Linear Regression: Training and Cost Functions

Recap: The Linear Regression Model Equation

As introduced previously, a Linear Regression model makes predictions by calculating a weighted sum of the input features, plus a constant term known as the bias or intercept. The general equation is:

**ŷ = θ₀ + θ₁x₁ + θ₂x₂ + ... + θ<0xE2><0x82><0x99>x<0xE2><0x82><0x99>** <<< Linear Regression Equation

Where:

* ŷ (y-hat) is the **predicted value**.
* n is the number of **features** (independent variables).
* xᵢ is the value of the **i-th feature**.
* θⱼ (theta-j) represents the **j-th model parameter**. This includes:
  + θ₀: The bias term (intercept).
  + θ₁, θ₂, ..., θ<0xE2><0x82><0x99>: The feature weights (coefficients or slopes).

*Example:* A learned model might look like y = 3.0\*x₁ + 4.0\*x₂ + 2.0\*x₃ + 8.0\*x₄ + 6.0\*x₅ + 9.7 (where 9.7 is θ₀).

The Learning Process: Finding the Best Parameters

How does the model determine the optimal values for these parameters (θ₀, θ₁, ..., θ<0xE2><0x82><0x99>)? This is where the "learning" happens.

1. **Data:** We start with a dataset containing known input features (X) and their corresponding actual output values (y).
2. **Learning Algorithm:** A specific algorithm (like Gradient Descent, which we'll discuss later, or analytical methods like the Normal Equation) is used.
3. **Objective:** The learning algorithm iteratively adjusts the parameter values (θs) based on the training data. Its goal is to find the set of parameters that minimizes the difference between the model's predictions (ŷ) and the actual values (y) across the entire training dataset.
4. **Model:** The output of the learning process is the **trained model** – essentially the Linear Regression equation ŷ = f(X) with the specific, optimal parameter values found by the algorithm.

*(Consider inserting the Data -> Learning Algorithm -> Model diagram here)*

**In essence: The Learning Algorithm *learns* from the Data to find the optimum values of the parameters (θ) that define the best-fitting model.**

Cost Functions: Measuring Model Error

To find the "optimum" or "best-fitting" parameters, the learning algorithm needs a way to quantify how well the model is performing with the *current* set of parameters. This is the role of the **Cost Function** (also known as a Loss Function or Objective Function).

* **Purpose:** A cost function measures the **aggregate error** or difference between the actual values (y) and the predicted values (ŷ) across all data points in the training set.
* **Goal:** The learning algorithm aims to **minimize** the value of the cost function. The lower the cost, the better the model fits the training data.

While the specific cost function calculation for Linear Regression models can vary slightly, they are all fundamentally based on the differences between the actual and predicted values.

Residuals: The Basis of Error Calculation

The difference between the actual value (yᵢ) and the predicted value (ŷᵢ) for a single data point i is called the **residual** or error.

**Residualᵢ = yᵢ - ŷᵢ**

*(Consider inserting the residuals visualization graph here)*

The residuals represent the vertical distance between each actual data point and the predicted regression line (the best fit line determined by the current parameters).

Common Cost Functions / Error Metrics for Linear Regression

These metrics aggregate the individual residuals across the dataset:

1. **SSR or Sum of Squared Residuals (also RSS - Residual Sum of Squares):**
   * Calculates the sum of the squares of all the residuals.
   * **Formula:** SSR = Σ [ (yᵢ - ŷᵢ)² ] (sum over all data points i=1 to n)
   * Squaring ensures errors don't cancel out and penalizes larger errors more heavily.
2. **MSE or Mean Squared Error:**
   * Calculates the average of the squared residuals. It's simply the SSR divided by the number of data points (n).
   * **Formula:** MSE = (1/n) \* Σ [ (yᵢ - ŷᵢ)² ] = SSR / n
   * This is the most common cost function minimized during the training of standard Linear Regression models.
3. **RMSE or Root Mean Squared Error:**
   * Calculates the square root of the MSE.
   * **Formula:** RMSE = sqrt(MSE) = sqrt [ (1/n) \* Σ [ (yᵢ - ŷᵢ)² ] ]
   * Often used as a final evaluation metric because its units are the same as the target variable (y), making it more interpretable than MSE.

Evaluating the Fit: Coefficient of Determination (R²)

While MSE or RMSE are typically minimized *during training*, the **Coefficient of Determination (R²)** is a standard metric used to *evaluate* how well the final, trained Linear Regression model fits the data and explains the variance in the target variable.

* **Definition:** R² measures the proportion of the total variance in the actual values (y) that is explained by the model.
* **Formula:**
* R² = 1 - (SSR / TSS)

Where:

* + SSR (or RSS) is the Sum of Squared Residuals (error of the model), as defined above: Σ(yᵢ - ŷᵢ)²
  + TSS is the Total Sum of Squares, representing the total variance in y: Σ(yᵢ - ȳ)², where ȳ is the mean of the actual y values.
* **Interpretation:**
  + R² values range from 0 to 1 (or can be negative for very poor fits).
  + **R² = 1:** Perfect fit. The model explains 100% of the variance in y. All data points lie exactly on the regression line.
  + **R² = 0:** The model explains none of the variance. It performs no better than simply predicting the mean (ȳ) for all observations.
  + **Higher R² values (closer to 1) generally indicate a better fit**, meaning the model's predictions are closer to the actual values relative to the overall variability of the data.

Physical Significance of R²

Visualizing R² helps understand its meaning:

* **R² = 1:** Data points form a perfect line.
* **R² = 0.70:** Data points are reasonably close to the regression line; a clear linear trend exists.
* **R² = 0.36:** Data points are more scattered, but a slight linear trend might still be visible.
* **R² = 0.05:** Data points are widely scattered; the regression line explains very little of the variation.

In summary, cost functions like MSE quantify the model's error during training, guiding the learning algorithm to find the best parameters. Metrics like R² help evaluate the goodness-of-fit of the final trained model.