Introduction to Machine Learning

Linear Regression

Mingchen Gao

September 14, 2022

Outline

Contents

Taking the next step

Input Space, x

- $\mathbf{x} \in \{0,1\}^d$
- $\mathbf{x} \in \mathbb{R}^d$

Output Space, y

- $y \in \{0,1\}$
- $y \in \{-1,+1\}$
- $y \in \mathbb{R}$

1 Linear Regression

1.1 Problem Formulation

- There is one scalar **target** variable y
- There is one vector **input** variable x
- Inductive bias:

$$y = \mathbf{w}^\top \mathbf{x}$$

Linear Regression Learning Task
Learn $\mathbf{w}$ given training examples, $\langle \mathbf{X}, \mathbf{y} \rangle$.

The training data is denoted as $\langle \mathbf{X}, \mathbf{y} \rangle$, where $\mathbf{X}$ is a $N \times D$ data matrix consisting of N data examples such that each data example is a D dimensional vector. $\mathbf{y}$ is a $N \times 1$ vector consisting of corresponding target values for the examples in $\mathbf{X}$.

1. Probabilistic Interpretation

- y is assumed to be normally distributed

$$y \sim \mathcal{N}(\mathbf{w}^\top \mathbf{x}, \sigma^2)$$

- or, equivalently:
$$y = \mathbf{w}^\top \mathbf{x} + \epsilon$$
where $\epsilon \sim \mathcal{N}(0, \sigma^2)$
- y is a *linear combination* of the input variables
- *Given* $\mathbf{w}$ *and* σ^2*, one can find the probability distribution of* y *for a given* $\mathbf{x}$

1.2 Geometric Interpretation

2. Geometric Interpretation

- Fitting a straight line to d dimensional data
$$y = \mathbf{w}^\top \mathbf{x}$$
$$y = \mathbf{w}^\top \mathbf{x} = w_1 x_1 + w_2 x_2 + \ldots + w_d x_d$$
- Will pass through origin
- Add intercept
$$y = w_0 + w_1 x_1 + w_2 x_2 + \ldots + w_d x_d$$
- Equivalent to adding another column in $\mathbf{X}$ of 1s.

1.3 Learning Parameters

- Find $\mathbf{w}$ and σ^2 that maximize the likelihood of training data

$$\begin{aligned} \widehat{\mathbf{w}}_{MLE} &= (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y} \\ \widehat{\sigma}^2_{MLE} &= \frac{1}{N} (\mathbf{y} - \mathbf{X}\mathbf{w})^\top (\mathbf{y} - \mathbf{X}\mathbf{w}) \end{aligned}$$

The derivation of the MLE estimates can be done by maximizing the log-likelihood of the data set. The likelihood of the training data set is given by:

$$L(\mathbf{w}) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}\sigma} exp(-\frac{(y_i - \mathbf{w}^\top \mathbf{x_i})^2}{2\sigma^2})$$

The log-likelihood is given by:

$$LL(\mathbf{w}) = -\frac{1}{2}\log 2\pi - \log\sigma - \frac{1}{2\sigma^2}\sum_{i=1}^{N}(y_i - \mathbf{w}^\top \mathbf{x}_i)^2$$

This can be rewritten in matrix notation as:

$$LL(\mathbf{w}) = -\frac{1}{2}\log 2\pi - \log\sigma - \frac{1}{2\sigma^2}(\mathbf{y} - \mathbf{X}\mathbf{w})^\top(\mathbf{y} - \mathbf{X}\mathbf{w})$$

To maximize the log-likelihood, we first compute its derivative with respect to $\mathbf{w}$ and σ.

$$\begin{aligned}\frac{\partial LL(\mathbf{w})}{\partial \mathbf{w}} &= -\frac{1}{2\sigma^2}\frac{\partial}{\partial \mathbf{w}}(\mathbf{y} - \mathbf{X}\mathbf{w})^\top(\mathbf{y} - \mathbf{X}\mathbf{w}) \\ &= -\frac{1}{2\sigma^2}\frac{\partial}{\partial \mathbf{w}}(\mathbf{y}^\top\mathbf{y} + \mathbf{w}^\top\mathbf{X}^\top\mathbf{X}\mathbf{w} - 2\mathbf{y}^\top\mathbf{X}\mathbf{w})\end{aligned}$$

Note that, we use the fact that $(\mathbf{X}\mathbf{w})^\top\mathbf{y} = \mathbf{y}^\top\mathbf{X}\mathbf{w}$, since both quantities are scalars and the transpose of a scalar is equal to itself. Continuing with the derivative:

$$\frac{\partial LL(\mathbf{w})}{\partial \mathbf{w}} = -\frac{1}{2\sigma^2}(2\mathbf{w}^\top\mathbf{X}^\top\mathbf{X} - 2\mathbf{y}^\top\mathbf{X})$$

Setting the derivative to 0, we get:

$$\begin{aligned}2\mathbf{w}^\top\mathbf{X}^\top\mathbf{X} - 2\mathbf{y}^\top\mathbf{X} &= 0 \\ \mathbf{w}^\top\mathbf{X}^\top\mathbf{X} &= \mathbf{y}^\top\mathbf{X} \\ (\mathbf{X}^\top\mathbf{X})^\top\mathbf{w} &= \mathbf{X}^\top\mathbf{y} \text{ (Taking transpose both sides)} \\ (\mathbf{X}^\top\mathbf{X})\mathbf{w} &= \mathbf{X}^\top\mathbf{y} \\ \mathbf{w} &= (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y}\end{aligned}$$

In a similar fashion, one can set the derivative to 0 with respect to σ and plug in the the optimal value of $\mathbf{w}$

- Minimize *squared loss*

$$J(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N}(y_i - \mathbf{w}^\top \mathbf{x}_i)^2$$

- Make prediction ($\mathbf{w}^\top \mathbf{x}_i$) as close to the target (y_i) as possible
- Least squares estimate

$$\widehat{\mathbf{w}} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}$$

The derivation of the least squares estimate follows the exact computation as the MLE derivation above, once the expression of the squared loss is converted into matrix notation, i.e.,

$$\begin{aligned} J(\mathbf{w}) &= \frac{1}{2} \sum_{i=1}^{N} (y_i - \mathbf{w}^\top \mathbf{x}_i)^2 \\ &= \frac{1}{2} (\mathbf{y} - \mathbf{X}\mathbf{w})^\top (\mathbf{y} - \mathbf{X}\mathbf{w}) \end{aligned}$$

- Minimize the squared loss using *Gradient Descent*

$$J(\mathbf{w}) = \frac{1}{2} \sum_{i=1}^{N} (y_i - \mathbf{w}^\top \mathbf{x}_i)^2$$

- The matrix inversion is expensive or numerically unstable.

The analytical approach discussed earlier involves a matrix inversion ($(\mathbf{X}^\top \mathbf{X})^{-1}$) which is a $(D+1) \times (D+1)$ matrix. Alternatively, one could solve a system of equations. When D is large, this inversion can be computationally expensive ($O * D^3$) for standard matrix inversion. Moreover, often, the linear system might have singularities and inversion or solving the system of equations might yield numerically unstable results.

To compute the gradient update rule one can differentiate the error with respect to each entry of $\mathbf{w}$.

$$\begin{aligned} \frac{\partial J(\mathbf{w})}{\partial w_j} &= \frac{1}{2} \frac{\partial}{\partial w_j} \sum_{i=1}^{N} (y_i - \mathbf{w}^\top \mathbf{x}_i)^2 \\ &= \sum_{i=1}^{N} (\mathbf{w}^\top \mathbf{x}_i - y_i) x_{ij} \end{aligned}$$

Using the above result, one can perform repeated updates of the weights:

$$w_j := w_j - \eta \frac{\partial J(\mathbf{w})}{\partial w_j}$$

2 Recap

Geometric

$$y = \mathbf{w}^\top \mathbf{x}$$

$$J(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N}(y_i - \mathbf{w}^\top \mathbf{x}_i)^2$$

1. Least Squares
$$\widehat{\mathbf{w}} = (\mathbf{X}^\top \mathbf{X})^{-1}\mathbf{X}^\top \mathbf{y}$$
2. Gradient Descent

Probabilistic

$$p(y) = \mathcal{N}(\mathbf{w}^\top \mathbf{x}, \sigma^2)$$

1. Maximum Likelihood Estimation

$$\begin{aligned} \widehat{\mathbf{w}} &= (\mathbf{X}^\top \mathbf{X})^{-1}\mathbf{X}^\top \mathbf{y} \\ \widehat{\sigma}^2_{MLE} &= \frac{1}{N}\sum_{i=1}^{N}(\mathbf{y} - \mathbf{X}\mathbf{w})^\top(\mathbf{y} - \mathbf{X}\mathbf{w}) \end{aligned}$$

2.1 Issues with Linear Regression

1. Not truly Bayesian
2. Susceptible to outliers
3. *Too simplistic* - Underfitting
4. No way to control overfitting
5. Unstable in presence of correlated input attributes
6. Gets "confused" by unnecessary attributes

3 Bayesian Linear Regression

4 Bayesian Regression

- A zero-mean Gaussian prior

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|0, \tau^2 I)$$

- What is posterior of $\mathbf{w}$

$$p(\mathbf{w}|\mathcal{D}) \propto \prod_i \mathcal{N}(y_i|\mathbf{w}^\top \mathbf{x}_i, \sigma^2) p(\mathbf{w})$$

- Posterior is also Gaussian
- Regularized least squares estimate of $\mathbf{w}$

$$\arg\max_{\mathbf{w}} \sum_{i=1}^{N} log\mathcal{N}(y_i|\mathbf{w}^\top \mathbf{x}_i, \sigma^2) + \log \mathcal{N}(\mathbf{w}|0, \tau^2 I)$$

4.1 Estimating Bayesian Regression Parameters

- Prior for $\mathbf{w}$

$$\mathbf{w} \sim \mathcal{N}(\mathbf{w}|\mathbf{0}, \tau^2 \mathbf{I}_D)$$

- Posterior for $\mathbf{w}$

$$\bar{\mathbf{w}}_{\mathbf{MAP}} = (\mathbf{X}^\top \mathbf{X} + \frac{\sigma^{\mathbf{2}}}{\tau^{\mathbf{2}}} \mathbf{I}_{\mathbf{N}})^{-\mathbf{1}} \mathbf{X}^\top \mathbf{y}$$

$$\bar{\sigma}_{\mathbf{MAP}} = \sigma^{\mathbf{2}} (\mathbf{X}^\top \mathbf{X} + \frac{\sigma^{\mathbf{2}}}{\tau^{\mathbf{2}}} \mathbf{I}_{\mathbf{N}})^{-\mathbf{1}}$$

- "Penalize" large values of $\mathbf{w}$, equivalent to *Ridge Regression*

The denominator term in the posterior above can be computed as the marginal likelihood of data by marginalizing $\mathbf{w}$:

$$p(\mathbf{y}|\mathbf{X}) = \int p(\mathbf{y}|\mathbf{X}, \mathbf{w}) p(\mathbf{w}) d\mathbf{w}$$

One can compute the posterior for $\mathbf{w}$ as follows. We first show that the likelihood of $\mathbf{y}$, i.e., all target values in the training data, can be jointly modeled as a Gaussian as follows:

$$\begin{aligned} p(\mathbf{y}|\mathbf{X},\mathbf{w}) &= \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} exp\left(-\frac{1}{2\sigma^2}(y_i - \mathbf{w}\top\mathbf{x}_i)\right) \\ &= \frac{1}{(2\pi\sigma^2)^{N/2}} exp\left(-\frac{1}{2\sigma^2}|\mathbf{y}-\mathbf{X}\mathbf{w}|^2\right) \\ &= \mathcal{N}(\mathbf{X}\mathbf{w}, \sigma^2\mathbf{I}_N) \end{aligned}$$

Ignoring the denominator which does not depend on $\mathbf{w}$:

$$\begin{aligned} p(\mathbf{w}|\mathbf{y},\mathbf{X}) &\propto exp(-\frac{1}{2\sigma^2}(\mathbf{y}-\mathbf{X}\mathbf{w})^\top(\mathbf{y}-\mathbf{X}\mathbf{w}))exp(-\frac{1}{2\tau^2}\mathbf{w}^\top\mathbf{w}) \\ &\propto exp(-\frac{1}{2}(\mathbf{w}-\bar{\mathbf{w}})^\top(\frac{1}{\sigma^2}\mathbf{X}^\top\mathbf{X}+\frac{1}{\tau^2}\mathbf{I}_N)(\mathbf{w}-\bar{\mathbf{w}})) \end{aligned}$$

where $\bar{\mathbf{w}} = (\mathbf{X}^\top\mathbf{X} + \frac{\sigma^2}{\tau^2}\mathbf{I}_N)^{-1}\mathbf{X}\mathbf{y}$.

4.2 Prediction with Bayesian Regression

- For a new $\mathbf{x}^*$, predict y^*
- Point estimate of y^*

$$y^* = \widehat{\mathbf{w}}_{MLE}^\top \mathbf{x}^*$$

- Treating y as a Gaussian random variable

$$p(y^*|\mathbf{x}^*) = \mathcal{N}(\widehat{\mathbf{w}}_{MLE}^\top\mathbf{x}^*, \widehat{\sigma}^2_{MLE})$$

$$p(y^*|\mathbf{x}^*) = \mathcal{N}(\widehat{\mathbf{w}}_{MAP}^\top\mathbf{x}^*, \widehat{\sigma}^2_{MAP})$$

- Treating y and $\mathbf{w}$ as random variables

$$p(y^*|\mathbf{x}^*) = \int p(y^*|\mathbf{x}^*,\mathbf{w})p(\mathbf{w}|\mathbf{X},\mathbf{y})d\mathbf{w}$$

- This is also *Gaussian*!

5 Handling Non-linear Relationships

- Replace $\mathbf{x}$ with non-linear functions $\boldsymbol{\phi}(\mathbf{x})$

$$p(y|\mathbf{x}, \boldsymbol{\theta}) \sim \mathcal{N}(\mathbf{w}^\top \boldsymbol{\phi}(\mathbf{x}))$$

- Model is still linear in $\mathbf{w}$
- Also known as **basis function expansion**

Example 1.

$$\boldsymbol{\phi}(x) = [1, x, x^2, \ldots, x^p]$$

- Increasing p results in more complex fits

5.1 Handling Overfitting via Regularization

How to Control Overfitting?

- Use simpler models (linear instead of polynomial)
  - Might have poor results (underfitting)
- Use regularized complex models

$$\widehat{\boldsymbol{\Theta}} = \arg\min_{\boldsymbol{\Theta}} J(\boldsymbol{\Theta}) + \lambda R(\boldsymbol{\Theta})$$

- $R()$ corresponds to the penalty paid for complexity of the model

l_2 Regularization

Ridge Regression

$$\widehat{\mathbf{w}} = \arg\min_{\mathbf{w}} J(\mathbf{w}) + \lambda \|\mathbf{w}\|_2^2$$

- Helps in reducing impact of correlated inputs

Exact Loss Function

$$J(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N}(y_i - \mathbf{w}^\top \mathbf{x}_i)^2 + \frac{1}{2}\lambda||\mathbf{w}||_2^2$$

Ridge Estimate of w

$$\widehat{\mathbf{w}}_{MAP} = (\lambda \mathbf{I}_D + \mathbf{X}^\top \mathbf{X})^{-1}\mathbf{X}^\top \mathbf{y}$$

- Equivalent to MAP estimate for Bayesian Regression with Gaussian prior on $\mathbf{w}$

l_1 Regularization

Least Absolute Shrinkage and Selection Operator - LASSO

$$\widehat{\mathbf{w}} = \arg\min_{\mathbf{w}} J(\mathbf{w}) + \lambda|\mathbf{w}|$$

- Helps in feature selection – favors sparse solutions
- Optimization is not as straightforward as in Ridge regression
  - Gradient not defined for $w_i = 0, \forall i$
- Equivalent to MAP estimate for Bayesian Regression with *Laplace* prior on $\mathbf{w}$

Laplace Distribution

$$p(\mathbf{w}) = \frac{1}{2b} \exp\left(-\frac{|\mathbf{w} - \boldsymbol{\mu}|}{b}\right)$$

- Has two parameters, $\boldsymbol{\mu}$ and b
- Has a less "fatter" tail than Gaussian

5.2 Elastic Net Regularization

LASSO vs. Ridge

- Both control overfitting
- Ridge helps reduce impact of correlated inputs, LASSO helps in feature selection

Elastic Net Regularization

$$\widehat{\mathbf{w}} = \arg\min_{\mathbf{w}} J(\mathbf{w}) + \lambda|\mathbf{w}| + (1 - \lambda)\|\mathbf{w}\|_2^2$$

- The best of both worlds
- Again, optimizing for $\mathbf{w}$ is not straightforward

6 Handling Outliers in Regression

- Linear regression training gets impacted by the presence of outliers
- The square term in the exponent of the Gaussian pdf is the culprit
  - Equivalent to the square term in the loss
- How to handle this (*Robust Regression*)?
- Probabilistic:
  - Use a different distribution instead of Gaussian for $p(y|\mathbf{x})$
  - Robust regression uses Laplace distribution

$$p(y|\mathbf{x}) \sim Laplace(\mathbf{w}^\top \mathbf{x}, b)$$

- Geometric:
  - *Least absolute deviations* instead of least squares

$$J(\mathbf{w}) = \sum_{i=1}^{N} |y_i - \mathbf{w}^\top \mathbf{x}|$$

References

Murphy Book Chapter 11