Week-5: Linear Regression - Introduction

Sherry Thomas
21f3001449

Contents

Abstract

The week commences with an exploration of Supervised Learning, specifically focusing on the topic of Regression. The aim is to provide a comprehensive understanding of the underlying mechanism of this popular machine learning technique, and its various applications. Additionally, this study delves into the variants of Regression, including kernel regression, and examines the probabilistic aspects of the technique.

Introduction to Supervised Learning

Supervised learning, a fundamental machine learning technique, involves training an algorithm on labeled data where the target variable or outcome is known. The primary objective of supervised learning is to establish a mapping between input data and corresponding output variables.

Given a dataset $\mathbf{x}_1, \ldots, \mathbf{x}_n$, where each $\mathbf{x}_i$ belongs to $\mathbb{R}^d$, the corresponding labels $\mathbf{y}_1, \ldots, \mathbf{y}_n$ can fall into the following categories:

- Regression: $\mathbf{y}_i \in \mathbb{R}$ (e.g., Rainfall Prediction)
- Binary Classification: $\mathbf{y}_i \in \{0, 1\}$ (e.g., Distinguishing between cats and dogs)
- Multi-class Classification: $\mathbf{y}_i \in \{0, 1, \ldots, K\}$ (e.g., Digit classification)

Linear Regression

Linear regression is a supervised learning algorithm employed to predict a continuous output variable based on one or more input features, assuming a linear relationship between the input and output variables. The primary objective of linear regression is to determine the line of best fit that minimizes the sum of squared errors between the predicted and actual output values.

Given a dataset $\mathbf{x}_1, \ldots, \mathbf{x}_n$ where each $\mathbf{x}_i$ belongs to $\mathbb{R}^d$, and the corresponding labels $\mathbf{y}_1, \ldots, \mathbf{y}_n$ belong to $\mathbb{R}$, the goal of linear regression is to find a mapping between the input and output variables, represented as follows:

$$h : \mathbb{R}^d \to \mathbb{R}$$

The error for this mapping function can be quantified as:

$$\text{error}(h) = \sum_{i=1}^{n} (h(\mathbf{x}_i) - \mathbf{y}_i)^2$$

Ideally, this error should be minimized, which occurs when $h(\mathbf{x}_i) = \mathbf{y}_i$ for all i. However, achieving this may only result in memorizing the data and its outputs, which is not a desired outcome.

To mitigate the memorization problem, introducing a structure to the mapping becomes necessary. The simplest and commonly used structure is linear, which we will adopt as the underlying structure for our data.

Let $\mathcal{H}_{\text{linear}}$ denote the solution space for the mapping in the linear domain:

$$\mathcal{H}_{\text{linear}} = \left\{ h_w : \mathbb{R}^d \to \mathbb{R} \text{ s.t. } h_w(\mathbf{x}) = \mathbf{w}^T \mathbf{x} \ \forall \mathbf{w} \in \mathbb{R}^d \right\}$$

Thus, our objective is to minimize:

$$\min_{h \in \mathcal{H}_{\text{linear}}} \sum_{i=1}^{n} (h(\mathbf{x}_i) - \mathbf{y}_i)^2$$

Equivalently,

$$\min_{\mathbf{w} \in \mathbb{R}^d} \sum_{i=1}^{n} (\mathbf{w}^T \mathbf{x}_i - \mathbf{y}_i)^2$$

Optimizing the above objective is the main aim of the linear regression algorithm.

Optimizing the Error Function

The minimization equation can be expressed in vectorized form as:

$$\min_{\mathbf{w}\in\mathbb{R}^d} \|\mathbf{X}^T\mathbf{w} - \mathbf{y}\|_2^2$$

Defining a function $f(\mathbf{w})$ that captures this minimization problem, we have:

$$\begin{aligned} f(\mathbf{w}) &= \min_{\mathbf{w}\in\mathbb{R}^d} \|\mathbf{X}^T\mathbf{w} - \mathbf{y}\|_2^2 \\ f(\mathbf{w}) &= (\mathbf{X}^T\mathbf{w} - \mathbf{y})^T(\mathbf{X}^T\mathbf{w} - \mathbf{y}) \\ \therefore \nabla f(\mathbf{w}) &= 2(\mathbf{X}\mathbf{X}^T)\mathbf{w} - 2(\mathbf{X}\mathbf{y}) \end{aligned}$$

Setting the gradient equation to zero, we obtain:

$$\begin{aligned} (\mathbf{X}\mathbf{X}^T)\mathbf{w}^* &= \mathbf{X}\mathbf{y} \\ \therefore \mathbf{w}^* &= (\mathbf{X}\mathbf{X}^T)^+\mathbf{X}\mathbf{y} \end{aligned}$$

Here, $(\mathbf{X}\mathbf{X}^T)^+$ represents the pseudo-inverse of $\mathbf{X}\mathbf{X}^T$.

Further analysis reveals that $\mathbf{X}^T\mathbf{w}$ corresponds to the projection of the labels onto the subspace spanned by the features.

Gradient Descent

The normal equation for linear regression, as shown above, involves calculating $(\mathbf{X}\mathbf{X}^T)^+$, which can be computationally expensive with a complexity of $O(d^3)$.

Since $\mathbf{w}^*$ represents the solution of an unconstrained optimization problem, it can be solved using gradient descent. The iterative formula for gradient descent is:

$$\begin{aligned} \mathbf{w}^{t+1} &= \mathbf{w}^t - \eta^t \nabla f(\mathbf{w}^t) \\ \therefore \mathbf{w}^{t+1} &= \mathbf{w}^t - \eta^t \left[2(\mathbf{X}\mathbf{X}^T)\mathbf{w}^t - 2(\mathbf{X}\mathbf{y})\right] \end{aligned}$$

Here, η is a scalar that controls the step-size of the descent, and t represents the current iteration.

Even in the above equation, the calculation of $\mathbf{X}\mathbf{X}^T$ is required, which remains computationally expensive. Is there a way to further enhance this process?

Stochastic Gradient Descent

Stochastic gradient descent (SGD) is an optimization algorithm widely employed in machine learning to minimize the loss function of a model by determining the optimal parameters. Unlike traditional gradient descent, which updates the model parameters based on the entire dataset, SGD updates the parameters using a randomly selected subset of the data, known as a batch. This approach leads to faster training times and makes SGD particularly suitable for handling large datasets.

Instead of updating $\mathbf{w}$ using the entire dataset at each step t, SGD leverages a small randomly selected subset of k data points to update $\mathbf{w}$. Consequently, the new gradient becomes $2(\tilde{\mathbf{X}}\tilde{\mathbf{X}}^T\mathbf{w}^t - \tilde{\mathbf{X}}\tilde{\mathbf{y}})$, where $\tilde{\mathbf{X}}$ and $\tilde{\mathbf{y}}$ represent small samples randomly chosen from the dataset. This strategy is feasible since $\tilde{\mathbf{X}} \in \mathbb{R}^{d\times k}$, which is considerably smaller compared to $\mathbf{X}$.

After T rounds of training, the final estimate is obtained as follows:

$$\mathbf{w}_{\text{SGD}}^T = \frac{1}{T}\sum_{i=1}^{T}\mathbf{w}^i$$

The stochastic nature of SGD contributes to optimal convergence to a certain extent.

Kernel Regression

What if the data points reside in a non-linear subspace? Similar to dealing with non-linear data clustering, kernel functions are employed in this scenario as well.

Let $\mathbf{w}^* = \mathbf{X}\alpha^*$, where $\alpha^* \in \mathbb{R}^n$.

$$\begin{aligned}
\mathbf{X}\alpha^* &= \mathbf{w}^* \\
\therefore \mathbf{X}\alpha^* &= (\mathbf{X}\mathbf{X}^T)^+\mathbf{X}\mathbf{y} \\
(\mathbf{X}\mathbf{X}^T)\mathbf{X}\alpha^* &= (\mathbf{X}\mathbf{X}^T)(\mathbf{X}\mathbf{X}^T)^+\mathbf{X}\mathbf{y} \\
(\mathbf{X}\mathbf{X}^T)\mathbf{X}\alpha^* &= \mathbf{X}\mathbf{y} \\
\mathbf{X}^T(\mathbf{X}\mathbf{X}^T)\mathbf{X}\alpha^* &= \mathbf{X}^T\mathbf{X}\mathbf{y} \\
(\mathbf{X}^T\mathbf{X})^2\alpha^* &= \mathbf{X}^T\mathbf{X}\mathbf{y} \\
\mathbf{K}^2\alpha^* &= \mathbf{K}\mathbf{y} \\
\therefore \alpha^* &= \mathbf{K}^{-1}\mathbf{y}
\end{aligned}$$

Here, $\mathbf{K} \in \mathbb{R}^{n\times n}$, and it can be obtained using a kernel function such as the Polynomial Kernel or RBF Kernel.

To predict using α and the kernel function, let $\mathbf{X}_{\text{test}} \in \mathbb{R}^{d\times m}$ represent the test dataset. The prediction is made as follows:

$$\mathbf{w}^*\phi(\mathbf{X}_{\text{test}}) = \sum_{i=1}^{n}\alpha_i^* k(\mathbf{x}_i, \mathbf{x}_{\text{test}_i})$$

Here, α_i^* denotes the importance of the i-th data point in relation to $\mathbf{w}^*$, and $k(\mathbf{x}_i, \mathbf{x}_{\text{test}_i})$ signifies the similarity between $\mathbf{x}_{\text{test}_i}$ and $\mathbf{x}_i$.

Probabilistic View of Linear Regression

Consider a dataset $\mathbf{x}_1, \dots, \mathbf{x}_n$ with $\mathbf{x}_i \in \mathbb{R}^d$, and the corresponding labels $\mathbf{y}_1, \dots, \mathbf{y}_n$ with $\mathbf{y}_i \in \mathbb{R}$. The probabilistic view of linear regression assumes that the target variable $\mathbf{y}_i$ can be modeled as a linear combination of the input features $\mathbf{x}_i$, with an additional noise term ϵ following a zero-mean Gaussian distribution with variance σ^2. Mathematically, this can be expressed as:

$$\mathbf{y}_i = \mathbf{w}^T \mathbf{x}_i + \epsilon_i$$

where $\mathbf{w} \in \mathbb{R}^d$ represents the weight vector that captures the relationship between the inputs and the target variable.

To estimate the weight vector $\mathbf{w}$ that best fits the data, we can apply the principle of Maximum Likelihood (ML). The ML estimation seeks to find the parameter values that maximize the likelihood of observing the given data.

Assuming that the noise term ϵ_i follows a zero-mean Gaussian distribution with variance σ^2, we can express the likelihood function as:

$$\begin{aligned}
\mathcal{L}(\mathbf{w}; \mathbf{X}, \mathbf{y}) &= P(\mathbf{y}|\mathbf{X}; \mathbf{w}) \\
&= \prod_{i=1}^{n} P(\mathbf{y}_i|\mathbf{x}_i; \mathbf{w}) \\
&= \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(\mathbf{w}^T \mathbf{x}_i - \mathbf{y}_i)^2}{2\sigma^2}\right)
\end{aligned}$$

Taking the logarithm of the likelihood function, we have:

$$\begin{aligned}
\log \mathcal{L}(\mathbf{w}; \mathbf{X}, \mathbf{y}) &= \sum_{i=1}^{n} \log\left(\frac{1}{\sqrt{2\pi}\sigma}\right) - \frac{(\mathbf{w}^T \mathbf{x}_i - \mathbf{y}_i)^2}{2\sigma^2} \\
&= -\frac{n}{2} \log(2\pi\sigma^2) - \frac{1}{2\sigma^2} \sum_{i=1}^{n} (\mathbf{w}^T \mathbf{x}_i - \mathbf{y}_i)^2
\end{aligned}$$

To find the maximum likelihood estimate $\mathbf{w}_{\text{ML}}$, we want to maximize $\log \mathcal{L}(\mathbf{w}; \mathbf{X}, \mathbf{y})$. Maximizing the likelihood is equivalent to minimizing the negative log-likelihood. Thus, we seek to minimize:

$$-\log \mathcal{L}(\mathbf{w}; \mathbf{X}, \mathbf{y}) = \frac{1}{2\sigma^2} \sum_{i=1}^{n} (\mathbf{w}^T \mathbf{x}_i - \mathbf{y}_i)^2$$

This expression is equivalent to the mean squared error (MSE) objective function used in linear regression. Therefore, finding the maximum likelihood estimate $\mathbf{w}_{\text{ML}}$ is equivalent to solving the linear regression problem using the squared error loss.

To obtain the closed-form solution for $\mathbf{w}_{\text{ML}}$, we differentiate the negative log-likelihood with respect to $\mathbf{w}$ and set the derivative to zero:

$$\nabla_{\mathbf{w}} \left(-\log \mathcal{L}(\mathbf{w}; \mathbf{X}, \mathbf{y})\right) = \frac{1}{\sigma^2} \sum_{i=1}^{n} (\mathbf{w}^T \mathbf{x}_i - \mathbf{y}_i) \mathbf{x}_i^T = \mathbf{0}$$

This can be rewritten as:

$$\frac{1}{\sigma^2} \left(\mathbf{X}\mathbf{X}^T \mathbf{w} - \mathbf{X}\mathbf{y}\right) = \mathbf{0}$$

where $\mathbf{X}$ is the matrix whose rows are the input vectors $\mathbf{x}_i$ and $\mathbf{y}$ is the column vector of labels. Rearranging the equation, we have:

$$\mathbf{X}\mathbf{X}^T \mathbf{w} = \mathbf{X}\mathbf{y}$$

To obtain the closed-form solution for $\mathbf{w}_{\text{ML}}$, we multiply both sides by the inverse of $\mathbf{X}\mathbf{X}^T$, denoted as $(\mathbf{X}\mathbf{X}^T)^{-1}$:

$$\mathbf{w}_{\text{ML}} = (\mathbf{X}\mathbf{X}^T)^{-1} \mathbf{X}\mathbf{y}$$

Thus, the closed-form solution for the maximum likelihood estimate $\mathbf{w}_{\text{ML}}$ is given by the product of $(\mathbf{X}\mathbf{X}^T)^{-1}$ and $\mathbf{X}\mathbf{y}$.

The closed-form solution for $\mathbf{w}_{\text{ML}}$ in linear regression demonstrates that it can be obtained by directly applying a matrix inverse operation to the product of the input matrix $\mathbf{X}$ and the target variable vector $\mathbf{y}$. This closed-form solution provides an efficient and direct way to estimate the weight vector $\mathbf{w}$ based on the given data.

Acknowledgments

Professor Arun Rajkumar: The content, including the concepts and notations presented in this document, has been sourced from his slides and lectures. His expertise and educational materials have greatly contributed to the development of this document.

ChatGPT: The AI language model used in this document has made corrections and improvements to the notations and language, ensuring clarity and accuracy in the presentation of the material. Its capabilities have enhanced the overall quality and readability of this document.