Machine Learning & Deep Learning

Tutorial

MSCV/ESIREM

Antoine Lavault
antoine.lavault@u-bourgogne.fr

Basics 1

Any question or exercise marked with a "*" is typically more technical or goes further into developing the tools and notions seen during class.

Problem 1

Basic ML concepts. This exercise aims to review some basic concepts of machine learning.

1. Describe a general ML workflow with the relevant data splits.
2. What should a validation set be used for? What is the difference between a validation and a test dataset?
3. Why should you never tune your model's hyperparameters on your test set?
4. For a given algorithm, how could you tell if the algorithm is optimization-based (or model-based) or learning-based (or data-driven)?
5. Describe briefly what overfitting and underfitting are. How could you detect whether either situation is happening?

Problem 2

Probability. A short reminder about probability and distributions.

Definition 1 (Dataset)**.** A dataset $\mathcal{D}$ of size n is composed of n individual examples $\{(x_1, y_1), \ldots, (x_n, y_n)\}$, where $x_i \in \mathbb{R}^d$ represents the i-th input feature, and y_i represents the i-th label. Datasets without the label y_i are called unlabeled datasets, and datasets with these labels are called labeled datasets.

Each example represents a data type: scalar values, images, text, audio waveform, etc. Suppose we would like to model the probability distribution of our data (e.g., snare drum samples). This will model our data's *joint distribution*, given by $P(x_1, \cdots, x_n)$.

Definition 2 (Joint Distribution)**.** The joint distribution of two random events, A and B, is the probability of both events co-occurring and is written as $P(A, B)$.

Definition 3 (Conditional Probability)**.** The conditional probability of two events, A and B, is the probability of one occurring, given that the other has occurred. The probability that A has occurred, given that B has occurred, is denoted $P(A|B) = P(A, B)/P(B)$ if $P(B) \neq 0$.

Definition 4 (Independence)**.** If A and B are independent events, and their probabilities are $P(A)$ and $P(B)$ respectively, then their joint probability is $P(A, B) = P(A) \times P(B)$.

In other words, A and B are independent if, and only if, $P(A) = P(A|B)$.

This result can be extended to random variables. With X and Y two random variables, X and Y are independant when the events $\{X \in S\}$ and $\{Y \in T\}$ for all sets of values S and T.

We often assume that datasets consist of independent, identically distributed (i.i.d.) samples.

1. Show that $P(A, B) = P(A|B)P(B) = P(B|A)P(A)$ and $P(A|B) = \frac{P(B|A)P(A)}{P(B)}$ if $P(B) \neq 0$.
   The last result is called *Bayes' Theorem*.

2. Application: Let $P(H)$ be the probability you have a headache, and $P(F)$ be the probability you have the flu with the following distribution:

Headache	Flu
N	N
Y	N
N	N
Y	Y
Y	Y
N	Y

Table 1: Flu vs Headache

Given the previous data, calculate $P(F)$, $P(H)$, $P(H|F)$. Then, calculate $P(F|H)$

Problem 3

Consistent Estimators and Loss Functions. There is a lot of mysticism surrounding neural networks, but at their core, they are just (universal) function approximators. For example, in the case of classification, we try to learn $P(y|x)$, which is the probability our true label is some class y (e.g., being a snare drum sound) given input features x (some sound samples). In the case of regression, it's a similar continuous response variable. In the case of generative models, we are trying to learn to approximate a whole distribution so that $P(x_{gen}) = P(x_{real})$. In all cases, we try to find an estimator $f_\theta(x)$ of a true distribution y. To find $f_\theta(x)$, we must adjust the weights and biases in the network, often called the parameters θ of the network, to minimize the "distance" between the estimated distribution $f_\theta(x)$ and the true distribution y.

But how do we define these distance metrics? It turns out we can use the Risk function to evaluate how well an estimator performs.

Definition 5 (Loss Function)**.** A loss function $L(x, y, \theta)$ measures the "badness" of an estimator and is often measured in terms of some distance between the estimate and the true value.

Loss functions

1. The sigmoid function is a popular activation function in neural networks. Let us denote the sigmoid function as:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

   Calculate the derivative of the sigmoid function with respect to x in terms of $\sigma(x)$. What type of equation does this represent?

2. Let us define the softmax function defined by:

$$p_i = \frac{e^{f_i(x)}}{\sum_{j=1}^{n} e^{f_j(x)}}$$

   Softmax can be considered a multi-class extension to the sigmoid function, and its derivative is often used for optimization in multi-class contexts. It can convert real numbers to a "probability distribution."

   - Show that $\sum_i p_i = 1$.
   - Calculate the partial derivative of the softmax function with respect to $f_k(x)$ for each k.

Definition 6 (Risk Function)**.** The risk function is the expected loss (known as the risk), measured as a function of the parameter θ, so that:

$$R(\theta; f_\theta(\cdot)) = \mathbb{E}[L(x, y, \theta)]$$

Risk functions, Bias/Variance trade-off (*) For example, if $L(x, y, \theta) = (y - f_\theta(x))^2$ (squared error loss), then $R(\theta; f_\theta(\cdot)) = \mathbb{E}[(y - f_\theta(x))^2]$, also known as the mean squared error (or MSE). This is the expected squared deviation of the estimator from the true distribution (over the true distribution). That said, we cannot directly optimize this objective (i.e., minimize the risk) since we do not have access to the true distribution, so we cannot sample $x \sim p(x)$, and we only have the dataset $\mathcal{D}$. Instead, we use empirical risk minimization, replacing the true distribution with the empirical distribution from $\mathcal{D}$.

Definition 7 (Empirical Risk)**.** The empirical risk is evaluated on samples from the true distribution and approximates the true risk. It is given by:

$$\frac{1}{n}\sum_{i=1}^{n} L(x_i, y_i, \theta)$$

Supervised learning is (usually) empirical risk minimization, but is it the same as true risk minimization? In supervised learning, we aim to learn a function that does well regarding the true risk. However, we generally do not know the true distribution and only have access to a dataset of samples from the distribution. This means we instead learn by minimizing the empirical risk (often with an additional regularization term). Even though we cannot directly optimize the risk, we can still attempt to understand better the sources of error in our estimation. In particular, when our loss is the squared error, we can derive the bias-variance decomposition of the MSE, especially at test inputs. Intuitively, the bias and variance can be summarized in fig. 1.

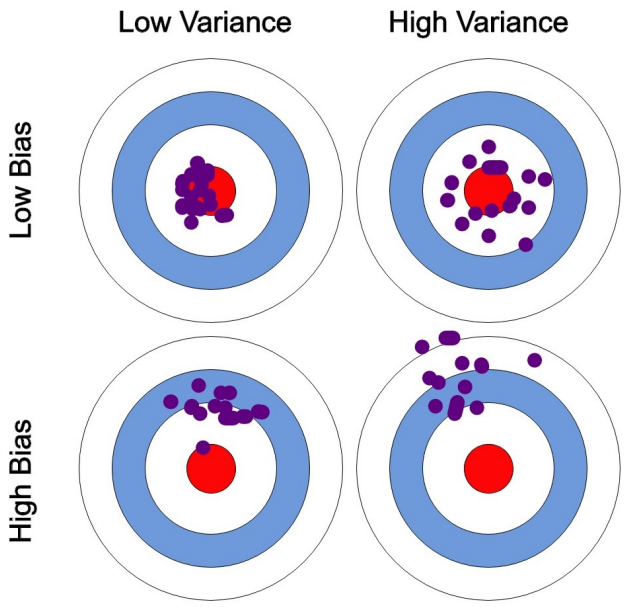

Figure 1: A visual explanation of the bias and variance. From `https://nvsyashwanth.github.io/machinelearningmaster/bias-variance/`

The interpretation of high bias means a deviation from the target, and high variance means to spread out measurements around a centroid. The best estimates are those with low variance and low bias since they mostly hit the target.

Bias-Variance tradeoff for the MSE (*) Suppose we have a randomly sampled training set $\mathcal{D}$ drawn independently from our test data, and we select an estimator denoted $\theta = \hat{\theta}(\mathcal{D})$ (for example, via empirical risk minimization). We can decompose our expected mean squared error for a particular test input x into a bias, a variance, and an irreducible error term:

$$E_{Y\sim p(y|x),\mathcal{D}}[(Y - f_{\hat{\theta}(\mathcal{D})}(x))^2] = Var(f_{\hat{\theta}(\mathcal{D})}(x)) + Bias(f_{\hat{\theta}(\mathcal{D})}(x))^2 + \sigma^2$$

.

We will use $\hat{f} = f_{\hat{\theta}(\mathcal{D})}(x)$, and $f = f(x)$ for a slicker notation.

1. Suppose that we have a training set consisting of a set of points $x_1, \cdots, x_n$ and real values y_i associated with each point x_i. We assume that there is a function $f(x)$ such as $y = f(x) + \varepsilon$, where the noise, ε, has zero mean and variance σ^2. We will derive the result above in this specific case.

   You may find it helpful to recall the formulaic definitions of Variance and Bias, reproduced for you below:

   $$Var(f_{\hat{\theta}(\mathcal{D})})(x) = E_{\mathcal{D}}[(f_{\hat{\theta}(\mathcal{D})}(x) - E[f_{\hat{\theta}(\mathcal{D})}(x)])^2]$$
   $$Bias(f_{\hat{\theta}(\mathcal{D})}(x)) = E_{Y\sim p(Y|x),\mathcal{D}}[f_{\hat{\theta}(\mathcal{D})}(x) - Y]$$

   (a) Show that $E(\hat{f}^2) = Var(\hat{f}) + E(\hat{f})^2$

   (b) Show that $E[y^2] = f^2 + \sigma^2$

   (c) Show that $E(y\hat{f}) = fE(\hat{f})$

   (d) By expanding the expression of the MSE in the present case, conclude.

2. Apply this result to the ordinary least-square linear regression under its canonical hypotheses. We let the label vector $Y = X\theta + \varepsilon$ where θ is the true linear predictor, and each noise variable ε_i is independent and identically distributed (i.i.d.) with mean 0 and variance 1.

   (a) The OLS estimator is $\hat{\theta} = (X^TX)^{-1}X^TY$. Calculate the bias and the bias for a test input x.

   (b) Calculate the covariance of $\hat{\theta}$, i.e., $E[(\hat{\theta} - \theta)(\hat{\theta} - \theta)^T]$.

   (c) Calculate the variance for a test input x.

   (d) Assume that X^TX is diagonal (we could have applied an orthogonal transformation to make this the case) with sorted entries. Conlude

3. Same questions with a L_2 regularized linear regression, given by $\hat{\theta} = (X^TX + \lambda I)^{-1}X^TY$.

4. How does this relate to underfitting and overfitting?

Problem 4

Vectors, Matrices, and Optimization This section aims to review some basic linear algebra notions aided by applications to optimization.

Gradient and Jacobian We first define the gradient of a scalar function with respect to a vector input.

Definition 8. Suppose we have a function $f : \mathbb{R}^d \mapsto \mathbb{R}$, which maps a d-dimensional vector to a scalar. Then, we define the gradient of f at a particular input x to be a column vector (the same shape as the input) consisting of partial derivatives at x:

$$\nabla f(x) = \begin{bmatrix} \frac{\partial f}{\partial x_1}(x) \\ \vdots \\ \frac{\partial f}{\partial x_n}(x) \end{bmatrix}.$$

Suppose $f : \mathbb{R}^n \mapsto \mathbb{R}^m$ is a function such that each of its first-order partial derivatives exists on $\mathbb{R}^n$. This function takes a point $x \in \mathbb{R}^n$ as input and produces the vector $f(x) \in \mathbb{R}^m$ as output. Then the Jacobian matrix of f is defined to be an $m \times n$ matrix, denoted by J, whose (i, j)-th entry is $J_{ij} = \frac{\partial f_i}{\partial x_j}$, or explicitly

$$J = \begin{bmatrix} \frac{\partial f}{\partial x_1} & \cdots & \frac{\partial f}{\partial x_n} \end{bmatrix} = \begin{bmatrix} \nabla^{\mathrm{T}} f_1 \\ \vdots \\ \nabla^{\mathrm{T}} f_m \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}$$

where $\nabla^{\mathrm{T}} f_i$ is the transpose (row vector) of the gradient of the i-th component.

Note that a definition of the Jacobian transposed is possible compared to the one given here. We, however, choose this convention as it is the most commonly found.

1. Suppose we have a vector $x \in \mathbb{R}^d$, and let $f(x) = ||x||^2 = x^T x$. Compute the gradient $\nabla f(x)$.
2. Suppose we have a vector $x \in \mathbb{R}^d$ and a matrix $A \in \mathbb{R}^{d \times n}$. Let $f(x) = A^T x \in \mathbb{R}^n$. Compute the Jacobian of f with respect to x.

Chain Rule We first recall the basic chain rule when everything is scalar-valued. Suppose we have an input x, compute $y = g(x)$, then compute $z = f(y)$. Then the chain rule says

$$\frac{dz}{dx} = \frac{dy}{dx} \cdot \frac{dz}{dy}$$

.

Note that the reverse order is also possible. We, however, choose this order to make the calculations with the Jacobian easier.

Suppose y is a vector-valued function in $\mathbb{R}^n$ (x and z remain scalars). Summing over the contributions of each entry of y, we see $\partial z/\partial x$ is now a scalar given by:

$$\sum_{i=1}^{n} \frac{\partial y_i}{\partial x} \frac{\partial z}{\partial y_i}.$$

Finally, consider the case when x is also a vector in $\mathbb{R}^m$. From our calculation with scalar x and vector y, we know the j-th entry of $\partial z/\partial x$ is given by the partial derivative

$$\frac{\partial z}{\partial x_j} = \sum_{i=1}^{n} \frac{\partial y_i}{\partial x_j} \frac{\partial z}{\partial y_i} = \left(\frac{\partial y_i}{\partial x_j} \right)^T_{1 \le I \le n} \nabla_y z.$$

Stacking together the entries $\partial z/\partial x_j$ into a vector, we see that the gradient of the output z with respect to x is given by the product of **the transpose of the Jacobian matrix** of y with respect to x and the gradient of z with respect to y:

$$\overbrace{\frac{\partial z}{\partial x}}^{\mathbb{R}^m} = \overbrace{\frac{\partial y}{\partial x}}^{\mathbb{R}^{m\times n}} \overbrace{\frac{\partial z}{\partial y}}^{\mathbb{R}^n}$$

1. Suppose we have a vector $x \in \mathbb{R}^d$ and a matrix $A \in \mathbb{R}^{d\times n}$. Let $g(x) = A^T x \in R^n$, and let $f(y) = ||y||^2$. Compute the gradient of $f(g(x))$ with respect to x.

   Recall that $\nabla f(x) = 2x$ and $J_g(x) = A^T$

Problem 5

Curse of Dimensionality (*) This exercise shows several problems with high dimensionality.

A first result Let $X \sim \mathcal{N}(0, 1, R^d)$. In other words, every single vector component follows the law $\mathcal{N}(0, 1)$, and all components are independent.

We will prove that in this case $\forall d \geq 1 : |E(||X|| - \sqrt{d})| \leq 1/\sqrt{d}$ and $\forall d \geq 1 : V(||X||) \leq 2$.

In particular, the expectation $E(||X|| - \sqrt{d})$ converges to zero for $d \to \infty$.

1. Show that $E(||X||^2) = d$.
2. Show that:
$$||X|| - \sqrt{d} = \frac{||X||^2 - d}{2\sqrt{d}} - \frac{(||X||^2 - d)^2}{2\sqrt{d}(||X|| + \sqrt{d})^2} = S_d - R_d$$
3. Assuming that $Var(||X||^2) = 2d$, show that $0 \leq E(R_d) \leq \frac{1}{\sqrt{d}}$.
4. Deduce that $\forall d \geq 1 : |E(||X|| - \sqrt{d})| \leq 1/\sqrt{d}$
5. Show that $Var(||X||) \leq E((||X|| - \sqrt{d})^2$.

   Hint: the variance is invariant under the addition of a constant
6. Show that $E((||X|| - \sqrt{d})^2 = 2\sqrt{d}E(R_d)$
7. Deduce that $\forall d \geq 1 : V(||X||) \leq 2$
8. Interpret both results when d goes towards $+\infty$.

Lipschitz function approximation and curse of dimensionality *(Adapted from Stephane Mallat's lectures at Collège de France.)*

We will see that in high dimensions, the approximation error of the nearest-neighbor algorithm decreases very slowly with the number of examples: this is the curse of dimensionality. The nearest-neighbor algorithm is defined as such, with n examples from the training set $(x_i, f(x_i))_i$

$$\tilde{f}(x) = f(x_i) \text{ with } i = \arg\min_{i' \leq n} ||x - x_{i'}||$$

We say that $f : \Omega \to \mathbb{R}$ is locally Lipschitz-α continuous in $x \in \Omega$ if there exists a constant $C_x > 0$ and a polynomial p_x of degree $q \leq \lfloor \alpha \rfloor$ such that

$$\forall x' \in \Omega, |f(x') - p_x(x')| \leq C_x ||x - x'||^\alpha$$

We say that f is uniformly Lipschitz-α on Ω if f is Lipschitz-α in all $x \in \Omega$ and if there exists $C > 0$ such that $C_x \leq C$. This higher-order Lipschitzian regularity measures the decay rate α of the residual between $f(x')$ and its best polynomial approximation p_x starting from x. We will use it as an a priori, assuming that the functions f we want to learn belong to function classes functions F_α :

$$F_\alpha = \{f : \Omega \to \mathbb{R} : \text{f is uniformly Lipschitz-}\alpha\}$$

The challenge will then be to control the uniform approximation error $\max_{f \in F} \min_{h \in H} ||f - h||\infty$ of the H approximation classes as a function of $\log(|H|)$. We will calculate how many parameters and examples we need to obtain an approximation error. We will first show that if F is the class of uniformly Lipschitz functions ($\alpha = 1$) of constant C on a compact Ω, then:

$$\sup_{f \in F} ||f - \tilde{f}||_\infty = C\epsilon \text{ with } \epsilon = \sup_{x \in \Omega} \min_{i \leq n} ||x - x_i||$$

1. Show that $||f - \tilde{f}||_\infty = \sup_{x \in \Omega} |f(x) - \tilde{f}(x)| \leq C \sup_{x \in \Omega} \min_{i \leq n} ||x - x_i||$
2. Show that the upper bound is reached.

   Hint: Since Ω is compact, there exists x_j an element of $\{x_i\}_{i \leq n}$ and $x \in \Omega$ such that $||x - x_j|| \leq ||x - x_i||$ for all $i \leq n$

We must ensure that the distance to the nearest example is never too great to control the approximation error. We will calculate the number of examples n we need as a function of the dimension d. $\Omega = [0, 1]^d$ is assumed here. We're also assuming the examples are ideally distributed, so we have a lower bound on the algorithm error that could be obtained with randomly distributed examples.

For a fixed ϵ, we denote $B(x_i)$ the ball of radius ϵ around example x_i. We have $\sup_{x \in \Omega} \min_{i \leq n} ||x - x_i|| \leq \epsilon$ if and only if $B(x_i)$ form a covering of Ω,i.e., $\Omega \subseteq \cup_{i=1}^n B(x_i)$.

We will now see that the number of balls required increases exponentially with d. Especially, the minimum radius of n balls that span $\Omega = [0, 1]^d$ satisfies:

$$\frac{\sqrt{d} n^{-1/d}}{2} \geq \epsilon \geq \frac{\sqrt{d} n^{-1/d}}{2} \sqrt{\frac{2}{\pi e}} \left(1 + O_{+\infty}\left(\frac{\log d}{d}\right)\right)$$

1. The upper bound can be obtained by assuming the x_i form a regular lattice on all d directions with a step Δ. What is the number of balls in that case? Propose an upper bound of $\min_i ||x - x_i||^2$ with d and Δ.
2. The lower bound can be achieved by noticing the overall volume of the covering is greater or equal to the volume of Ω. Write this condition for n balls of radius ϵ part of a covering of Ω.
3. The volume of the ball of radius r is given by, when d is even:

   $$Vol(B_\epsilon) = \frac{\pi^{d/2} \epsilon^d}{(d/2)!}.$$

   Show that

   $$Vol(B_\epsilon) = (\frac{d}{2e\pi})^{-d/2} \frac{\epsilon^d}{\sqrt{\pi d}} (1 + O_{+\infty}(1/d))$$

   Hint: use the Stirling formula.

4. Deduce a lower bound of ϵ.

5. If F is the class of uniformly Lipschitz functions of constant C on Ω compact, propose a lower bound for $\sup_{f \in F} ||f - \tilde{f}||_\infty$

6. Conclude on the number of training samples required for this training method with respect to d for an error $C\epsilon$.