Linear Regression

Quách Đình Hoàng

Outline

- Empirical Risk Minimization
- Regression
- Linear Regression
- Regularized Regression
- Hyperparameter Tuning

Empirical Risk Minimization – ERM

- ERM is a common framework for supervised learning
- Given:
  - some labelled training dataset $D = \{(x^{(i)}, y^{(i)})\}_{i=1}^{n}$
  - a loss function $l: Y \times Y \rightarrow R$
  - a hypothesis class or set of functions F
- The goal is to find

$$\hat{f} = \underset{f \in F}{argmin} \sum_{i=1}^{n} l\left(f\left(x^{(i)}\right), y^{(i)}\right)$$

with the hope that

$$E_{p(x,y)}[l(f(x), y)] \approx \frac{1}{n} \sum_{i=1}^{n} l\left(f\left(x^{(i)}\right), y^{(i)}\right)$$

Empirical Risk Minimization – ERM

- ERM is a common framework for supervised learning
- Given:
  - a labelled training dataset $D = \{(x^{(i)}, y^{(i)})\}_{i=1}^{n}, x^{(i)} \in \mathbb{R}^{d+1}$
  - a loss function $l: Y \times Y \rightarrow R$
  - a hypothesis class or set of functions F
- The goal is to find

$$\hat{f} = \underset{f \in F}{argmin} \sum_{i=1}^{n} l\left(f\left(x^{(i)}\right), y^{(i)}\right)$$

- Depending on the choice of F and l, this objective function may be convex (easy to optimize) or non-convex (hard)

Regression

- Regression is a type of supervised learning
- Given:
  - a labelled training dataset $D = \{(x^{(i)}, y^{(i)})\}_{i=1}^{n}$
  - a loss function $l: Y \times Y \to R$, where $Y = \mathbb{R}$
  - a hypothesis class or set of functions F
- The goal is to find

$$\hat{f} = \underset{f \in F}{argmin} \sum_{i=1}^{n} l\left(f\left(x^{(i)}\right), y^{(i)}\right)$$

- Depending on the choice of F and l, this objective function may be convex (easy to optimize) or non-convex (hard)

Linear Regression (Ordinary Least Squares)

- Linear regression is a simplest type of regression
- Given:
  - a labelled training dataset $D = \{(x^{(i)}, y^{(i)})\}_{i=1}^{n}$
  - a loss function $l(y, y') = (y - y')^2$
  - a hypothesis class or set of functions F of the form $f(x) = w_0 + \sum_{i=1}^{d} w_i x_i$
- The goal is to find

$$\hat{f} = \underset{f \in F}{argmin} \sum_{i=1}^{n} l\left(f\left(x^{(i)}\right), y^{(i)}\right)$$

Linear Regression (Ordinary Least Squares)

- Classification is a type of supervised learning
- Given:
  - a labelled training dataset $D = \{(x^{(i)}, y^{(i)})\}_{i=1}^{n}$
  - a loss function $l(y, y') = (y - y')^2$
  - a hypothesis class or set of functions F of the form $f(x) = w_0 + \sum_{i=1}^{d} w_i x_i = w^T x$

$$w^T = (w_0, w_1, \cdots, w_d)$$
$$x^T = (1, x_1, \cdots, x_d)$$

- The goal is to find

$$\hat{w} = \underset{w}{argmin} \sum_{i=1}^{n} \left(w^T x^{(i)} - y^{(i)}\right)^2$$

Linear Regression (Ordinary Least Squares)

- Linear regression is a simplest type of regression
- Given:
  - a labelled training dataset $D = \{(x^{(i)}, y^{(i)})\}_{i=1}^{n}$
  - a loss function $l(y, y') = (y - y')^2$
  - F = all functions of the form $f(x) = w_0 + \sum_{i=1}^{d} w_i x_i = w^T x$

$$w^T = (w_0, w_1, \cdots, w_d)$$
$$x^T = (1, x_1, \cdots, x_d)$$

  - The goal is to find

$$\hat{w} = \underset{w}{argmin} \sum_{i=1}^{n} \left(w^T x^{(i)} - y^{(i)}\right)^2$$

or

$$\hat{w} = \underset{w}{argmin} \|Xw - y\|^2$$

where:

$$X = \begin{bmatrix} {x^{(1)}}^T \\ \vdots \\ {x^{(n)}}^T \end{bmatrix}, y = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(1)} \end{bmatrix}, x^{(i)} = \begin{bmatrix} 1 \\ x_1^{(i)} \\ \vdots \\ x_d^{(i)} \end{bmatrix}, w = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_d \end{bmatrix}$$

Linear Regression (Ordinary Least Squares)

$$L_D(w) = \|Xw - y\|^2 = (Xw - y)^T(Xw - y)$$

$$\vdots$$

$$\hat{w} = (X^T X)^{-1} X^T y$$

Regularizied Linear Regression

- Regularized empirical risk minimization that penalizes model complexityto deal with overfitting
- Given:
  - some labelled training dataset $D = \{(x^{(i)}, y^{(i)})\}_{i=1}^{n}$
  - a loss function $l: Y \times Y \to R$, where $Y = R$
  - a hypothesis class or set of functions F
  - a regularizer function $r: w \to R$
  - a regularization parameter λ
- The goal is to find

$$\hat{w} = \underset{w}{argmim} \sum_{i=1}^{n} l\left(f_w\left(x^{(i)}\right), y^{(i)}\right) + \lambda r(w)$$

Regularized Linear Regression

- Linear regression

$$L_D(w) = \sum_{i=1}^{n} \left(w^T x^{(i)} - y^{(i)}\right)^2$$

- Ridge regression (L_2 regularization)

$$\underset{w}{\text{minimize}} \left(L_D(w) + \frac{\lambda}{2} \|w\|_2^2 \right)$$

λ is hyperparameter

- Lasso regression (L_1 regularization)

$$\underset{w}{\text{minimize}} \left(L_D(w) + \frac{\lambda}{2} \|w\|_1 \right)$$

λ is hyperparameter

- ElasticNet regression ($L_1 + L_2$ regularization)

$$\underset{w}{\text{minimize}} \left(L_D(w) + \lambda \left(\alpha \|w\|_1 + \frac{1-\alpha}{2} \|w\|_2^2 \right) \right)$$

λ and α are hyperparameters

Ridge Regression

$$L_D(w) = \|Xw - y\|^2 + \lambda\|w\|^2 = (Xw - y)^T(Xw - y) + \lambda w^T w$$

$$\vdots$$

$$\hat{\theta} = (X^T X + \lambda I_{d+1})^{-1} X^T y$$

I_{d+1}: is the $(d+1) \times (d+1)$ identity matrix

Gradient Descent

- We're trying to minimize some function L
- Suppose at iteration t: we're get $w^{(t)}$
- With learning rate η, we update $w^{(t)}$ (at iteration $t+1$) as follow

$$w^{(t+1)} = w^{(t)} - \eta \nabla_w L(w^{(t)})$$

Gradient Descent for Linear Regression

- Dataset $D = \{(x^{(i)}, y^{(i)})\}_{i=1}^{n}$

1. Initialize $w^{(0)} = 0$ (zero vector) and set $t = 0$
2. While not converged
   - Compute the gradient

$$\nabla_w L_D(\theta^{(t)}) = 2X^T X w^{(t)} - 2X^T y$$

   - Update the weights

$$w^{(t+1)} = w^{(\mathrm{t})} - \eta \nabla_w L_D(w^{(t)})$$

   - Increment t:

$$t = t + 1$$

- Output $w^{(\mathrm{t})}$

Gradient Descent for Linear Regression – Change Step Size

- Dataset $D = \{(x^{(i)}, y^{(i)})\}_{i=1}^{n}$

1. Initialize $w^{(0)} = 0$ (zero vector) and set $t = 0$
2. While not converged
   - Compute the gradient

$$\nabla_w L_D(w^{(t)}) = 2X^T X w^{(t)} - 2X^T y$$

   - Update the weights

$$w^{(t+1)} = w^{(t)} - \eta_t \nabla_w L_D(w^{(t)}) \qquad \eta_t = \frac{\eta_0}{n\sqrt{t+1}}$$

   - Increment t:

$$t = t + 1$$

- Output $w^{(t)}$

Gradient Descent for Linear Regression – Change Step Size

- Dataset $D = \left\{\left(x^{(i)}, y^{(i)}\right)\right\}_{i=1}^{n}$

1. Initialize $w^{(0)} = 0$ (zero vector) and set $t = 0$
2. While not converged
   - Compute the gradient

$$\nabla_w L_D\left(w^{(t)}\right) = 2\sum_{i=1}^{n}\left({w^{(t)}}^T x^{(i)} - y^{(i)}\right) x^{(i)}$$

   - Update the weights

$$w^{(t+1)} = w^{(t)} - \eta_t \sum_{i=1}^{n}\left({w^{(t)}}^T x^{(i)} - y^{(i)}\right) x^{(i)} \quad \eta_t = \frac{\eta_0}{n\sqrt{t+1}}$$

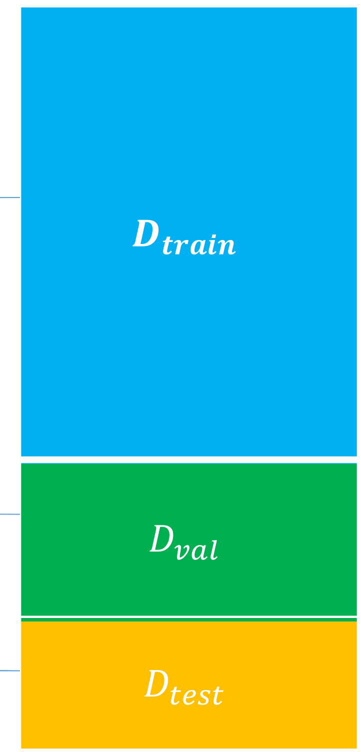
   - Increment t:

$$t = t + 1$$

- Output $w^{(t)}$

Hyperparameter Tuning

- Suppose we want to compare multiple hyperparameter settings $\lambda_1, \lambda_2, \cdots, \lambda_k$
- For $i = 1,2, \ldots, k$
  - Train a model on D_{train} using λ_i
- Evaluate each model on D_{val} and find the best hyperparameter setting, λ_{i^*}
- Compute the error of a model trained with λ_{i^*} on D_{test}



Summay

- Empirical Risk Minimization
- Regression
- Linear Regression
- Regularized Regression
- Hyperparameter Tuning