is viewed as a random sample ω of a random variable Ω, called the output variable (also “response" or, more classically, “dependent variable.").

In either case, the goal is to find a rule, usually a deterministic function f, to estimate the class ω given the data x. In estimation theory, the value $\hat{\omega}$ of the function at a sample x is called an “estimator" :

$$\hat{\omega} = f(x).$$

In classification problems, ω takes on discrete (categorical) values, usually within a finite set. When ω takes on continuous values, the problem is known as “regression."

Notice the subtle, but important difference between the signal processing and the statistical point of view. In the second case, classes are explicitly viewed as random variables (though they are often modeled as such in the first case as well). Not all classification problems fit this assumption. For example, when manufacturing a piece of hardware, there could be potentially infinitely many different possible defects which would be cause for rejection. The process generating these defects may not be a random process. In other words, there may not be a probability law from which the defects are drawn. Recall that, for a variable to be a random variables, there needs to be a well-defined function from the set of possible outcomes to a measurable space. Such a function does not necessarily exists. For instance, the process controlling the quality of the pieces may vary so unpredictably from one moment to the next that there is no consistency in the process, no level reproducibility in the outcome, so that the process cannot be modeled.

However, the foundation of machine-learning based pattern recognition methods (including neural networks) lie in theories phrased in probabilistic terms. Thus the rationale for the use of these methods falls apart when statistical modeling is not possible. While this does not

necessarily mean that the such pattern recognition problems cannot be solved, one needs to thread carefully.

1.2 Example of Pattern Recognition Problems

There are two main categories of pattern recognition problems, called “supervised" and “unsupervised." A hybrid category called “semi-supervised" also exists.

Supervised pattern recognition problems are those where the classes are predetermined. For example

- Determine if a patient has heart disease;
- Determine if a building on a foreign territory map is a school;
- Predict whether a criminal will relapse into crime if granted parole;
- Determine if a delivery robot should stop or continue;
- Identify the ZIP code on a letter;
- Identify airport patrons about to commit a terrorist act;
- Determine the next move when solving a Rubik cube;
- Determine if a positive integer is prime;
- Determine if a picture contains a cat;
- Identify spam email;
- Identify if a credit card transaction is fraudulent.

Exercise: For each of these problems, discuss whether the input data and the classes can be modeled as random variables. □

Unsupervised pattern recognition problems are those where the classes are not predetermined and must be derived from the data. For example:

- Define species within groups of animals;
- Segment an image into its different objects and backgrounds;
- Organize a database of objects for efficient retrieval;
- Divide integers into groups with shared properties.

Unsupervised pattern recognition has traditionally been equated with “clustering." However, the intuition between the term clustering is misleading, as the classes need not consist of objects that are particularly “close" together. The case of even/odd integers, which is a perfectly valid way to group integers into two well-defined similarity classes, illustrates this clearly. While

objects in the same class share a similar characteristics (divisibility by two), they are not close in any way. In other words, pattern recognition is about similarity than proximity.

Exercise: Give other examples of classes of objects that are well-defined yet do not correspond to “clusters" or “proximity" in the input data. □

1.3 Classification by Look-up

When the set of possible values for the input variable x is finite, then it may be possible to list the decision to be made in all possible cases. For example, if the input data consists in 4 binary valued pixels in an image, then the set of possibilities for the input contains $2^4 = 16$ cases. If the decisions to be made for each of these 16 cases is determined (e.g., by an expert or with machine learning techniques), then they can be stored in a look-up table for later retrieval. In such case, the function f need not be assumed to take any particular form.

When the look-up approach is viable, it should not be dismissed. Today’s electronic machines often allow fairly large look-up tables to be consulted quickly. Thus many of them use look-ups to make decisions, as the trade-off between computation and memory is often positive.

For example, we have developed a look-up based method to classify edges within color images [10]. This was part of a *color trapping* project funded by the Hewlett Packard company. Color trapping is a solution to the problem of color plane misalignment during printing, which cause the color at the edge of objects to be inconsistent with the color inside the object. Part of the solution to this problem involves identify pixels that lie on the edge of a colored area, and classifying the edges into one of several types based on their shape and the colors they involve. After quantizing the pixel values to 8 bits and normalizing the orientation of a 5-by-5 patch surrounding the pixel, the look-up table required to store all the possible edge cases used only 3.7 MBytes of memory.

1.4 Classification by Extrapolation

When the the set of possible values for the input variable x is infinite, or when it is finite but of too large a size for all the cases to be separately analyzed and/or stored in a look-up table, then the function f must be constructed somehow.

Once can do this by relying on understanding the principles that control the decision. For example, if one needs to decide if an integer is even or odd, then it would be impossible to build a look-up table with the parity of all integers. However, we know the rule to make this decision, and so building the required function f does not pose a challenge. For example, using 0 as the label for even integers and 1 for odd integers, we can set $f(x) = x \bmod 2$.

When the rules to make the decision are unknown, or only partially known, one can attempt to build the function f by extrapolating from known examples. This approaches requires to assume some level of continuity/regularity/consistency in the pattern distribution. For example,

it is not possible to decide if an integer is prime or not by extrapolating from known examples, as prime integers tend to be isolated and their distribution follows no specific pattern (only asymptotically). But it may be a viable approach if points from the same class tend to be grouped together. In particular, this is a reasonable assumption if the classes can be viewed as one ideal prototype pattern perturbed by a large number of independent noise processes [6]. Indeed, by the Central Limit Theorem, the combined effects of all these independent variables is approximately Gaussian.

1.5 Evaluation of Pattern recognition Methods

If the space of possible values for the input is finite and small, one can assess the accuracy of a given pattern recognition method by evaluating its accuracy at every single point of the input space.

If the space is infinite, or if it is too large, one can try to use a sampling strategy to estimate the accuracy of the classifier. Again, this can only work under some kind of “continuity" or “consistency" assumption.

One approach to do this is to assume that the input is a random variable. More specifically, one needs to be given some input values, with their corresponding class (e.g. the “test data"), and be able to assume that these input values correspond to independent samples from some distribution. A popular approach is to compute the accuracy of the classifier on the test data. Since the test data is random, the accuracy obtained is also a random number. Therefore, this test is usually repeated several times, and the empirical average and standard deviation of the test accuracy is reported.

There is, however, a strong theoretical basis for using the test accuracy as an estimate for the accuracy of a classifier. This theory relies on concentrations inequalities in probability theory. Let me explain.

Let Z_i, $i = 1, \ldots, n$ be a binary valued random variable representing the success ($Z_i = 1$) or failure ($Z_i = 0$) of the classification of test input value i. Then $Z = \frac{1}{n} \sum_{i=1}^{n} Z_i$ is a measure of the accuracy of the classifier on the test data. The quantity Z is not fixed, but rather random. Each random variable Z_i is an identically distributed Bernoulli random variable with parameter p equal to the population accuracy of the classifier. Thus nZ is a binomial random variable with mean np and variance $np(1-p)$.

Concentration inequalities quantify to what extent a random variable Y deviates from its mean μ, and so they can be used to quantify to what extent the test accuracy deviates from the population accuracy of a classifier. They usually take the form [9]

$$\mathbb{P}\{|Y - \mu| > \epsilon\} < \text{something small.}$$

A basic concentration inequality is Chebyshev’s Inequality

Theorem 1.1. *(Chebyshev's Inequality) Let Y be a random variable with finite mean μ and variance σ^2. Then*

$$\mathbb{P}\{|Y-\mu| \geq \epsilon\} \leq \frac{\sigma^2}{\epsilon^2},$$

for any $\epsilon > 0$

Proof. This inequality follows from applying Markov's inequality to the random variable $(Z-\mu)^2$.

Markov's inequality states that if a random variable Y' taking values in $\mathbb{R}$ is non-negative and has finite mean μ', then for any $\epsilon > 0$,

$$\mathbb{P}\{Y' \geq \epsilon\} \leq \frac{\mu}{\epsilon}.$$

Setting $Y' = (Z-\mu)^2$, we have $\mu' = E\{(Z-\mu)^2\} = \sigma^2$ and the conclusion follows. □

Applying Chebyshev's Inequality to the test accuracy measure Z, we obtain

$$\mathbb{P}\{|Z-p| \geq \epsilon\} \leq \frac{p(1-p)}{n\epsilon^2}, \text{ for any } \epsilon > 0.$$

Thus, as the number of samples n grows to infinity, the probability that the test accuracy is not within an ϵ of the classifier accuracy p goes to zero. The bound decreases to zero is inversely proportional to the number of test points.

Note that the bound on the probability that the test accuracy is within ϵ of the true accuracy p of the classifier depends on p, which is unknown. We can make the bound independent of p by observing that $p(1-p) \leq \frac{1}{4}$. Thus we have

$$\mathbb{P}\{|Z-p| \geq \epsilon\} \leq \frac{1}{4n\epsilon^2}, \text{ for any } \epsilon > 0.$$

This inequality holds for any classifier, and for any data distribution.

Another important concentration inequality is Hoeffding's inequality.

Theorem 1.2. *(Hoeffding's Inequality) Let $Y_1, \ldots, Y_n$ be random variables such that, for all $i = 1, \ldots, n$, there exists finite $a_i, b_i \in \mathbb{R}$ such that $a_i \leq Y_i \leq b_i$ with probability one. Let $Y = \sum_{i=1}^n Y_i$. Then for every $\epsilon > 0$, we have*

$$\mathbb{P}\{|Y - E(Y)| \geq \epsilon\} \leq 2e^{\frac{-2\epsilon^2}{\sum_i (a_i - b_i)^2}}.$$

It the Y_i are Bernoulli random variables, this boils down to

$$\mathbb{P}\{|Y - E(Y)| \geq \epsilon\} \leq 2e^{\frac{-2\epsilon^2}{n}}.$$

Dividing by n:

$$\mathbb{P}\left\{\left|\frac{Y}{n} - \frac{E(Y)}{n}\right| \geq \frac{\epsilon}{n}\right\} \leq 2e^{\frac{-2\epsilon^2}{n}}.$$

We have $Z = \frac{Y}{n}$ and the population accuracy $p = E(Z) = \frac{E(Y)}{n}$. Setting $t = \frac{\epsilon}{n}$ yields

$$\mathbb{P}\{|Z - p| \geq t\} \leq 2e^{-2nt^2}.$$

(This is one of many things called “Chernoff’s bound") This means that the difference between the probability that the test accuracy and the population accuracy of a classifier differ by more than t decreases (at least) exponentially with the number of samples n. This time, the the bound on the probability that the test accuracy is within ϵ of the true accuracy p of the classifier only depends on n, and no other unknown parameters. The bounds decreases to zero exponentially with respect to the number of test points.

Again, this inequality holds for any classifier, and for any data distribution.

1.6 The Statistical Machine Learning Paradigm

In the empirical view of unsupervised pattern recognition, one is given some data points $p_1, \ldots, p_n$. These data points are viewed as points belonging to a vector space equipped. The goal is to partition the the set of points into subsets that form meaningful structures in the vector space. For example, one might equip the vector space with a metric and ask that points in the same subset be “closer," in some sense, to points within the same subset than points within other subsets.

The statistical point of view models each p_i as a random sample x_i in a probability space S. The goal is to build a classifier $\hat{\omega} = f(x)$ which can be used to classify any point x of the probability space S. So rather than merely grouping the data at hand, we wish to obtain a rule to be able to group further random samples. The emphasis is on the future performance of the classifier.

$$\text{data points} \longrightarrow \begin{array}{c}\text{samples}\\ x_1, \ldots, x_n \in S\\ \text{of a r.v. X}\end{array} \longrightarrow \boxed{\text{Machine Learning}} \rightarrow \begin{array}{c} f\\ \text{classifier}\end{array}$$

The statistical view of learning with supervision is similar. The emphasize is still on building a classifier that will classify future points accurately. The difference is that the data points used to learn are already classified; both the point and their classes are then viewed as random samples.

$$\begin{array}{c}\text{classified}\\ \text{data points}\end{array} \longrightarrow \begin{array}{c}\text{samples}\\ (x_1, \omega_1) \ldots, (x_n, \omega_n)\\ \text{of r.v.}(X, \Omega)\end{array} \longrightarrow \boxed{\text{Machine Learning}} \rightarrow \begin{array}{c} f\\ \text{classifier}\end{array}$$

1.7 Appeal of the Black Box Approach

Antoine De Saint-Exupéry, a wise man, explained the appeal of the black box approach in his famous book: "Le Petit Prince." In the story, the author, who is the main protagonist, is stranded

in the desert after his plane suffers an engine failure. While attempting a repair, he is visited by the little prince, who recently arrived on planet Earth.

The little prince asks him "Please, draw me a sheep." The author is very surprised at this request, and initially protests, remembering his struggles as a child whose drawings were misunderstood by adults. He eventually obliges and does his best to draw a sheep. The little prince looks at the result, and complains that the sheep is too sick. He requests another one.

The author draws another one, this time with horns, which makes the little prince smile. "That's not a sheep. It's a ram."

The third attempt is also rejected. This time the little prince deems the sheep too old; he wants one that will live a long time.

So the author, who is getting impatient, finally draws a crate and says: "The sheep you want is inside." He even draws some breathing holes on the side of the crate. Then the little prince exclaims: "That's exactly the sheep I wanted."

1.8 Discriminant Function Versus Decision Boundary

In the case where there are two classes to distinguish (binary classification), there are two related but different points of view. The first one seeks to construct a discriminant function $g(x) : S \longrightarrow \mathbb{R}$ which will be used to classify data points as follows [4]:

$$\begin{aligned} &\text{if } g(x) > 0, \qquad \text{decide class 1,} \\ &\text{if } g(x) < 0, \qquad \text{decide class 2.} \end{aligned}$$

Observe that the discriminant function defines two regions of the feature space

$$\begin{aligned} S_1 &= \{x | g(x) > 0\}, \\ S_2 &= \{x | g(x) < 0\}, \end{aligned}$$

each associated to a different class label. If the feature space is $\mathbb{R}^d$ and g is a continuous function, then the two regions are separated by boundary points where g vanishes. An alternative point of view in this case is to focus on finding the boundary between the regions

$$\{x | g(x) = 0\}.$$

1.9 Two simple examples of classifiers

We now present two simple examples of classification methods, which we will use to illustrate our further discussion. Although both methods are strictly empirical, we will later see that they can be viewed in statistical terms.

Nearest Neighbor Classification Method

Consider a supervised classification problem where one is asked to classify a data point $x \in S$ into one of c classes, denoted by the labels $\Omega = \{1, 2, \ldots, c\}$. Given are n data points with known classes:

$$(x_i, \omega_i), \quad , i = 1, \ldots, n$$

with $x_i \in S$ and $\omega_i \in \Omega$. Assume that the space S in which the points lie is equipped with a metric d:

$$d : S \times S \to \mathbb{R}_{\geq 0}.$$

That is to say, a function d such that, for all x, x', x'', we have

1. $d(x, x) \geq 0$ and $d(x, x') = 0 \Leftrightarrow x = x'$,
2. $d(x, x') = d(x', x)$,
3. $d(x, x') + d(x', x'') \geq d(x, x'')$.

The "nearest neighbor classification rule" is to assign x to the class of the nearest point among $x_1, \ldots, x_n$:

$$\hat{\omega} = \omega_{i^*}, \text{ where } i^* = \underset{i=1,\ldots,n}{\arg\min}\, d(x, x_i).$$

We can also describe the classifier using a discriminant function, for example

$$g(x) = \min_{i s.t. \omega_i = 1} d(x, x_i) - \min_{i s.t. \omega_i = 2} d(x, x_i)$$

Notice that this classifier makes no mention of probability. In particular, it does not assume that the points x_i are samples drawn following a distribution. We will later see that this classifier can be viewed from a probability standpoint.

Linear Separation Between Means

This time, we consider a supervised classification problem where one is asked to classify a data point $x \in S$ into one of 2 classes, denoted by the labels $\Omega = \{1, 2\}$. Given are n data points with known classes:

$$(x_i, \omega_i), \quad i = 1, \ldots, n$$

with $x_i \in S$ and $\omega_i \in \Omega$. Assume that S is a vector space equipped with a real-valued inner product $\cdot : S \times S \to \mathbb{R}$.

Without loss of generality, assume $\omega_1 = \omega_2 = \ldots = \omega_{n_1} = 1$ and $\omega_{n_1+1} = \ldots = \omega_n = 2$. Let

$$\mu_1 = \frac{1}{n_1} \sum_{i=1}^{n_1} x_i \quad \text{and} \quad \mu_2 = \frac{1}{n - n_1} \sum_{i=n_1}^{n} x_i.$$

If $\mu_1 \neq \mu_2$, define the discriminant function

$$g(x) = (\mu_1 - \mu_2) \cdot x - \frac{1}{2}(\mu_1 - \mu_2) \cdot (\mu_1 + \mu_2).$$

Thus we have the classification rule

$$\begin{aligned} \text{if} \quad & (\mu_1 - \mu_2) \cdot x - \frac{1}{2}(\mu_1 - \mu_2) \cdot (\mu_1 + \mu_2) > 0, \qquad \text{decide class 1,} \\ \text{if} \quad & (\mu_1 - \mu_2) \cdot x - \frac{1}{2}(\mu_1 - \mu_2) \cdot (\mu_1 + \mu_2) < 0, \qquad \text{decide class 2.} \end{aligned}$$

The separation between the classes is the hyperplane

$$(\mu_1 - \mu_2) \cdot x - \frac{1}{2}(\mu_1 - \mu_2) \cdot (\mu_1 + \mu_2) = 0$$

We can check that μ_1 (and, by continuity, the entire side containing μ_1) is classified as class 1 by checking that $g(\mu_1) > 0$. Similarly μ_2 (and, by continuity, the entire side containing μ_2) is classified as class 2 since $g(\mu_2) < 0$. Also, the midpoint $\frac{1}{2}(\mu_1 - \mu_2)$ is on the boundary since $g(\frac{1}{2}(\mu_1 - \mu_2)) = 0$. Furthermore the normal to the hyperplane is $(\mu_1 - \mu_2)$ since $(\mu_1 - \mu_2) \cdot (x_1 - x_2) = 0$ for any two points x_1, x_2 on the hyperplane $g(x_1) = g(x_2) = 0$.

So the separation hyperplane is the hyperplane passing through the half-point between the means $\frac{1}{2}(\mu_1 - \mu_2)$ that is perpendicular (in the sense given by the inner product) to the vector linking the means.

Connections between the two example classifiers

Consider the space vector S with inner product of the linear separation method, and equip it with the distance function induced by the inner product:

$$d(x, x') = \sqrt{(x - x') \cdot (x - x')}.$$

Then the linear separation method is the same as classification to the class of the nearest mean:

$$\hat{\omega} = \omega_{i^*}, \text{ where } i^* = \arg\min_{i=1,2} d(x, \mu_i).$$

Indeed we have

$$\begin{aligned} d(x, \mu_1) &\leq d(x, \mu_2) \\ \Leftrightarrow d^2(x, \mu_1) &\leq d^2(x, \mu_2) \\ \Leftrightarrow (x - \mu_1) \cdot (x - \mu_1) &\leq (x - \mu_2) \cdot (x - \mu_2) \\ \Leftrightarrow x \cdot x - 2x \cdot \mu_1 + \mu_1 \cdot \mu_1 &\leq x \cdot x - 2x \cdot \mu_2 + \mu_2 \cdot \mu_2 \\ \Leftrightarrow -2x \cdot \mu_1 + \mu_1 \cdot \mu_1 &\leq -2x \cdot \mu_2 + \mu_2 \cdot \mu_2 \\ \Leftrightarrow 0 &\leq -2x \cdot (\mu_2 - \mu_1) + \mu_2 \cdot \mu_2 - \mu_1 \cdot \mu_1 \\ 0 &\leq -x \cdot (\mu_2 - \mu_1) + \frac{1}{2}(\mu_2 \cdot \mu_2 - \mu_1 \cdot \mu_1) \\ \Leftrightarrow 0 &\leq x \cdot (\mu_1 - \mu_2) + \mu_2 \cdot \mu_2 + \frac{1}{2}(\mu_1 + \mu_2)(\mu_1 - \mu_2) \end{aligned}$$

So if we add a little bit more structure to the space S, this linear separation method become a shortest distance method.

Conversely, if we take the space S equipped with a metric d in the nearest neighbor classification and equip it with an inner product, then the method can be viewed as one where the decision boundary is a collection of hyperplane segments.

Arbitrariness of Structure

The choice of inner product or metric is arbitrary. It is up to the data scientist to decide what they will use. However, this choice will affect the resulting classifier.

For example, one can use the Euclidean inner product

$$x \cdot x' = x^T x.$$

Then the discriminant function can be written as

$$\begin{aligned} g(x) &= (\mu_1 - \mu_2) \cdot x - \frac{1}{2}(\mu_1 - \mu_2) \cdot (\mu_1 + \mu_2), \\ &= (\mu_1 - \mu_2)^T x - \frac{1}{2}(\mu_1 - \mu_2)^T (\mu_1 + \mu_2), \end{aligned}$$

and thus the separation hyperplane has normal vector $(\mu_1 - \mu_2)$.

More generally, one can use any symmetric, positive definite matrix M to define an inner product as

$$x \cdot x' = x^T M x.$$

Then the discriminant function can be written as

$$\begin{aligned} g(x) &= (\mu_1 - \mu_2) \cdot x - \frac{1}{2}(\mu_1 - \mu_2) \cdot (\mu_1 + \mu_2), \\ &= (\mu_1 - \mu_2)^T M x - \frac{1}{2}(\mu_1 - \mu_2)^T M (\mu_1 + \mu_2). \end{aligned}$$

Here the separation hyperplane has normal vector $M(\mu_1 - \mu_2)$.

Note that one could perform a change of coordinate to map one inner product to the other. For example, if the singular value decomposition of M is

$$M = U^T \Sigma U,$$

then after mapping x to a new coordinate $\bar{x}$ as

$$x \longrightarrow \bar{x} = \sqrt{\Sigma} U x,$$

the inner product $x \cdot x' = x^T M x$ becomes the Euclidean inner product $\bar{x} \cdot \bar{x}' = \bar{x}^T \bar{x}'$.

Exercise: Consider the nearest-mean classification rule on $\mathbb{R}^2$ equipped with the Euclidean metric. We have seen that the separation line between the two classes in $\mathbb{R}^2$ is a straight line

passing between the two class means μ_1, μ_2. We have also seen that, if the metric is redefined as $d(x, x') = (x - x')^T M(x - x')$, where M is any positive definite symmetric 2-by-2 matrix, then the separation is still a straight line passing between the two class means. Is it possible to define a metric on $\mathbb{R}^2$ such that the above classification method will yield a separation line that is not straight? □

1.10 Selecting a Classifier

We have seen that the choice of structure on the space in which the training samples lie will affect the resulting classifier. We have also seen that this choice is arbitrary. So what to do if we have several possible choices and would like to pick one?

Consider N different classifiers (e.g., N different choices of metrics for a nearest neighbor classifier). Assume that any data point x is a sample of a random variable X. Assume also that we are given labeled samples (x_i, ω_i) drawn independently.

Let Z_i^k be a binary valued random variable representing the success ($Z_i^k = 1$) or failure ($Z_i^k = 0$) of the classification of test input value i using classifier k. Then $Z^k = \frac{1}{n} \sum_{i=1}^{n} Z_i^k$ is a (random) measure of the accuracy of classifier k on the test data. The population accuracy of classifier k is equal to the expectation $E(Z^k)$.

If we pick the method k_0 with the highest test accuracy:

$$k_0 = \arg\max_{k=1,2,\ldots,N} Z^k,$$

then to what extent can be expect this method to perform well on future data?

We have

$$\begin{aligned} Prob\left\{\left|Z^{k_0} - E(Z^{k_0})\right| > \epsilon\right\} &\leq Prob\left\{\cup_{k=1}^{N}\left|Z^k - E(Z^k)\right| > \epsilon\right\} \\ &\leq \sum_{k=1}^{N} Prob\left\{\left|Z^k - E(Z^k)\right| > \epsilon\right\} \\ &\leq \sum_{k=1}^{N} 2e^{-2n\epsilon^2} \\ &= 2Ne^{-2n\epsilon^2}. \end{aligned}$$

So, for a high enough number n of test samples, and a small enough number N of choices, the empirical error of the chosen classifier is likely to estimate the classifier well. The higher the number of classifiers considered, the bigger the likely difference. If the number of considered classifiers is infinite, then the bound is useless. See [1] for bounds that apply to the infinite case.

2 From Data to Random Samples

Data points are not the same as samples from a random variables. However, one can view data points as samples from a random variable provided that a suitable theoretical framework can be assumed to hold. In order to do this, two steps of abstraction are needed:

1. the data points needed to be seen as taking values in some space S;
2. the space S needs to be equipped with a structure so to make it a probability space.

2.1 Types of Random Variables

There are three types of random variables, along with variables of mixed type. Here is a short summary. For more details see Section 1.2 of [7].

1. Quantitative random variables. These are characterized by notions of size (large/small) and a quantified notion of closeness. Random variables taking values in $\mathbb{N}$ or in $\mathbb{R}$ are examples of such.
2. Ordered categorical random variables. These are discrete random variables whose values can be ranked. For example, the three following values are ranked $\{small, medium, large\}$, as $large > medium$ and $large, medium > small$. However, one cannot say that the difference between *small* and *medium* is the same as the difference between *medium* and *large*. Another example is the set $\{child, teen, adult\}$.
3. Unordered categorical random variables. These are discrete random variables without any notion of rank. For example, gender as categorized into three cases as $\{male, female, other\}$ has no ordering. Similarly, the sides of a die, even if they are represented by numbers $\{1, 2, 3, 4, 5, 6\}$, are not ordered in any specific way.

The mapping from data to its representation as a point in some space is a choice. Care must be taken not to introduce structures in S that are not compatible with the data. For example, given categorical data like the genders $\{male, female, other\}$, one should not map these to $\{1, 2, 3\} \in \mathbb{N}$ with a structure such that $1 < 2 < 3$ or $1 + 2 = 3$.

2.2 A Technique to Map Non-numerical Data to $\mathbb{R}^d$

Now we showcase a method to map complex data into points in $\mathbb{R}^d$ through the theoretical framework of a rubric. The technique was developed and used to analyse data in [11].

		1 Basic	2 Intermediate	3 Advanced
A	Rigor			
B	Communication			
C	Estimation			

Table 2.1: Example of Rubric

Suppose you are given data such as set of student exams and homework, or a set of interviews, books, or movies. A traditional method for analysing the data consists in using a theoretical rubric and have an expert go over the data "by hand" to label it according to the rubric. This is similar to grading, where one if given a set of attributes along with values for each attribute.

For example, in [11], we were interested in the "habits of mind" of engineering students, as examplified by certain characteristics such as "rigor," "communication skills" and "estimation skills." For each item, a level of achievement was defined, such as "basic," "intermediate," and "advanced." These skills and levels were arranged in a grid, as in Table 2.1.

The expert uses the rubric to label different parts of the work/homework/movie according to the skill displayed and its level of achievement. This yields a sequence of tags. For example, if using the rubric of Table 2.1, a particular homework could be labeled as

$$(A1), (A3, B1), (C2), (A1, B2, C2), (B3), \ldots, (B3, C1)$$

In other words, the data is summarized as a sequence of vectors of various lengths, where each vector entry consists of a letter (representing a skill) and a number (representing an achievement level).

This sequence of vectors can be viewed as a random process. If the process is modeled as a certain parametric random process, then the specific values of the tags in the sequence, along with their order (and possibly timing as well) can be used to estimate the parameters $\theta = (\theta_1, \ldots, \theta_d)$ of the random process corresponding to one data point (e.g., one movie or one interview). Then the estimated vector of parameters $\hat{\theta}$ provides a representation in $\mathbb{R}^d$ for the given data point.

Note that the representation obtained depends on the rubric chosen: the same data can be interpreted through a different rubric, leading to a different sequence of tabs and thus a different point in $\mathbb{R}^d$.

2.3 Mapping Data to Invariant Coordinates

Suppose the data points are in a space P on which there is an equivalence relation $\sim$ such that

$$\text{class of } p = \text{class of } p', \quad \forall p \sim p'.$$

Recall that an equivalence relation on a space P is a subset of $P \times P$ whose elements, denoted by $p \sim p'$, are such that, for all $p, p', p'' \in P$ we have

- $p \sim p$ for all $p \in P$;
- if $p \sim p'$ then $p' \sim p$, for all $p, p' \in P$;
- if $p \sim p'$ and $p' \sim p''$, then $p \sim p''$, for all $p, p', p'' \in P$.

Definition 1. *A real-value function $I : P \to \mathbb{R}$ is called an* invariant *under $\sim$ if*

$$I(p) = I(p'), \text{ for all } p \sim p'.$$

Under many circumstances, one can map each $p \in P$ to new coordinates

$$x = x(p) = (I_1(p), I_2(p), \ldots, I_d(p)),$$

in such a way that

$$p \sim p' \quad \text{if and only if} \quad x(p) = x(p').$$

In such case, we call $\{I_1, \ldots, I_d\}$ a *separating set of invariants* because their values separate the equivalence classes defined by $\sim$. In other words, the feature vector $(I_1(p), \ldots, I_d(p)) \in \mathbb{R}^d$ is an invariant representation for the point p: it only removes information that is irrelevant to the classification.

There are various methods and algebraic computational tools to obtain a separating set of invariants for a given equivalence relation $\sim$ on a space P. For example, when the equivalence relations is given by the action of a Lie group, one can use the Moving Frame method of Fels and Olver [3]. For equivalence relations given by finite group actions, see for example [2].

Example 1. ***Rotations in the Plane***

Example 2. ***Addition of two units***

Example 3. ***Permuting a set of points***

Example 4. ***Weighted graphs under isomorphism***

Example 5. ***Point Sets under rigid motion and relabeling***

2.4 Extension to an Implicit Feature Space

Recall that the nearest neighbor classification rule can be implemented on any real inner product space S as

$$\hat{\omega} = \omega_{i*}, \text{ where } i* = \arg\min_{i=1,\ldots,n} (x - x_i) \cdot (x - x_i).$$

Similarly, one can assign x to the class of the closest mean on an inner product vector space S as

$$\hat{\omega} = \omega_{i*}, \text{ where } i* = \arg\min_{i=1,2} (x - \mu_i) \cdot (x - \mu_i).$$

In other words, the only structure that is needed to make a decision with these methods is an inner product.

If we map the data in P to a space S equipped with an inner product

$$\phi : P \to S,$$

then we can implement the nearest neighbor classification rule in S as

$$\hat{\omega} = \omega_{i*}, \text{ where } i* = \underset{i=1,\ldots,n}{\arg\min}(\phi(x) - \phi(x_i)) \cdot (\phi(x) - \phi(x_i))$$

and

$$\hat{\omega} = \omega_{i*}, \text{ where } i* = \underset{i=1,2}{\arg\min}(\phi(x) - \phi(\mu_i)) \cdot (\phi(x) - \mu_i),$$

respectively.

Many other methods for classification, and many methods for building a classifier, also only rely on computing inner products.

The kernel trick allows one to compute inner products in the space S without having to know the mapping ϕ or even the space S.

Definition 2. *A function $k : P \times P \to \mathbb{R}$ is called a* kernel function *if there exists a Hilbert space S and a map $\phi : P \to S$ called a* feature map *such that*

$$\phi(p) \cdot \phi(p') = k(p, p') \text{ for all } p, p' \in P.$$

Recall that a Hilbert space is a real inner-product space that is complete with respect to the norm defined by the inner product. Recall also that a *Complete* space is one where every Cauchy sequence converges to a point in the space.

Example 6. *Suppose $P = \mathbb{R}^2$ and $S = \mathbb{R}^3$ equipped with the Euclidean inner product. Write $p = (u, v)$ and consider the map $\phi : \mathbb{R}^2 \to \mathbb{R}^3$ given by*

$$\phi(u, v) = (u^2, \sqrt{2}uv, v^2).$$

Then

$$\begin{aligned} \phi(u,v) \cdot \phi(u',v') &= (u^2, \sqrt{2}uv, v^2) \cdot (u'^2, \sqrt{2}u'v', v'^2) \\ &= u^2u'^2 + 2uvu'v' + v^2v'^2 \\ &= (uu' + vv')^2 \\ &= \left[(u,v)\begin{pmatrix} u \\ v \end{pmatrix}\right]^2. \end{aligned}$$

So here the function $k(p, p') = \left(p^T p\right)^2$ is a kernel. Note that this function is a kernel for other mappings as well, for example

$$\bar{\phi}(u, v) = \frac{1}{\sqrt{2}}(u^2 - v^2, 2uv, u^2 + v^2).$$

Example 7. *Let l_2 be the set of infinite sequences $\{c_i\}_{i=1}^{\infty}$ in $\mathbb{R}$ with $\sum_{i=1}^{\infty} |c_i|^2 < \infty$ equipped with the inner product*

$$\{c_i\}_{i=1}^{\infty} \cdot \{c_i\}_{i=1}^{\infty} = \sum_{i=1}^{\infty} c_i c_i'.$$

Consider a set of functions $f_i : P \to R$ such that

$$\sum_{i=1}^{\infty} f_i(p) \in l_2, \text{ for all } p \in P.$$

Define the map $\phi : P \to l_2$ as

$$\phi(p) = \{f_i(p)\}_{i=1}^{\infty}.$$

We have

$$\phi(p) \cdot \phi(p') = \sum_{i=1}^{\infty} f_i(p) f_i(p').$$

Therefore the kernel in this case is

$$k(p, p') = \sum_{i=1}^{\infty} f_i(p) f_i(p').$$

What kind of functions can be kernels? The following theorem clarifies that kernels are any symmetric, positive definite functions.

Theorem 2.1 (Version of Mercer's Theorem)**.** *A function $k : P \times P\mathbb{R}$ is a kernel if and only if*

1. *$k(p, p') = K(p', p)$ for all $p, p' \in P$*
2. *For any $n \in \mathbb{N}$, the matrix*

$$\big(k(p_i, p_j)\big)_{i,j=1}^{n}$$

is positive semi-definite for any $p_1, \ldots, p_n \in P$.

Proof. Covered in class. □

For more details about implicit coordinates and kernels see Chapter 4 of [8].

Exercise: Given is a kernel $k : P \times P \to \mathbb{R}$ associated to a map $\phi : P \to S$ into some Hilbert space S. Recall that the inner product on S induces a metric on S, namely $d_S(x, x') = \sqrt{(x - x') \cdot (x - x')}$. Does this metric on S induce a metric on P as well? □

2.5 From Metric Space to Probability Space

A probability space (S, Σ, μ) is made of

- S, the *sample space*, that is to say the set of possible outcomes of a random experiment.
- Σ, a σ-algebra called the *event space*, that is to sat the set of possible events, where each event is a set of outcomes in S. Recall that a σ-algebra satisfies (1) $S \in \Sigma$, (2) if $A \in \Sigma$, then $A^c = S \setminus A \in \Sigma$, (3) if $A_i \in \Sigma$, then $\bigcap A_i \in \Sigma$.
- μ, the *probability function*, that is to say a probability measure $\mu :\longrightarrow [0, 1]$. Recall that a probability measure satisfies (1) $\mu\left(\bigcup_i A_i\right) = \sum_i \mu\left(A_i\right)$ if the $A_i \in \Sigma$ have an empty intersection $A_i \bigcap A_j =$ when $i \neq j$, and (2) $\mu(S) = 1$.

We have previously discussed how to map data to points in a metric space. There is a very natural way to make a metric space into a probability space.

One can construct a σ-algebra called a “Borel algebra" on a metric space S, d as follows.

1. Use the metric $d : S \times A \longrightarrow \mathbb{R}$ to define open sets as

$$\{x \in S | d(x, x_0) < r\}, \text{ for some } r \in \mathbb{R}_{\geq 0} \text{ and some } x_0 \in S.$$

2. add more open sets by taking complements and intersections until you get a σ-algebra Σ.

In other words, if one equips a space S with a metric (or with an inner product, which naturally induces a metric), then viewing S as the sample space, one is indirectly building a σ-algebra on S.

So the only thing that remains in order to have a probability space is the probability function μ. Such a function can be assumed to exist if the data points can be assumed to be obtained following some “pattern." In other words, we expect some level of repeatability in the process that yields the data. Assuming this, then the probability measure can be estimated by counting data points (vieweded as samples) inside open sets of a fixed size, as per the metric chosen. The assumption of repeatability implies that the probability measure obtained in this fashion should be consistent over different data sets (i.e. random draws of samples).

3 Hypothesis Testing

3.1 The MAP criterion

References section 8.1 of [5] (or Chapter 3.1 of Gallager's corresponding lecture notes).

We begin with a motivating example to illustrate the importance of considering the probability function underlying the data. Suppose an advertisement company would like to be able to determine if a forum user is male or not. They are hiring a data scientist to design a classifier.

Consider the two following classifiers:

Classifer 1: Decide "male" all the time.

Classifier 2: Decide "not male" all the time.

Which classifier is better?

Assuming that the probability that a forum user is male is 0.5, then the probability that the forum use if not male is also 0.5. Therefore, both Classifier 1 and Classifier 2 have a 0.5 probability of error. So, under this assumption, both classifiers are equally bad.

Now, if after more investigation, it is determined that the forum is for students in ECE at Purdue, where about 90% of the students are male, then Classifier 1 would only have a 0.1 probability of error while Classifier 2 would have a 0.9 probability of error. In this scenario, Classifier 1 is far better.

This example illustrates several things.

1. whether a classifier is good or not depends on the context in which it is applied,
2. The probability distribution of the data does matter when designing a classifier,
3. the question of the accuracy of a classifier does not make any sense, unless a specific context (and thereby probability law for the data) is specified
4. The probability of error of a classifier must be interpreted in the context of the problem at hand. For example, a low probability of error does not mean that the classifier is good. In particular, if the ratio of the class in the data is very unbalanced, that is to say if one class has a very high probability of being drawn from the population, then it is very easy to obtain a classifier with high accuracy.

3.2 Bayes Decision Rule

3.3 Bayes Decision Rule for Normally Distributed Features

3.4 Bounds on Bayes Error

3.5 Minimum Cost Criterion

3.6 Neyman-Pearson Rule

3.7 Sequential Hypothesis Testing

3.8 Single Hypothesis Testing

Bibliography

[1] Y. S. Abu-Mostafa, M. Magdon-Ismail, and H.-T. Lin. *Learning from data*, volume 4. AMLBook New York, NY, USA:, 2012.

[2] H. Derksen and G. Kemper. *Computational invariant theory*. Springer, 2015.

[3] M. Fels and P. J. Olver. Moving coframes: I. a practical algorithm. *Acta Applicandae Mathematica*, 51(2):161–213, 1998.

[4] K. Fukunaga. *Introduction to statistical pattern recognition*. Elsevier, 2013.

[5] R. G. Gallager. *Stochastic processes: theory for applications*. Cambridge University Press, 2013.

[6] P. E. Hart, D. G. Stork, and R. O. Duda. *Pattern classification*. Wiley Hoboken, 2000.

[7] T. Hastie, J. Friedman, and R. Tibshirani. *The elements of statistical learning*. Springer series in statistics New York, 2008.

[8] I. Steinwart and A. Christmann. *Support Vector Machines*. Springer Publishing Company, Incorporated, 1st edition, 2008.

[9] R. Vershynin. *High-dimensional probability: An introduction with applications in data science*, volume 47. Cambridge university press, 2018.

[10] H. Wang, M. Boutin, J. Trask, and J. Allebach. Three efficient, low-complexity algorithms for automatic color trapping. *arXiv preprint arXiv:1808.07096*, 2018.

[11] T. Yellamraju, A. J. Magana, and M. Boutin. Investigating students' habits of mind in a course on digital signal processing. *IEEE Transactions on Education*, 62(4):312–324, 2019.