Advanced Machine Learning **Fall 2020**

Lecture 2: Representations, measurements, data types

Note: *LaTeX template courtesy of UC Berkeley EECS dept.*

Disclaimer: *These notes are adapted from ETH's Advanced Machine Learning Course, All Of Statistics, Larry Wasserman, Springer and Statistical Machine Learning Notes 3, Justin Domke, UMASS.*

2.1 Data Representation

One of the fundamental problems in Machine Learning is how a machine should represent an object/concept. Machines only understand numbers, thus we need to find a way to condensate all the information about an object in a set of numbers (i.e a vector). Such representation can later be used to perform different tasks; whether we want to classify the object, generate new ones or do anything else.
One must be extremely careful in choosing the data representation: a wrong data representation can induce inappropriate similarity measures.

2.1.1 Definition of Structures

A statistical definition of good and poor structures is mandatory for rational pattern recognition.
Multi-scale optimization yields efficient algorithms to detect good structures in data, but are the structures indeed in the data or are they explained by fluctuations?
Without **validation**, any pattern recognition strategy is doomed to fail.

2.1.2 Definition of Data

Measurements: associations of numbers with physical quantities and natural phenomena by comparing an unknown quantity with a known quantity of the same kind.
Our goal is to represent objects of interest and characterize them according to their typical patterns for detection, classification and abstraction. Measurements represent objects in a data space, e.g digits as objects and pixel intensities as measurements.

2.2 Feature Space

Measurement space $\mathcal{X}$: the mathematical space in which the data are represented, e.g numerical ($\mathcal{X} \subset \mathbb{R}^d$), Boolean ($\mathcal{X} = \mathbb{B}$) or categorical ($\mathcal{X} = \{1, ..., K\}$) features.
Features are derived quantities or indirect observations which often significantly compress the information content of measurements.
The selection of a specific feature space predetermines the metric to compare data, this choice is the first significant design decision in a machine learning system.

2.3 Learning Problems

Learning requires to infer a functional or statistical relationship between variables when we only observe noisy samples. Approximation and interpolation in function estimation are such procedures.
The problem without additional assumptions is mathematically ill-defined since many different functions might be compatible with our observations. We, therefore, require that our inference has to "work" on future data. Mathematically, the expected quality of inference should be high and not necessarily the empirically observed quality.

Applications in which the training data comprises examples of the input vectors along with their corresponding target vectors are known as **supervised learning** problems.
Cases such as the digit recognition example, in which the aim is to assign each input vector to one of a finite numbers of discrete categories, are called **classification** problems.
If the desired output consists of one or more continuous variables, then the task is called **regression**.

In other pattern recognition problems, the training data consists of a set of input vectors $\mathcal{X}$ without any corresponding target values. The goal in such **unsupervised learning** problems may be to discover groups of similar examples within the data, where it is called **clustering**, or to determine the distribution of data within the input space, known as **density estimation**, or to project the data from a high dimensional space down to two or three dimensions for the purpose of visualization (**dimension reduction**).

Another task could be **data compression**, where a system predicts the posterior probabilities of a sequence given its entire history. This can be used for optimal data compression.

Finally, the technique called **reinforcement learning** is concerned with the problem of finding suitable actions to take in a given situation in order to maximise a reward. Here, the learning algorithm is not given examples of optimal outputs, in contrast to supervised learning, but must instead discover them by a process of trial and error.

2.3.1 Supervised Learning

A teacher (oracle) provides the correct answer during training.
Data are pairs of features and response variables $\{(x_1, y_1), ..., (x_n, y_n) : x_i \in \mathcal{X} \subset \mathbb{R}^d, y_i \in \mathbb{K}\}$ with:
$\mathbb{K} = \{1, ..., K\}$ for classification where $\mathbb{K}$ is an index set for the classes;
$\mathbb{K} = [0, 1]^K$ is the space of assignments for probabilistic classification;
$\mathbb{K} \subset \mathbb{R}$ for regression.

Problem: The data are noise contaminated, e.g the response variable $y = f(x, w) + \eta$ depends on the function with parameters w and Gaussian white noise η.
Question: How can we infer a functional relationship $f(x, w)$ from data which are described by the statistical relationship $P(X = x, Y = y)$?
Statistical learning theory provides the **answer**: define a function class $\mathcal{C} = \{f(x, w) : w \in W, x \in \mathbb{R}^d\}$ where w indexes the functions (hypothesis) in class $\mathcal{C}$. It turns out that the "complexity" of the function class $\mathcal{C}$ is the essential concept to describe the difficulty of learning. If we have too few data and we work with a too complex function class then learning algorithms have a strong tendency to overfit, i.e to confuse/interpret noise as signal.

2.4 The Dilemma of Learning

What should we do about overfitting?

- Minimize **expected** classification error.
- Maximize generalisation.

What can we do about overfitting?

- Minimize **empirical** classification error.
- Maximize estimated empirical generalisation performance by cross-validation.

We search a function $f(x) \in \mathcal{C}$ out of the hypothesis class/solution space C such that:

$$\begin{gathered} f : X \to Y \\ x \to y = f(x) \end{gathered}$$

Often we index the function $f(x) = f_\theta(x)$ by a parameter θ.

2.4.1 Generalization

Given an input space $\mathcal{X}$ with data $x \in \mathcal{X}$ find an interpretation $c \in \mathcal{C}$.

$$x \text{ is a random variable and } c \text{ is calculated from } x \text{ by a procedure } \mathcal{A} \implies c \text{ is a random variable}$$

Thus, we can search for a posterior distribution $P(c|x)$.
One special choice is the Gibbs Distribution: $P(c|x) = \dfrac{\exp\left(-\beta \mathcal{R}(c,x)\right)}{\sum_{c \in \mathcal{C}} \exp\left(-\beta \mathcal{R}(c,x)\right)}$

If you know what the $\mathcal{R}$ is then this probability distribution tells you how for a certain parameter β you should draw your solution. The problem with this special choice is that now we need to model this $\mathcal{R}$ and furthermore, it has to be **validated**.

Assume that two datasets $\mathcal{X}', \mathcal{X}'' \sim p(x', x'') = p(x')p(x'')$ are given. We can consider the expected risk to assess how well our choice of $\mathcal{R}$ generalizes:

$$\mathbb{E}_{\mathcal{X}'\mathcal{X}''}\Big[\sum_{c \in \mathcal{C}} p(c|x')\mathcal{R}(c, x'')\Big] = \sum_{\mathcal{X}'}\sum_{\mathcal{X}''} p(x'') \sum_{c \in \mathcal{C}} p(c, x')\mathcal{R}(c, x'') = \sum_{c \in \mathcal{C}} p(c)\mathbb{E}_{\mathcal{X}''}\left[\mathcal{R}(c, x'')\right]$$

One might argue that minimizing the expected risk is not the optimal inference principle for validating $\mathcal{R}$. The problem is that $\mathcal{R}$ is particularly large when you're far away from the minimum and that's exactly when you expect $p(c)$ to be very small because solutions which give you an high cost on the test data presumably should be rare also on the train data. Thus, the situation is unstable, you multiply a very large cost value with a very small probability $(0 \cdot \infty)$. From an information-theory point of view, score-maximization is a better principle; however, risk-minimization leads to convex problems that are computationally more feasible.

2.4.2 Quality of the estimate

The **loss function** L measures the deviation between dependent variables y and prediction $f(x)$:

$$L(f(x), y) = \begin{cases} (y - f(x))^2 & \text{quadratic loss (regression)} \\ \mathbb{1}_{y \neq f(x)} & \text{0-1 loss (classification)} \\ \exp\left(-\beta y f(x)\right) & \text{exponential loss (classification)} \end{cases}$$

Conditional expected risk:
Given the random variable X the conditional expected risk is defined as:

$$R(f, x) = \int_y L(f(x), y) P(Y|X) dY$$

Expected true risk:

$$R_{true}(f, x) = \mathop{\mathbb{E}}_{p}[L(f(x), y)] = \int_x \int_y L(f(x), y) P(x, y) dx dy$$

Where p is the true distribution over the inputs x and y. The risk measures how much error we have, on average, using f as our prediction algorithm.

This can be clarified by considering an example. Suppose we want to fit a function for predicting if it will rain or not. The input x will be the sky: CLEAR, CLOUDY or MIXED.
The output y will be either RAIN or NOPE. The loss function is now a function $L : \{\text{RAIN, NOPE}\}^2 \to \mathbb{R}$. Which loss function is appropriate? It depends on the priorities of the user. For example, we have L:

$Y_1 \backslash Y_2$	RAIN	NOPE
RAIN	0	1
NOPE	25	0

Meaning that predicting NOPE instead of RAIN it is 25 times worse than predicting RAIN when it does not rain.
Now suppose the distribution p is given as follows:

$X \backslash Y$	RAIN	NOPE
CLEAR	0	1/4
CLOUDY	1/4	0
MIXED	1/6	1/3

Let's consider two possible predictors:

$$f_1(x) = \begin{cases} \text{CLEAR NOPE} \\ \text{CLOUDY RAIN} \\ \text{MIXED NOPE} \end{cases}$$

$$f_2(x) = \begin{cases} \text{CLEAR NOPE} \\ \text{CLOUDY RAIN} \\ \text{MIXED RAIN} \end{cases}$$

If we use L, it is easy to calculate $R(f_1) = 1/6 \cdot 25$ and $R(f_2) = 1/3 \cdot 1$ and so f_2 has the lowest risk.
So it sounds like the thing to do is picking f to minimize the risk. Trouble is, that it is impossible. To calculate the risk, we would need to know the true distribution p. If we knew it, we would not be doing machine learning.
Since the data comes from p, we should be able to get a reasonable approximation:

$$\mathop{\mathbb{E}}_{p}[L(f(x), y)] \approx \hat{\mathbb{E}}[L(f(x), y)]$$

The right hand side of the equation is called the **empirical risk**.

$$R(f) = \hat{\mathbb{E}}[L(f(x), y)]$$

Picking the function f^* that minimizes it is known as **Empirical Risk Minimization**.

$$f^* = \arg\min_{f \in C} R(f)$$

Our hope is that empirical risk minimization performs similarly to true risk minimization, i.e that:

$$\arg\min_{f \in C} R(f) \approx \arg\min_{f \in C} R_{true}(f) \tag{2.1}$$

How true is Eq. 2.1 in practice depends on four factors:

- How much data we have. For any given function f as we get more and more data we can expect that $R(f) \to R_{true}(f)$.
- The true distribution p. Depending on how "complex" the true distribution is, more or less data may be necessary to get a good approximation of it.
- The loss function L. If the loss function is very "weird" - giving extremely high loss in certain unlikely situations - this can lead to trouble.
- The class of functions of C. Roughly speaking, if the size of C is "large", and the functions in C are "complex", this worsens the approximation, all else being equal.

So why not use a small set of "simple" functions? It is true, this will lead to empirical risk minimization approximating true risk minimization. However, it also worsens the value of the minimum of the true risk:

$$\arg\min_{f \in C} R_{true}(f)$$

This phenomenon is called **Bias-Variance Trade-off**.
When used in practice, it is usually necessary to perform some sort of model selection or regularization to make empirical risk minimization generalize well to new data.

In fact, what usually happens is that the samples are split into training data and test data. Additional validation data is used to guide the estimator selection.
Test data cannot be used before the final estimator has been selected.

$$\mathcal{Z}^{train} = \{(x_1, y_1), ..., (x_n, y_n)\}$$

$$\mathcal{Z}^{test} = \{(x_{n+1}, y_{n+1}), ..., (x_{n+m}, y_{n+m})\}$$

We have the training error $\hat{R}(f, \mathcal{Z}^{train}) = \frac{1}{n} \sum\limits_{i=1}^{n} L(f^*(x_i), y_i)$ and its minimizer $f^* = \arg\min_{f \in C} \hat{R}(f)$.

The test error amounts to $\hat{R}(f, \mathcal{Z}^{test}) = \frac{1}{m} \sum\limits_{i=n+1}^{m+n} L(f^*(x_i), y_i)$.
When we use test data for validation, then estimator adaption introduces statistical dependencies between outcome of the learning process (estimator) and test data. This design flow yields a too optimistic estimate of the test error.
Furthermore, it is important to **distinguish between test error and expected risk**:

$$\hat{\mathcal{R}}(f^*, \mathcal{Z}^{test}) \neq \mathop{\mathbb{E}}_{x}[\mathcal{R}(f^*, X)]$$

Notice that $\mathop{\mathbb{E}}_{x}[\mathcal{R}(f^*, X)]$ is a random variable since f^* is random. Moreover, this inequality holds because there is more randomness on the LHS than on the RHS (f^* and $\mathcal{Z}^{test}$ vs f^*).

We can ask what is the probability $P(|\hat{R}(f^*, \mathcal{Z}^{test}) - \mathop{\mathbb{E}}_{x}[R(f^*, X)]| > \epsilon)$? If we succeed in bounding this probability close to 0, then we have an assurance against bad surprises.
The test error empirically estimates the expected risk. To assess the quality of the estimate we should report mean and variance or another measure of deviation.

2.5 Taxonomy of Data

Pattern analysis requires to find structures in sets of object representations.
We are given a object space O and a measurement X that maps an object set into a domain $\mathbb{K}$. Measurements provide information from reality to feed our modeling in more quantitative terms.

$$X : O^{(1)} \times ... \times O^{(R)} \to \mathbb{K}$$

$$(o_1, ..., o_R) \to X_{o_1,...,o_R}$$

Examples:

- Feature Vectors: $X : O \to \mathbb{R}^d$, $o \to X_o$
- Classification Data: $X : O \to \mathbb{R}^d \times \{1, ..., k\}$, $o \to (X_o, Y_o)$
- Regression Data: $X : O \to \mathbb{R}^d \times \mathbb{R}$, $o \to (X_o, Y_o)$
- Proximity Data: $X : O \times O \to \mathbb{R}$, $(o_1, o_2) \to X_{o_1,o_2}$

a. **Monadic Data**: $X : O \to \mathbb{R}^d$, $o \to X_o$
   Monadic data characterize configurations or objects without reference to other configurations, e.g temperature and pressure are measured for each location in absolute terms.

b. **Diadic Data**: $X : O^{(1)} \times O^{(2)} \to \mathbb{R}$, $(o_1, o_2) \to X_{o_1,o_2}$

c. **Polyadic Data:** $X : O^{(1)} \times O^{(2)} \times O^{(3)} \to \mathbb{R}$, $(o_1, o_2, o_3) \to X_{o_1,o_2,o_3}$

Choosing the correct representation of objects is important, because the choice reflects on the algorithm itself. Otherwise the model has to learn a wrong representation and try to adjust itself to wrong constraints.

2.5.1 Scales

a. **Nominal or Categorical Scale**: qualitative but without quantitative measurements, e.g *binary scale* $X = \{0, 1\}$.

b. **Ordinal Scale**: measurement values are meaningful only with respect to other measurements, i.e the rank order carries the information, not the numerical differences.

c. **Quantitative Scale**:

   (i) **Interval Scale**: the relation of numerical differences carries the information. Invariance w.r.t translation and scaling (e.g Fahrenheit scale).

   (ii) **Ratio Scale**: zero value of the scale carries the information but not the measurement unit (e.g Kelvin scale).

   (iii) **Absolute Scale**: absolute values are meaningful (e.g grades).

Scale Type	Transformation invariances
Nominal	$T = \{f : \mathbb{R} \to \mathbb{R} \,\vert\, f \text{ bijective}\}$
Ordinal	$T = \{f : \mathbb{R} \to \mathbb{R} \,\vert\, f(x_1) < f(x_2), \forall x_1 < x_2\}$
Interval	$T = \{f : \mathbb{R} \to \mathbb{R} \,\vert\, f(x) = ax + c, a \in \mathbb{R}^+, c \in \mathbb{R}\}$
Ratio	$T = \{f : \mathbb{R} \to \mathbb{R} \,\vert\, f(x) = ax, a \in \mathbb{R}^+\}$
Absolute	$T = \{f : \mathbb{R} \to \mathbb{R} \,\vert\, f \text{ is identity map}\}$

The cost function has to obey the invariants and if they are known in advance we can search for a better learning procedure.

2.6 Mathematical Spaces

Definition 2.1 ***Topological Space***
Let X be a non-empty set and $\Im$ a collection of subsets of X such that:

I. $X \in \Im$;

II. $\emptyset \in \Im$;

III. *If* $O_1, ..., O_n \in \Im$, *then* $O_1 \cap ... \cap O_n \in \Im$;

IV. *If for each* $\alpha \in I$, $O_\alpha \in \Im$, *then* $\cup_{\alpha \in I} \in \Im$;

The pair of objects $(X, \Im)$ *is called a topological space.*

Topological spaces only describe the closeness/neighborhood of objects but they do not model any quantitative differences (distances) between the "degrees of closeness". The concept of a topological space is one of the most fruitful concepts of modern mathematics. It is the proper setting for discussion based on considerations of continuity.
Topological spaces allow us to introduce the concept of a neighborhood and to define *neighborhood spaces* in a natural way.

Definition 2.2 ***Metric Space*** *A pair of objects* (X, d) *consisting of a non-empty set* X *and a function* $d : X \times X \to \mathbb{R}$ *is called a metric space provided that:*

I. *Positivity:* $d(x, y) \geq 0$, $\forall x, y \in X$;

II. *Uniqueness:* $d(x, y) = 0 \Leftrightarrow x = y, \quad \forall x, y \in X$;

III. *Simmetry:* $d(x, y) = d(y, x)$, $\forall x, y \in X$;

IV. Δ *inequality:* $d(x, z) \leq d(x, y) + d(y, z)$, $\forall x, y, z \in X$;

The function d is called a distance function or metric on X and the set X is called the underlying set.

Example: $(\mathbb{R}, d)$ is a metric space, where d is the function defined by $d(a, b) = |a - b| \forall a, b \in \mathbb{R}$.

Definition 2.3 ***Scalar Product*** *Let X be a non-empty set and $\mathcal{V} = (X, +, \cdot)$ a vector space. A function $\phi : X \times X \to \mathbb{R}$ which assigns two vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{V}$ a real number is called* ***scalar product*** *on $\mathcal{V}$ if the following properties hold:*

I. *Distributivity:* $\phi(\boldsymbol{x_1} + \boldsymbol{x_2}, \boldsymbol{y}) = \phi(\boldsymbol{x_1}, \boldsymbol{y}) + \phi(\boldsymbol{x_2}, \boldsymbol{y})$;

II. *Commutativity:* $\phi(\boldsymbol{x}, \boldsymbol{y}) = \phi(\boldsymbol{y}, \boldsymbol{x})$;

III. *Homogeneity:* $\phi(\alpha \boldsymbol{x}, \boldsymbol{y}) = \alpha \phi(\boldsymbol{x}, \boldsymbol{y})$, $\forall \alpha \in \mathbb{R}$;

IV. Positive definiteness: $\phi(\boldsymbol{x}, \boldsymbol{x}) > 0$, $\forall \boldsymbol{x} \neq 0$;

A vector space with such a scalar product is called **Euclidean vector space**. The scalar product defines the norm $\|\boldsymbol{x}\| \triangleq \sqrt{\phi(\boldsymbol{x}, \boldsymbol{x})}$.

Every element of a metric space corresponds to an element in a metrizable space.
From a machine learning point of view we have to answer the question how precisely we can actually gather metric information rather than topological information in an application scenario. Such an analysis then suggests the appropriate space to model the structures in the data.

2.6.1 Probability Spaces

Definition 2.4 ***Elementary event***
$\omega_1, ..., \omega_n$ *are samples points*

Definition 2.5 ***Sample Space***
$\Omega = \{\omega_1, ..., \omega_n\}$
The sample space Ω *is the set of possible outcomes of an experiment. Points* ω *in* Ω *are called sample outcomes, realizations, or elements. Subsets of* Ω *are called Events.*

Example: If we toss a coin twice then $\Omega = \{\text{HH,HT,TH,TT}\}$. The event that the first toss is heads is $A = \{\text{HH,HT}\}$.

Definition 2.6 ***Family of Sets***
An event A of an experiment is a set of elementary events with the following conditions:

- $A \subset \Omega$
- $\omega \in A \vee \omega \notin A$

Definition 2.7 ***Algebra of events***
Let A, B *be events.* $\mathcal{A}$ *is an algebra of events, i.e, a set of subsets* $A \subset \Omega$*, for which holds:*

- $\Omega \in \mathcal{A}$
- *if* $A \in \mathcal{A} \wedge B \in \mathcal{A}$*, then* $A \cup B \in \mathcal{A} \wedge A \setminus B \in \mathcal{A}$

Definition 2.8 ***Probability of events***
A function $\mathbb{P}$ *that assigns a real number* $\mathbb{P}(A)$ *to each event A is a* ***probability distribution*** *or a* ***probability measure*** *if it satisfies the following three axioms:*

- ***Axiom 1***: $\mathbb{P}(A) \geq 0$ *for every A*
- ***Axiom 2***: $\mathbb{P}(\Omega) = 1$
- ***Axiom 3***: *If* $A_1, A_2, ...$ *are disjoint then:*

$$\mathbb{P}(\bigcup_{i=1}^{\infty} A_i) = \sum_{i=1}^{\infty} \mathbb{P}(A_i)$$

Definition 2.9 ***Probability model***
A probability model or a probability space is a triple

$$(\Omega, \mathcal{A}, \mathbb{P})$$

with the sample set $\Omega = \{\omega_1, ..., \omega_n\}$ *the event algebra* $\mathcal{A}$ *and the probabilities* $\mathcal{P} = \{\mathbb{P}(A) | A \in \mathcal{A}\}$.