Machine Learning Basics

He He

NEW YORK UNIVERSITY

January 24, 2023

Table of Contents

Rule-based approach

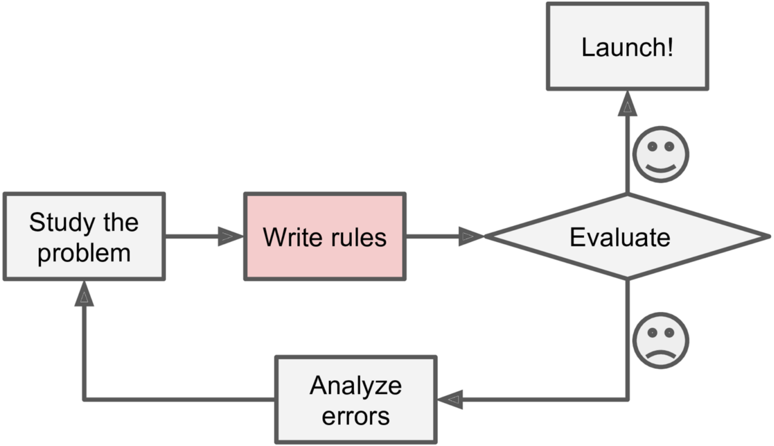


Figure: Fig 1-1 from *Hands-On Machine Learning with Scikit-Learn and TensorFlow* by Aurelien Geron (2017).

Machine learning approach

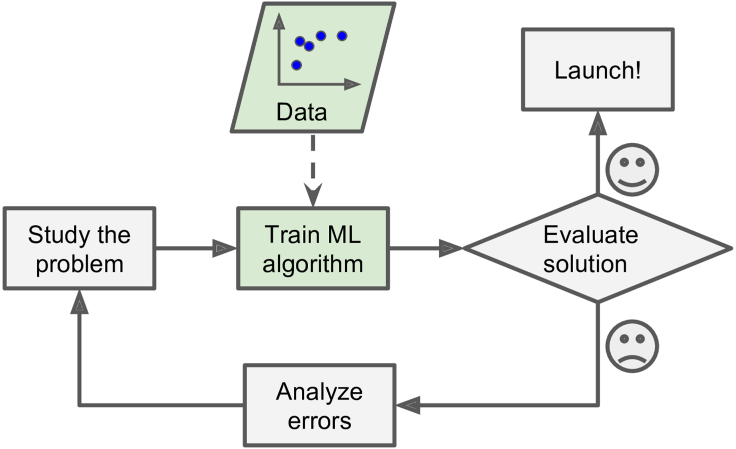


Figure: Fig 1-2 from *Hands-On Machine Learning with Scikit-Learn and TensorFlow* by Aurelien Geron (2017).

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, e.g., first word of the subject, nouns in the main text
  3. Learn a binary classifier

🤔 Pros and cons of each approach?

Key challenges in machine learning

- Availability of large amounts of (annotated) data
  - Data collection: scraping, crowdsourcing, expert annotation
  - Quality control: data quality can have large impact on the final model (garbage in garbage out)
  - Don’t take it for granted: always check the data source!

🤔 How would you collect a dataset for the spam filtering task?

Key challenges in machine learning

- **Generalize** to unseen samples
  - We want to build a model: $h\colon \mathcal{X}$ (input space) $\to \mathcal{Y}$ (output space)
  - It is easy to achieve high accuracy on the training set.
  - But we want the model to perform well on unseen data, too.
  - How should we evaluate the model?

Empirical risk minimization (ERM)

- Assume a data generating distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$ (e.g., spam writers and non-spam writers)

Empirical risk minimization (ERM)

- Assume a data generating distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$ (e.g., spam writers and non-spam writers)
- We have access to a training set: m samples from $\mathcal{D}$ $\left\{\left(x^{(i)}, y^{(i)}\right)\right\}_{i=1}^{m}$

Empirical risk minimization (ERM)

- Assume a data generating distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$ (e.g., spam writers and non-spam writers)
- We have access to a training set: m samples from $\mathcal{D}$ $\left\{(x^{(i)}, y^{(i)})\right\}_{i=1}^{m}$
- We can measure the goodness of a prediction $h(x)$ by comparing it against the groundtruth y using some **loss function**

Empirical risk minimization (ERM)

- Assume a data generating distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$ (e.g., spam writers and non-spam writers)
- We have access to a training set: m samples from $\mathcal{D}$ $\left\{(x^{(i)}, y^{(i)})\right\}_{i=1}^{m}$
- We can measure the goodness of a prediction $h(x)$ by comparing it against the groundtruth y using some **loss function**
- Our goal is to minimize the expected loss over $\mathcal{D}$ (**risk**):

$$\text{minimize} \quad \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\mathrm{error}(h, x, y)\right] \ ,$$

but it cannot be computed (why?).

Empirical risk minimization (ERM)

- Assume a data generating distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$ (e.g., spam writers and non-spam writers)
- We have access to a training set: m samples from $\mathcal{D}$ $\{(x^{(i)}, y^{(i)})\}_{i=1}^{m}$
- We can measure the goodness of a prediction $h(x)$ by comparing it against the groundtruth y using some **loss function**
- Our goal is to minimize the expected loss over $\mathcal{D}$ (**risk**):

$$\text{minimize} \quad \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\mathrm{error}(h, x, y)\right] \ ,$$

  but it cannot be computed (why?).
- Instead, we minimize the average loss on the training set (**empirical risk**)

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

Empirical risk minimization (ERM)

- Assume a data generating distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$ (e.g., spam writers and non-spam writers)
- We have access to a training set: m samples from $\mathcal{D}$ $\left\{(x^{(i)}, y^{(i)})\right\}_{i=1}^{m}$
- We can measure the goodness of a prediction $h(x)$ by comparing it against the groundtruth y using some **loss function**
- Our goal is to minimize the expected loss over $\mathcal{D}$ (**risk**):

$$\text{minimize} \quad \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\mathrm{error}(h, x, y)\right] \ ,$$

  but it cannot be computed (why?).
- Instead, we minimize the average loss on the training set (**empirical risk**)

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

- Key question: does small empirical risk imply small risk?

Overfitting vs underfitting

[board]

- Trivial solution to (unconstrained) ERM: memorize the data points
- Need to extrapolate information from one part of the input space to unobserved parts!
- Solution: constrain the prediction function to a subset, i.e. a **hypothesis space** $h \in \mathcal{H}$.

Overfitting vs underfitting

[board]

- Trivial solution to (unconstrained) ERM: memorize the data points
- Need to extrapolate information from one part of the input space to unobserved parts!
- Solution: constrain the prediction function to a subset, i.e. a **hypothesis space** $h \in \mathcal{H}$.
- Trade-off between complexity of $\mathcal{H}$ and generalization
- Question for us: how to choose a good $\mathcal{H}$ for certain domains

Summary

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).

Table of Contents

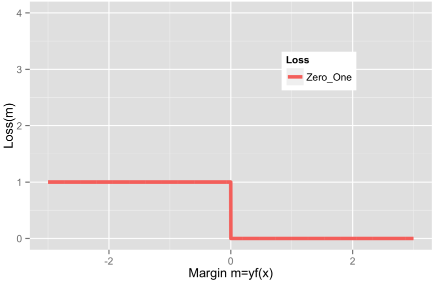
Setup

- Task: binary classification $y \in \{+1, -1\}$
- Model: $f_w : \mathcal{X} \to \mathbb{R}$ parametrized by $w \in \mathbb{R}^d$
  - Output a score for each example
- Prediction: $\mathrm{sign}(f_w(x))$
  - Positive scores are mapped to the positive class
- Goal: quantify the goodness of the model output $f_w(x)$ given y

Zero-one loss

First idea: check if the prediction is the same as the label

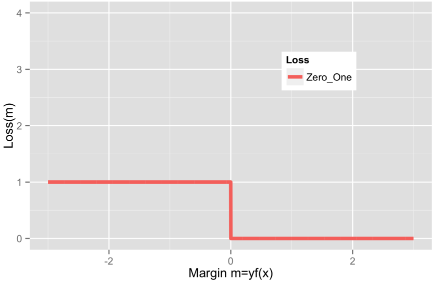
$$L(x, y, f_w) = \mathbb{I}\left[\operatorname{sign}(f_w(x)) = y\right] = \mathbb{I}\left[\underbrace{yf_w(x)}_{\text{margin}} \leq 0\right] \tag{1}$$



Zero-one loss

First idea: check if the prediction is the same as the label

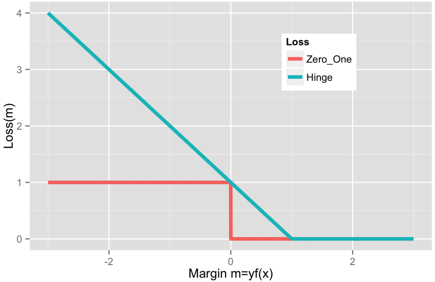
$$L(x, y, f_w) = \mathbb{I}\left[\operatorname{sign}(f_w(x)) = y\right] = \mathbb{I}\left[\underbrace{yf_w(x)}_{\text{margin}} \leq 0\right] \tag{1}$$



Problem: not differentiable

Hinge loss

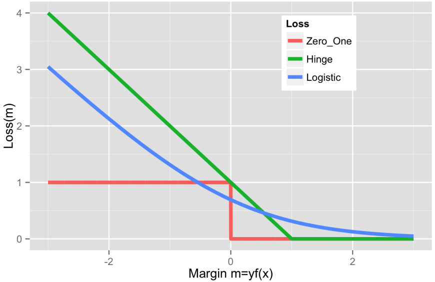
$$L(x, y, f_w) = \max(1 - yf_w(x), 0)$$



- A (sub)differentiable upperbound of the zero-one loss
- Not differentiable at margin $= 1$ (use subgradients)

Logistic loss

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



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

Summary

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).

Table of Contents

Gradient descent

- The gradient of a function F at a point $w \in \mathbb{R}^d$ 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 η

Stochastic gradient descent

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

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

Stochastic gradient descent

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

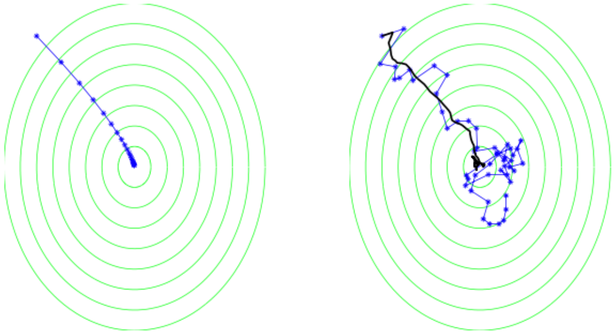
$$w \leftarrow w - \eta \nabla_w \underbrace{\sum_{i=1}^{n} L(x^{(i)}, y^{(i)}, w)}_{\text{training set loss}}$$

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

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

GD vs SGD

Figure: Minimize $1.25(x+6)^2 + (y-8)^2$. Example from "Understanding Machine Learning: From Theory to Algorithms"



SGD step is noisier as it gets closer to the optimum; need to reduce step size gradually.

SGD summary

- 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.
- Our main optimization techinque

Summary

- Choose hypothesis class based on domain knowledge
- Learning algorithm: empirical risk minimization
- Optimization: stochastic gradient descent