Machine Learning Basics

He He

New York University

September 8, 2021

Table of Contents

Rule-based approach

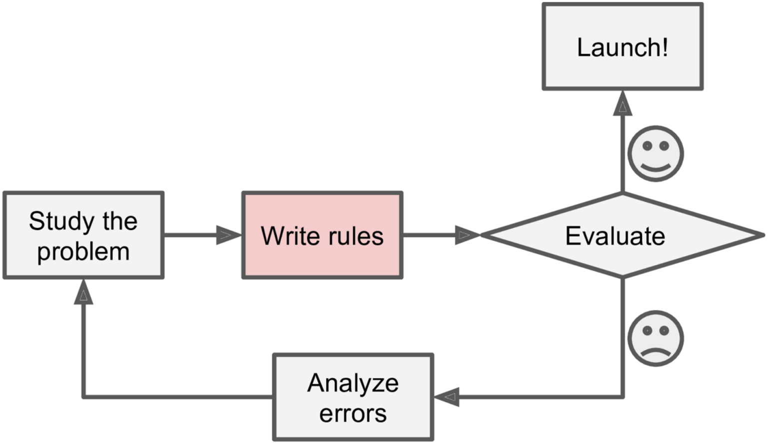


Figure: Fig 1-1 from *Hands-On Machine Learning with Scikit-Learn and TensorFlow* by Aurelien Geron (2017).

Machine learning approach

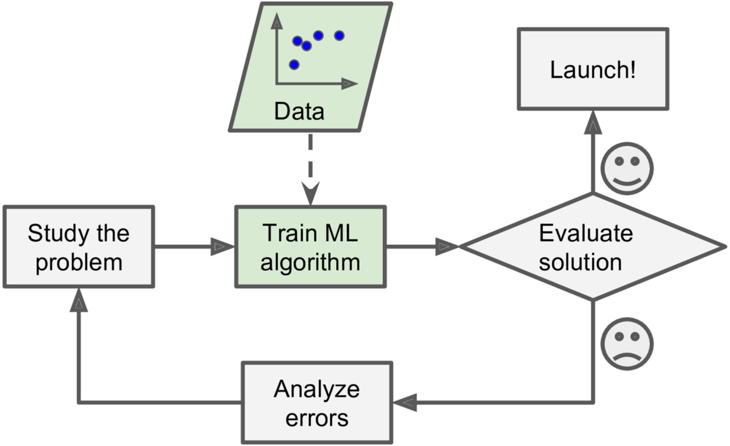


Figure: Fig 1-2 from *Hands-On Machine Learning with Scikit-Learn and TensorFlow* by Aurelien Geron (2017).

Example: spam filter

- Rules

  Contains “Viagra”
  Contains “Rolex”
  Subject line is all caps
  ...

- Learning from data
  1. Collect emails labeled as spam or non-spam
  2. (Design features)
  3. Learn a predictor

Pros and cons?

Keys to success

- Availability of large amounts of (annotated) data

  Scraping, crowdsourcing, expert annotation

- Generalize to unseen samples (test set)
  - Assume that there is a (unknown) data generating distribution: $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$
  - Training set: m samples from $\mathcal{D}$ $\left\{(x^{(i)}, y^{(i)})\right\}_{i=1}^{m}$
  - Learn model $h \colon \mathcal{X} \to \mathcal{Y}$
  - Goal: minimize $\mathbb{E}_{(x,y)\sim\mathcal{D}}[\text{error}(h, x, y)]$ (estimated on the test set)

Empirical risk minimization (ERM)

- Our goal is to minimize the expected loss (**risk**), but it cannot be computed (why?).
- How can we estimate it?

Empirical risk minimization (ERM)

- Our goal is to minimize the expected loss (**risk**), but it cannot be computed (why?).
- How can we estimate it?
- Minimize the average loss (**empirical risk**) on the training set

$$\min_h \frac{1}{m} \sum_{i=1}^{m} \text{error}(h, x^{(i)}, y^{(i)})$$

- In the limit of infinite samples, empirical risk converges to risk (LLN).
- Given limited data though, can we generalize by ERM?

Overfitting vs underfitting

- Trivial solution to (unconstrained) ERM: memorize the data points
- Need to extrapolate information from one part of the input space to unobserved parts!

Overfitting vs underfitting

- Trivial solution to (unconstrained) ERM: memorize the data points
- Need to extrapolate information from one part of the input space to unobserved parts!

- Constrain the prediction function to a subset, i.e. a **hypothesis space** $h \in \mathcal{H}$.
- Trade-off between complexity of $\mathcal{H}$ (approximiation error) and estimation error
- Question for us: how to choose a good $\mathcal{H}$ for certain domains

Table of Contents

Overall picture

1. Obtain training data $D_{\text{train}} = \{(x^{(i)}, y^{(i)})\}_{i=1}^{n}$.

2. Choose a loss function L and a hypothesis class $\mathcal{H}$ (domain knowledge).

3. Learn a predictor by minimizing the empirical risk (optimization).

Gradient descent

- The gradient of a function F at a point w is the direction of fastest increase in the function value
- To minimze $F(w)$, move in the opposite direction

$$w \leftarrow w - \eta \nabla_w F(w)$$

- Converge to a local minimum (also global minimum if $F(w)$ is **convex**) with carefully chosen step sizes

Convex optimization (unconstrained)

- A function $f : \mathbb{R}^d \to \mathbb{R}$ is convex if for all $x, y \in \mathbb{R}^d$ and $\theta \in [0, 1]$ we have

$$f(\theta x + (1 - \theta)y) \leq \theta f(x) + (1 - \theta)f(y) \ .$$

- f is concave if $-f$ is convex.
- Locally optimal points are also globally optimal.
- For unconstrained problems, x is optimal iff $\nabla f(x) = 0$.

Stochastic gradient descent

- **Gradient descent (GD)** for ERM

$$w \leftarrow w - \eta \nabla_w \underbrace{\sum_{i=1}^{n} L(x^{(i)}, y^{(i)}, f_w)}_{\text{training loss}}$$

Stochastic gradient descent

- **Gradient descent (GD)** for ERM

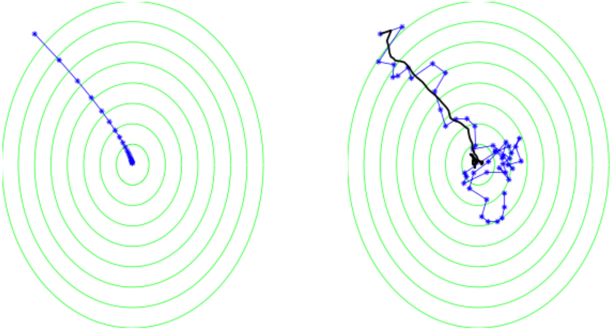
$$w \leftarrow w - \eta \nabla_w \underbrace{\sum_{i=1}^{n} L(x^{(i)}, y^{(i)}, f_w)}_{\text{training loss}}$$

- **Stochastic gradient descent (SGD)**: take noisy but faster steps

$$\text{For each } (x, y) \in D_{\text{train}} :$$
$$w \leftarrow w - \eta \nabla_w \underbrace{L(x, y, f_w)}_{\text{example loss}}$$

GD vs SGD

Figure: Minimize $1.25(x+6)^2 + (y-8)^2$



(Figure from "Understanding Machine Learning: From Theory to Algorithms".)

Stochastic gradient descent

- Each update is efficient in both time and space
- Can be slow to converge
- Popular in large-scale ML, including non-convex problems
- In practice,

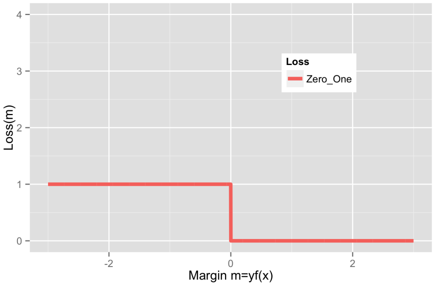
  Randomly sample examples.
  Fixed or diminishing step sizes, e.g. $1/t$, $1/\sqrt{t}$.
  Stop when objective does not improve.

Table of Contents

Zero-one loss

- Binary classification: $y \in \{+1, -1\}$.
  - Model: $f_w : \mathcal{X} \to \mathrm{R}$ parametrized by $w \in \mathrm{R}^d$.
  - Output prediction: $\mathrm{sign}(f_w(x))$.
- Zero-one (0-1) loss

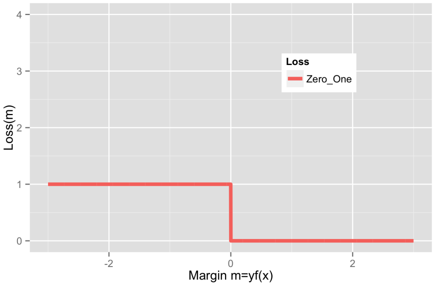
$$L(x, y, f_w) = \mathbb{I}\left[\mathrm{sign}(f_w(x)) = y\right] = \mathbb{I}\left[y f_w(x) \leq 0\right] \tag{1}$$



Zero-one loss

- Binary classification: $y \in \{+1, -1\}$.
  - Model: $f_w : \mathcal{X} \to \mathrm{R}$ parametrized by $w \in \mathrm{R}^d$.
  - Output prediction: $\mathrm{sign}(f_w(x))$.
- Zero-one (0-1) loss

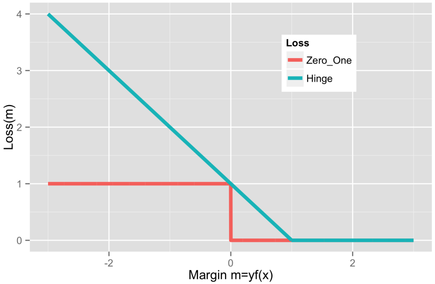
$$L(x, y, f_w) = \mathbb{I}\left[\mathrm{sign}(f_w(x)) = y\right] = \mathbb{I}\left[y f_w(x) \leq 0\right] \quad (1)$$



Not feasible for ERM

Hinge loss

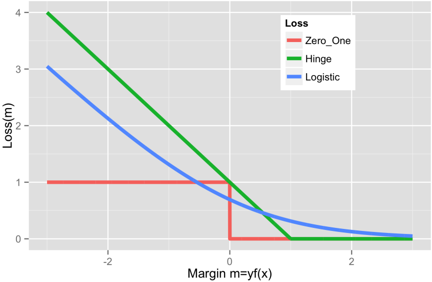
$$L(x, y, f_w) = \max(1 - yf_w(x), 0)$$



- Loss is zero if margin is larger than 1
- Not differentiable at margin $= 1$
- Subgradient: $\{g : f(x) \geq x_0 + g^T(x - x_0)\}$

Logistic loss

$$L(x, y, f_w) = \log(1 + e^{-yf_w(x)})$$



- Differentiable
- Always wants more margin (loss is never 0)

Summary

- Bias-complexity trade-off: choose hypothesis class based on prior knowledge
- Learning algorithm: empirical risk minimization
- Optimization: stochastic gradient descent