Fast supervised learning with guarantees

Ulysse Marteau-Ferey – Inria, ENS Paris, Sierra Project-Team

February 16, 2021
Introduction

Construction of a criterion of choice

Note on model selection

Finding the parameters

My phd

Supervised learning

Empirical risk minimization: OLS, Logistic regression, Ridge, Lasso, Quantile regression

Deep learning
Introduction
**What is ML?**

**Machine Learning**: artificial intelligence which can learn and model some phenomena without being explicitly programmed

Machine Learning ⊂ Statistics + Computer Sciences

- **Mathematics**: (Statistics, Optimization)
- **Computer science**: Machine learning, Applied Machine Learning
- **Domain expertise**: Dangerous software, Traditional research
What is ML?

**Machine Learning**: artificial intelligence which can learn and model some phenomena without being explicitly programmed

Machine Learning $\subseteq$ Statistics + Computer Sciences

- **Traditional programming**:
  - Input: 'zebra'
  - Output: 'tiger'

- **Machine learning**:
  - Training data: 'zebra', 'tiger'
  - Learning program
  - Input: training data
  - Output: chosen program / model
Supervised learning

Predict output $Y$ from some input data $X$. The training data has a known label $Y$.

Examples:
- $X$ is a picture, and $Y$ is a cat or a dog
- $X$ is a picture, and $Y \in \{0, \ldots, 9\}$ is a digit
- $X$ is videos captured by a robot playing table tennis, and $Y$ are the parameters of the robots to return the ball correctly
- $X$ is a music track and $Y$ are the audio signals of each instrument
Dog or cookie?
More formal formulation

**Goal**: Predict output $Y \in \mathcal{X}$ from some input data $X \in \mathcal{X}$.

**Predictor (program)**: $f: \mathcal{X} \to \mathcal{Y}$: find $f$ such that $f(X) = Y$

- **Traditional programming**:
  - Program
  - Input: 'zebra'
  - Output: 'tiger'

- **Machine learning**:
  - Training data
  - Learning program
  - Input: 'zebra', 'tiger'
  - Output: chosen program / model

**Machine learning model**: $\{f_\theta : \theta \in \Theta\}$: set of predictors

**Machine learning algorithm (learning program)**: Chooses the best $\theta$ (i.e. best $f_\theta$)

**Training data**: Examples $(x_1, y_1), \ldots, (x_n, y_n)$.
Example: linear model

Goal: Predict output $Y \in \mathbb{R}$ from some input data $X \in \mathbb{R}$.

Predictor (program): $f_{a,b}(x) = ax + b$

Machine learning model: $\{f_{a,b} : (a, b) \in \Theta = \mathbb{R}^2\}$: set of predictors

Machine learning algorithm (learning program): Chooses the best $a, b$ such that the line $aX + b$ fits $Y$

Training data: Examples $(x_1, y_1), ..., (x_n, y_n)$.
- Which $\theta$ do we want the algorithm to choose? What are the criteria (Statistics)?
- How do we choose the best model? (line, parabola?)
- How do we find the $\theta$ effectively (Optimization)?
Construction of a criterion of choice
Two points of views

- Idealistic point of view: best performance for every possible example \((x, y)\)
- The computer point of view: only sees examples \((x_1, y_1), \ldots, (x_n, y_n)\).

Need for both criterion: one to find the ”best” \(\theta\) during the algorithm, and one to evaluate it afterwards.
**Goal:** from training data, we want to predict an output $Y$ (or the best action) from the observation of some input $X$ using our model $f_\theta$

**Difficulties:** $Y$ is not a deterministic function of $X$. There can be some noise.

**Loss function:** $\ell$ to measure the difference between prediction $f_\theta(X)$ and the truth $Y$:

Loss given an example $(x, y)$: $\ell_{x,y}(\theta) = \ell(f_\theta(X), Y)$

<table>
<thead>
<tr>
<th></th>
<th>Least square regression</th>
<th>Classification</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A = Y$</td>
<td>$\mathbb{R}$</td>
<td>${0, 1, \ldots, K - 1}$</td>
</tr>
<tr>
<td>$\ell(a, y)$</td>
<td>$(a - y)^2$</td>
<td>$\mathbb{1}_{a \neq y}$</td>
</tr>
<tr>
<td>$R(f)$</td>
<td>$\mathbb{E}[(f(X) - Y)^2]$</td>
<td>$\mathbb{P}(f(X) \neq Y)$</td>
</tr>
<tr>
<td>$f^*$</td>
<td>$\mathbb{E}[Y</td>
<td>X]$</td>
</tr>
</tbody>
</table>
Ideal problem vs algorithmic problem

- **Ideal problem**: as if we had seen *everything*. We define the risk

\[
R(\theta) := \mathbb{E}[\ell_{X,Y}(\theta)] = \text{expected loss at } \theta
\]

Ideal goal: minimize \( R \)

- **Algorithmic problem**: using only the *training data*. Idea: estimate \( R(\theta) \) thanks to the training data with the *empirical risk*

\[
\hat{R}_n(\theta) := \frac{1}{n} \sum_{i=1}^{n} \ell_{X_i,Y_i}(\theta) \approx R(\theta) = \mathbb{E}[\ell_{X,Y}(\theta)]
\]

average error on training data ≈ expected error

We tell our algorithm to find \( \hat{\theta}_n \) by minimizing the empirical risk

\[
\hat{\theta}_n \in \arg \min_{\theta \in \Theta} \hat{R}_n(\theta).
\]
Note on model selection
Model selection: decomposition of the error

\[
\hat{R}_n(\theta) := \frac{1}{n} \sum_{i=1}^{n} \ell_{X_i, Y_i}(\theta) \approx R(\theta) = \mathbb{E}[\ell_{X, Y}(\theta)]
\]

average error on training data ≈ expected error

We tell our algorithm to find \( \hat{\theta}_n \) by minimizing the empirical risk

\[
\hat{\theta}_n \in \arg\min_{\theta \in \Theta} \hat{R}_n(\theta).
\]

Decomposition of the error

- **Approximation error**: model
- **Estimation error**: number of examples

\[
R(\hat{\theta}_n) = \min_{\theta \in \Theta} R(\theta) + R(\hat{\theta}_n) - \min_{\theta \in \Theta} R(\theta)
\]

Approximation error + Estimation error
Overfitting

<table>
<thead>
<tr>
<th>Complexity of F</th>
<th>Error</th>
<th>Training error</th>
<th>Expected error</th>
<th>Overfitting</th>
<th>Underfitting</th>
<th>Best choice</th>
</tr>
</thead>
</table>

Diagram showing the relationship between complexity of F and error. The graph illustrates the trade-off between training error and expected error, highlighting the optimal choice of complexity for minimizing error.
Overfitting: example in regression

Linear model: $Y = aX + b$

Training error: 0.1
Expected error: 0.08
Overfitting: example in regression

Cubic model: $Y = aX + bX^2 + cX^3 + d$

Training error: 0.03
Expected error: 0.05
Overfitting: example in regression

Polynomial model: Degree = 14

Training error: 0.01
Expected error: 0.17
Finding the parameters
\[ \hat{R}_n(\theta) := \frac{1}{n} \sum_{i=1}^{n} \ell_{X_i, Y_i}(\theta) \approx R(\theta) = \mathbb{E}[\ell_{X, Y}(\theta)] \]

average error on training data

expected error

We tell our algorithm to find \( \hat{\theta}_n \) by minimizing the empirical risk

\[ \hat{\theta}_n \in \arg \min_{\theta \in \Theta} \hat{R}_n(\theta). \]

How do we solve this problem? Optimization
Linear classification

Predict binary label $Y \in \{0, 1\}$ from $X$

Best linear classifier such that

$$f_{a,b}(X) = aX + b \begin{cases} 
\geq 0 & \Rightarrow \ Y = +1 \\
< 0 & \Rightarrow \ Y = 0
\end{cases}$$

Binary loss: $\ell$ is 0 if right, and 1 if wrong.

$$\hat{a}, \hat{b} = \arg\min_{a,b} \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f_{a,b}(X_i)).$$

This is not convex in $a, b$. Very hard to compute!
**Idea:** replace the loss with a convex loss

\[
\ell(w^\top X, y) = y \log (1 + e^{-w^\top X}) + (1 - y) \log (1 + e^{w^\top X})
\]

Convex losses: fast algorithms with guarantees
My phd
- Provide theoretical bounds in \( n \) (guarantees!)

\[
R(\hat{\theta}_n) = \min_{\theta \in \Theta} R(\theta) + R(\hat{\theta}_n) - \min_{\theta \in \Theta} R(\theta)
\]

- Approximation error
- Estimation error

- Provide a fast convex optimization algorithm to compute \( \hat{\theta}_n \).
Thank you for your attention!!