Machine Learning Notes

Amit Thakur

January 3, 2025

Contents

Chapter 1

Linear Regression

Linear Regression is a statistical method used for modeling the relationship between a dependent variable (target or output) and one or more independent variables (predictors or features).

1.1 Linear Regression with Single Feature

This regression deals with one independent variable (x).

1.1.1 Linear Regression Model

$$y = \theta_0^t + \theta_1^t x + \epsilon \tag{1.1}$$

$$\hat{y} = h_\theta(x) = \theta_0 + \theta_1 x \tag{1.2}$$

where:

- y: the dependent variable (target)
- x: the independent variable or input feature used to predict y.
- θ_1^t: the slope or coefficient of x at iteration t
- θ_0^t: intercept at iteration t
- ϵ: the error term or residual. It captures the noise or other unmodeled effects.
- $\hat{y}$: the output of the linear regression model (h_θ) for a given input (x)
- $h_\theta(x)$: the hypothesis function for linear regression.

1.1.2 Cost Function

The cost function for linear regression measures the average squared error between predicted and actual values. It is defined as:

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(\hat{y}^{(i)} - y^{(i)}\right)^2 = \frac{1}{2m} \sum_{i=1}^{m} \left((\theta_0 + \theta_1 x^{(i)}) - y^{(i)}\right)^2 \tag{1.3}$$

where:

- m: Number of input and output datapoints.
- $\frac{1}{2}$: A factor kept for convenience, as it simplifies the derivative calculations during gradient descent.

1.1.3 Minimizing Cost

The error for each data point can be written as:

$$e^{(i)} = y^{(i)} - (\theta_0 + \theta_1 x^{(i)}) \tag{1.4}$$

The cost function becomes:

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} e^{(i)2} \tag{1.5}$$

We need to find θ_0 and θ_1 that minimize J. This requires setting the partial derivatives of J with respect to θ_0 and θ_1 to zero as the cost function is quadratic and there's just one critical point.

Solving for θ_0:

$$\frac{\partial J}{\partial \theta_0} = -\frac{1}{m} \sum_{i=1}^{m} \left(y^{(i)} - (\theta_0 + \theta_1 x^{(i)})\right) = 0$$

$$\sum_{i=1}^{m} \left(y^{(i)} - (\theta_0 + \theta_1 x^{(i)})\right) = 0$$

$$\sum_{i=1}^{m} y^{(i)} = m\theta_0 + \theta_1 \sum_{i=1}^{m} x^{(i)}$$

Divide through by m:

$$\bar{y} = \theta_0 + \theta_1 \bar{x}$$

where $\bar{y}$ and $\bar{x}$ are the means of $y^{(i)}$ and $x^{(i)}$, respectively.
Rearranging:

$$\theta_0 = \bar{y} - \theta_1 \bar{x} \tag{1.6}$$

Solving for θ_1:

$$\frac{\partial J}{\partial \theta_1} = -\frac{1}{m} \sum_{i=1}^{m} x^{(i)} \left(y^{(i)} - (\theta_0 + \theta_1 x^{(i)})\right) = 0$$

Expanding:

$$\sum_{i=1}^{m} x^{(i)} y^{(i)} = \theta_0 \sum_{i=1}^{m} x^{(i)} + \theta_1 \sum_{i=1}^{m} \left(x^{(i)}\right)^2$$

Substitute $\theta_0 = \bar{y} - \theta_1 \bar{x}$:

$$\sum_{i=1}^{m} x^{(i)} y^{(i)} = (\bar{y} - \theta_1 \bar{x}) \sum_{i=1}^{m} x^{(i)} + \theta_1 \sum_{i=1}^{m} (x^{(i)})^2$$

Simplify:

$$\sum_{i=1}^{m} x^{(i)} y^{(i)} = \bar{y} \sum_{i=1}^{m} x^{(i)} - \theta_1 \bar{x} \sum_{i=1}^{m} x^{(i)} + \theta_1 \sum_{i=1}^{m} (x^{(i)})^2$$

Reorganize terms:

$$\theta_1 \left(\sum_{i=1}^{m} (x^{(i)})^2 - \frac{1}{m} \left(\sum_{i=1}^{m} x^{(i)} \right)^2 \right) = \sum_{i=1}^{m} x^{(i)} y^{(i)} - \frac{1}{m} \sum_{i=1}^{m} x^{(i)} \sum_{i=1}^{m} y^{(i)}$$

Using simplified notation:

- $\bar{x} = \frac{1}{m} \sum_{i=1}^{m} x^{(i)}$ (mean of x)
- $\bar{y} = \frac{1}{m} \sum_{i=1}^{m} y^{(i)}$ (mean of y)

$$\theta_1 = \frac{\sum_{i=1}^{m} \left(x^{(i)} - \bar{x}\right) \left(y^{(i)} - \bar{y}\right)}{\sum_{i=1}^{m} \left(x^{(i)} - \bar{x}\right)^2} \tag{1.7}$$

1.1.4 Gradient Descent Algorithm to find optimal θ_0 and θ_1

1. Gradient descent moves the parameters in the direction of the negative gradient (steepest descent) of the cost function.
2. The learning rate α determines the size of each step. If α is too large, the algorithm may overshoot the minimum. If it is too small, convergence may take too long.
3. The iterative process ensures gradual improvement in the model parameters until the cost function is minimized.

Algorithm 1 Gradient Descent for Linear Regression

Input: Learning rate α, initial values for θ_0 and θ_1, and maximum iterations or convergence threshold ϵ.
Output: Optimized parameters θ_0 and θ_1.
Set initial values for θ_0 and θ_1.
repeat
    Compute updates for parameters:

$$\text{temp0} := \theta_0 - \alpha \cdot \frac{1}{m} \sum_{i=1}^{m} \left(\theta_0 + \theta_1 x^{(i)} - y^{(i)} \right)$$
$$\text{temp1} := \theta_1 - \alpha \cdot \frac{1}{m} \sum_{i=1}^{m} \left(\theta_0 + \theta_1 x^{(i)} - y^{(i)} \right) x^{(i)}$$

    Update parameters simultaneously:

$$\theta_0 := \text{temp0}$$
$$\theta_1 := \text{temp1}$$

    Compute cost function:

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(\theta_0 + \theta_1 x^{(i)} - y^{(i)} \right)^2$$

until maximum iterations reached **or** change in $J(\theta_0, \theta_1)$ is below threshold ϵ.
return θ_0, θ_1

1.2 Linear Regression with Multiple Features

$$y = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n + e \tag{1.8}$$

where

- y: The dependent variable (or target/output variable) that the model predicts
- θ_0: The intercept term, representing the value of y when all $x_i = 0$
- $\theta_1, \theta_2, \ldots, \theta_n$: The coefficients or weights for the independent variables $x_1, x_2, \ldots, x_n$.

1.2.1 General Form

$$y = \boldsymbol{\theta}^T \mathbf{x} + e \tag{1.9}$$

where $\boldsymbol{\theta} \in \mathbb{R}^{(n+1)\times 1}$ is the parameter vector (column vector of coefficients), defined as:

$$\boldsymbol{\theta} = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_n \end{bmatrix} \tag{1.10}$$

And the input feature vector $\mathbf{x} \in \mathbb{R}^{(n+1)\times 1}$ defined as:

$$\mathbf{x} = \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_n \end{bmatrix} \tag{1.11}$$

The prediction function is:

$$\hat{y} = h_\theta(\mathbf{x}) = \boldsymbol{\theta}^T \mathbf{x} \tag{1.12}$$

1.2.2 Cost Function

Let $X \in \mathbb{R}^{m\times(n+1)}$ be all the data consisting of m examples:

$$\text{Let } X = \begin{bmatrix} 1 & x^{(1)} & (x^{(1)})^2 & (x^{(1)})^3 & \cdots & (x^{(1)})^n \\ 1 & x^{(2)} & (x^{(2)})^2 & (x^{(2)})^3 & \cdots & (x^{(2)})^n \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x^{(m)} & (x^{(m)})^2 & (x^{(m)})^3 & \cdots & (x^{(m)})^n \end{bmatrix} \tag{1.13}$$

and the target vector $y \in \mathbb{R}^m$:

$$\mathbf{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{bmatrix} \tag{1.14}$$

and the parameter vector $\boldsymbol{\theta} \in \mathbb{R}^{n+1}$:

$$\boldsymbol{\theta} = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_n \end{bmatrix} \tag{1.15}$$

The residuals ($\mathbf{E} \in \mathbb{R}^m$) represent the difference between the actual target values ($\mathbf{y}$) and the predicted values ($X\boldsymbol{\theta}$). For m data points:

$$\mathbf{E} = \begin{bmatrix} e^{(1)} \\ e^{(2)} \\ \vdots \\ e^{(m)} \end{bmatrix} = X\boldsymbol{\theta} - y \tag{1.16}$$

The cost function $J(\boldsymbol{\theta})$ is:

$$J(\boldsymbol{\theta}) = \frac{1}{2m} \sum_{i=1}^{m} \left(e^{(i)}\right)^2 \tag{1.17}$$

Using $E = X\boldsymbol{\theta} - \mathbf{y}$, we can write:

$$J(\boldsymbol{\theta}) = \frac{1}{2m} \|\mathbf{E}\|^2$$

Expanding this using the transpose:

$$J(\boldsymbol{\theta}) = \frac{1}{2m}\mathbf{E}^T\mathbf{E}$$

Substituting $\mathbf{E} = X\boldsymbol{\theta} - y$:

$$J(\boldsymbol{\theta}) = \frac{1}{2m}(X\boldsymbol{\theta} - \mathbf{y})^T(X\boldsymbol{\theta} - \mathbf{y}) \tag{1.18}$$

To minimize $J(\boldsymbol{\theta})$, we compute the gradient (partial derivative) with respect to each parameter θ_j:

$$\frac{\partial J(\boldsymbol{\theta})}{\partial \theta_j} = \frac{\partial}{\partial \theta_j}\left[\frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2\right].$$

Using the chain rule:

$$\frac{\partial J(\boldsymbol{\theta})}{\partial \theta_j} = \frac{1}{2m}\sum_{i=1}^{m} 2\left(h_{\boldsymbol{\theta}}(x^{(i)}) - y^{(i)}\right)\frac{\partial}{\partial \theta_j}\left(h_\theta(x^{(i)}) - y^{(i)}\right).$$

We take the partial derivative of $h_\theta(x^{(i)})$ with respect to θ_j:

$$\frac{\partial}{\partial \theta_j}h_\theta(x^{(i)}) = \frac{\partial}{\partial \theta_j}\sum_{j=0}^{n}\theta_j x_j^{(i)}$$

Since differentiation is linear, we apply the derivative term-by-term:

$$\sum_{k=0}^{n}\frac{\partial}{\partial \theta_j}(\theta_k x_k^{(i)})$$

- When $k = j$, the term is $\theta_j x_j^{(i)}$, and its derivative with respect to θ_j is $x_j^{(i)}$.
- When $k \neq j$, θ_k is independent of θ_j, so its derivative is 0.

Thus, we are left with:

$$\frac{\partial}{\partial \theta_j}h_\theta(x^{(i)}) = x_j^{(i)}.$$

Finally:

$$\frac{\partial J(\theta)}{\partial \theta_j} = \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)x_j^{(i)}. \tag{1.19}$$

Using the property of transposes $(A - B)^T = A^T - B^T$, expand:

$$(X\boldsymbol{\theta} - \mathbf{y})^T(X\boldsymbol{\theta} - \mathbf{y}) = (X\boldsymbol{\theta})^T(X\boldsymbol{\theta}) - (X\boldsymbol{\theta})^T\mathbf{y} - \mathbf{y}^T(X\boldsymbol{\theta}) + \mathbf{y}^T\mathbf{y}$$

Since these are scalar terms, they are equal (by commutative property of dot product):

$$-(X\boldsymbol{\theta})^T\mathbf{y} = -\mathbf{y}^T(X\boldsymbol{\theta})$$

Substituting the simplified terms back, we get:

$$J(\boldsymbol{\theta}) = \frac{1}{2m}\left(\boldsymbol{\theta}^T X^T X\boldsymbol{\theta} - 2\mathbf{y}^T X\theta + \mathbf{y}^T\mathbf{y}\right)$$

Compute the derivative with respect to $\boldsymbol{\theta}$ (matrix calculus):

- Derivative of $\boldsymbol{\theta}^T X^T X\boldsymbol{\theta}$:

$$\frac{\partial}{\partial\boldsymbol{\theta}}\left(\boldsymbol{\theta}^T X^T X\boldsymbol{\theta}\right) = 2X^T X\boldsymbol{\theta}$$

- Derivative of $-2y^T X\boldsymbol{\theta}$:

$$\frac{\partial}{\partial\boldsymbol{\theta}}\left(-2y^T X\boldsymbol{\theta}\right) = -2X^T y$$

- Derivative of $\mathbf{y}^T\mathbf{y}$:

$$\frac{\partial}{\partial\boldsymbol{\theta}}\left(\mathbf{y}^T\mathbf{y}\right) = 0$$

Substituting above values, we get:

$$\frac{\partial J(\boldsymbol{\theta})}{\partial\boldsymbol{\theta}} = \frac{1}{m}\left(X^T X\boldsymbol{\theta} - X^T\mathbf{y}\right) \tag{1.20}$$

From the general gradient, we can isolate the derivative with respect to a single parameter θ_j. The gradient is:

$$\frac{\partial J(\boldsymbol{\theta})}{\partial\boldsymbol{\theta}} = \frac{1}{m}X^T(X\boldsymbol{\theta} - \mathbf{y})$$

The j-th element of the gradient corresponds to:

$$\frac{\partial J(\boldsymbol{\theta})}{\partial\theta_j} = \frac{1}{m}(X_{\text{column},j})^T(X\boldsymbol{\theta} - \mathbf{y})$$

1.2.3 Gradient of the Cost Function

The gradient of the cost function with respect to all parameters $\boldsymbol{\theta}$ is a vector:

$$\text{grad} = \begin{bmatrix} \frac{\partial J(\boldsymbol{\theta})}{\partial\theta_0} \\ \frac{\partial J(\boldsymbol{\theta})}{\partial\theta_1} \\ \vdots \\ \frac{\partial J(\boldsymbol{\theta})}{\partial\theta_n} \end{bmatrix} \tag{1.21}$$

$$\text{grad} = \frac{1}{m}X^T(X\boldsymbol{\theta} - \mathbf{y}) \tag{1.22}$$

1.2.4 Analytical Solution

To find the optimal $\boldsymbol{\theta}$, set the gradient grad to zero:

$$\frac{1}{m}X^T(X\boldsymbol{\theta} - \mathbf{y}) = 0$$

Simplify:

$$X^T(X\boldsymbol{\theta}) = X^T\mathbf{y}$$

Rearranging:

$$\boldsymbol{\theta} = (X^TX)^{-1}X^T\mathbf{y} \tag{1.23}$$

This is known as the normal equation, which gives the closed-form solution for $\boldsymbol{\theta}$ in linear regression.

1.2.5 Gradient Descent Algorithm

Algorithm 2 Gradient Descent for Linear Regression for multiple features

1: **Input:** Learning rate α, initial values for $\boldsymbol{\theta}$, and maximum iterations or convergence threshold ϵ.
2: **Output:** Optimized parameter vector, $\boldsymbol{\theta}$.
3: **repeat**
4: Compute gradient vector: $\mathbf{grad} \in \mathbb{R}^{(n+1)\times 1}$:

$$\mathbf{grad} = X^T(X \cdot \boldsymbol{\theta} - \mathbf{y})$$

5: Update the parameter vector, $\boldsymbol{\theta}$:

$$\boldsymbol{\theta} := \boldsymbol{\theta} - \alpha \cdot \mathbf{grad}$$

6: Compute cost function (just for monitoring):

$$J(\boldsymbol{\theta}) = \frac{1}{2m}(X\boldsymbol{\theta} - \mathbf{y})^T(X\boldsymbol{\theta} - \mathbf{y})$$

7: **until** maximum iterations reached **or** change in $J(\boldsymbol{\theta})$ is below threshold ϵ.
8: **return** $\boldsymbol{\theta}$

1.3 Linear Models for Regression

1.3.1 Polynomial Curve Fitting

$$\hat{y}(x, \boldsymbol{\theta}) = \theta_0 + \theta_1 x + \theta_2 x^2 + \cdots + \theta_n x^n \tag{1.24}$$

$$\text{Let } X = \begin{bmatrix} 1 & x^{(1)} & (x^{(1)})^2 & (x^{(1)})^3 & \cdots & (x^{(1)})^n \\ 1 & x^{(2)} & (x^{(2)})^2 & (x^{(2)})^3 & \cdots & (x^{(2)})^n \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x^{(m)} & (x^{(m)})^2 & (x^{(m)})^3 & \cdots & (x^{(m)})^n \end{bmatrix} \tag{1.25}$$

and the target vector:

$$\mathbf{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{bmatrix} \tag{1.26}$$

1.3.2 Analytical Solution

Just like previous muti-feature solution:

$$\boldsymbol{\theta}^* = (X^T X)^{-1} X^T \mathbf{y} \tag{1.27}$$

1.3.3 Linear Models with Basis Functions

$$\hat{y} = h_\theta(\mathbf{x}) = \boldsymbol{\theta}^T \boldsymbol{\Phi}(\mathbf{x}) = \sum_{j=0}^{n} \theta_j \phi_j(\mathbf{x}) \tag{1.28}$$

where $\boldsymbol{\Phi}(\mathbf{x})$ is a matrix of basis functions:

$$\boldsymbol{\Phi} = \begin{bmatrix} \phi_0(x^{(1)}) & \phi_1(x^{(1)}) & \phi_2(x^{(1)}) & \cdots & \phi_n(x^{(1)}) \\ \phi_0(x^{(2)}) & \phi_1(x^{(2)}) & \phi_2(x^{(2)}) & \cdots & \phi_n(x^{(2)}) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \phi_0(x^{(m)}) & \phi_1(x^{(m)}) & \phi_2(x^{(m)}) & \cdots & \phi_n(x^{(m)}) \end{bmatrix} \tag{1.29}$$

The cost function is:

$$J(\boldsymbol{\theta}) = \frac{1}{2m} \sum_{i=1}^{m} \left(e^{(i)}\right)^2$$

Using the matrix form, the residuals for all data points can be written as $E = \Phi \cdot \boldsymbol{\theta} - y$, and the cost function becomes:

$$J(\boldsymbol{\theta}) = \frac{1}{2m} \mathbf{E}^T \mathbf{E} = \frac{1}{2m} (\boldsymbol{\Phi} \cdot \boldsymbol{\theta} - \mathbf{y})^T (\boldsymbol{\Phi} \cdot \boldsymbol{\theta} - \mathbf{y})$$

The gradient of $J(\theta)$ with respect to the parameters $\boldsymbol{\theta}$ is:

$$\text{grad} = \frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \begin{bmatrix} \frac{\partial J(\boldsymbol{\theta})}{\partial \theta_0} \\ \frac{\partial J(\boldsymbol{\theta})}{\partial \theta_1} \\ \vdots \\ \frac{\partial J(\boldsymbol{\theta})}{\partial \theta_n} \end{bmatrix} = \frac{1}{m} \boldsymbol{\Phi}^T \cdot (\boldsymbol{\Phi} \cdot \boldsymbol{\theta} - \mathbf{y})$$

From the cost function:

$$J(\boldsymbol{\theta}) = \frac{1}{2m} (\boldsymbol{\Phi} \cdot \boldsymbol{\theta} - \mathbf{y})^T (\boldsymbol{\Phi} \cdot \boldsymbol{\theta} - \mathbf{y})$$

Taking the derivative with respect to $\boldsymbol{\theta}$:

$$\text{grad} = \frac{1}{m} \boldsymbol{\Phi}^T \cdot (\boldsymbol{\Phi} \cdot \boldsymbol{\theta} - \mathbf{y})$$

To find the optimal $\boldsymbol{\theta}^*$, we set the gradient to zero:

$$\frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = 0$$

This gives the normal equation:

$$\boldsymbol{\Phi}^T \cdot \boldsymbol{\Phi} \cdot \boldsymbol{\theta} = \boldsymbol{\Phi}^T \cdot \mathbf{y}$$

Rearranging:

$$\boldsymbol{\theta}^* = (\boldsymbol{\Phi}^T \cdot \boldsymbol{\Phi})^{-1} \cdot \boldsymbol{\Phi}^T \cdot \mathbf{y}$$

Gradient Descent Update Rule:

$$\boldsymbol{\theta} := \boldsymbol{\theta} - \alpha \cdot \text{grad}$$

Substituting the gradient:

$$\boldsymbol{\theta} := \boldsymbol{\theta} - \alpha \cdot \frac{1}{m} \boldsymbol{\Phi}^T \cdot (\boldsymbol{\Phi} \cdot \boldsymbol{\theta} - \mathbf{y})$$

Here the α is the learning rate, controlling the step size.

1.4 Probabilistic Interpretation of Linear Regression

Treating the target variable y as a random variable conditioned on the input features $\mathbf{x}$, with the assumption that the observed y values are drawn from a probability distribution centered around the model's prediction.

1.4.1 Equivalence of least square error and maximum likelihood estimation

$$y = \boldsymbol{\theta}^T \mathbf{x} + e$$

Here we have:

- y: The observed output (dependent variable or target)
- $\boldsymbol{\theta}$: is the parameter vector
- $\mathbf{x}$: is the feature/input vector
- e: The random noise term, representing unmodeled or unexplained effects

For i^{th} data point:

$$y^{(i)} = \boldsymbol{\theta}^T \mathbf{x}^{(i)} + e^{(i)}$$

Model Assumptions:

- The relationship between the features ($\mathbf{x}$) and the target (y) is linear:

  $$\mu_y = h_\theta(\mathbf{x}) = \boldsymbol{\theta}^T \mathbf{x},$$

  where μ_y is the mean of y, predicted by the linear model.

- Observations y deviate from the linear prediction due to noise or unmodeled effects, captured by an additive random error term e:

$$y = \boldsymbol{\theta}^T \mathbf{x} + e.$$

The noise e is assumed to follow a Gaussian distribution:

$$e \sim \mathcal{N}(0, \sigma^2),$$

where σ^2 is the variance of the noise.

- Given $\mathbf{x}$, y is conditionally normally distributed:

$$y \mid \mathbf{x} \sim \mathcal{N}(\boldsymbol{\theta}^T \mathbf{x}, \sigma^2).$$

Probabilistic Model:
The probability density function (PDF) of y given $\mathbf{x}$ is:

$$p(y \mid \mathbf{x}, \boldsymbol{\theta}, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y - \boldsymbol{\theta}^T \mathbf{x})^2}{2\sigma^2}\right). \tag{1.30}$$

This represents the likelihood of observing a particular y value for a given input $\mathbf{x}$. Now we can estimate the parameters $\boldsymbol{\theta}$ and σ^2 by maximizing the likelihood of the observed data. For a dataset $\{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^m$, the likelihood is the joint probability of all m observations:

$$\mathcal{L}(\boldsymbol{\theta}, \sigma^2) = \prod_{i=1}^m p(y^{(i)} \mid \mathbf{x}^{(i)}, \boldsymbol{\theta}, \sigma^2) = \prod_{i=1}^m \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{\left(y^{(i)} - h_\theta\left(x^{(i)}\right)\right)^2}{2\sigma^2}\right) \tag{1.31}$$

Taking ln both sides:

$$\ell(\boldsymbol{\theta}, \sigma^2) = \ln \mathcal{L}(\boldsymbol{\theta}, \sigma^2) = -\frac{m}{2}\ln(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_{i=1}^m \left(y^{(i)} - h_\theta\left(\mathbf{x}^{(i)}\right)\right)^2 \tag{1.32}$$

Here, maximize the log-likelihood with respect to $\boldsymbol{\theta}$ is equivalent to minimizing the sum of squared errors, under the assumption of Gaussian distribution for the error:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^m \left(y^{(i)} - \boldsymbol{\theta}^T \mathbf{x}^{(i)}\right)^2. \tag{1.33}$$

This leads to the familiar solution of linear regression (normal equation):

$$\boldsymbol{\theta}^* = (\Phi^T \Phi)^{-1} \Phi^T \mathbf{y} \tag{1.34}$$

Thus, the optimal value of σ^2 can be calculated as:

$$\sigma_{ML}^2 = \frac{1}{m}\sum_{i=1}^m \left(y^{(i)} - h_{\theta_{ML}}\left(\mathbf{x}^{(i)}\right)\right)^2 \tag{1.35}$$

which is equivalent to minimized cost function.
For any new input $\mathbf{x}$, the predicted value $\hat{y}$ is computed using $h_{\theta_{ML}}(\mathbf{x})$.

Bibliography

[1] Weidong Kuang. *Fundamentals of deep learning: a step-by-step guide.* Self, 2024. https://faculty.utrgv.edu/weidong.kuang/book/book.html(visited 2024-12-01).

[2] Simon J.D. Prince. *Understanding Deep Learning*. The MIT Press, 2023.

[3] Amit Thakur. *Maths for Machine Learning Notes*. Self, 2025. http://amitthakur.org/learnings/maths/maths-for-ml.