**Simple Linear Regression Theory**

**Regression:** used to find the relationship between variables.

**Linear Relation:** A linearrelation means that the change in one variable is directly proportional (or almost proportional) to the change in another variable.

**Linear Regression:** Used to find the relationship between 2 variables.

**Equation:** Y=β0 +β1X+ε

**Y:** Dependent variable

**X:** Independent variable

**Β1 (m):** Slope of the line (how much y changes for unit changes in x)

**Β0 (b):** Intercept (value of y when x is zero)

**ε:** Error Term

**Formula for β1:** sum of product of deviations/ sum of square of deviations for x

**Formula for β2:** Mean of Y - (β1 \* mean of β1)

### **Assumptions of Linear Regression**

* Linear relationship between X and Y
* Residuals (errors) are normally distributed
* Homoscedasticity (equal variance of residuals)
* Independence of observations
* No/mild multicollinearity (Means predictors should be either not correlated or only weakly correlated with each other, not relevant for simple regression, but mention briefly)

### **How the Model Works**

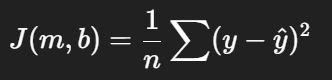
* Goal: Find best fit line that minimizes error.
* **Ordinary Least Squares (OLS):** Ordinary Least Squares (OLS) is a method that finds the best-fit line by minimizing the total squared differences between actual and predicted values.

### **In Machine Learning**

* It’s the **foundation for regression problems**.
* Provides interpretability (easy to explain how X affects Y).
* Forms the base for more advanced algorithms like **Multiple Linear Regression, Logistic Regression, and Neural Networks**.

**Cost Function & Gradient Descent**

**Cost Function (aka Loss Function):** In linear regression, the cost function tells us how far off our predictions are from the actual values.

* **Formula For Cost Function:** 
* Here, the first y value is the actual y value while the second is the y predicted value
* Cost should be reduced.
* If predictions are close to actual values → cost is low.
* If predictions are far → cost is high.

**Gradient descent:** algorithm that finds the best fit line for the given training data set.

* Start with random values of slope *Β1 (m)* and intercept *Β0 (b)*.
* Compute the gradient (slope of the cost function).
* Update the parameters step by step in the opposite direction of the gradient (downhill).
* ***Β1 (m):*** *m – learning\_rate\*m\_derivative*
* ***Β0 (b):*** *b – learning\_rate\*b\_derivative*
* ***B\_derviative:*** -(2/n)\*sum(y-y\_predicted)
* ***m\_derviative:*** -(2/n)\*sum(x\* (y-y\_predicted))

**Key Intuition:**

* Think of cost function as a bowl-shaped curve.
* Gradient Descent is like a ball rolling down the bowl until it reaches the bottom (minimum cost).
* The bottom point corresponds to the best values of β₀ and β₁ → best-fit line.

**Key Takeaways:**

* Cost Function measures prediction error.
* Gradient Descent minimizes that error step by step.
* Learning rate (a) is crucial:
* Too small → very slow.
* Too large → may overshoot and never converge.

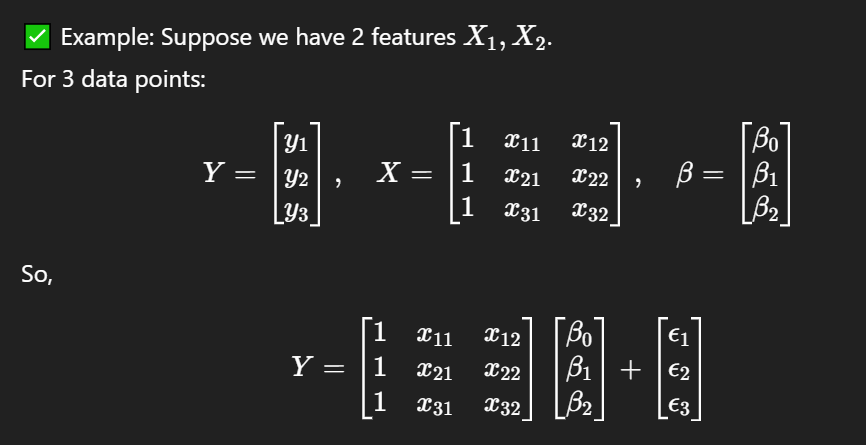
**Multiple Linear Regression**

**Regression:** used to find the relationship between variables.

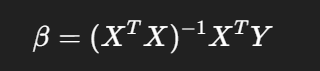
**Multiple Linear Regression:** This relation will have multiple independent variables and 1 dependent variables.

**Formula:** *Y=β0 +β1 X1 +β2 X2 +⋯+βn Xn +ϵ*

**Finding values of coefficients:**



**Formula to find coefficients:**



**Why do we use it?**

* To predict outcomes using multiple factors.
* To understand relationships between variables.
* To identify which variables, matter most.

**Polynomial Regression**

**What is Polynomial Regression:**

* It’s still linear regression at the core, but instead of fitting a straight line, we fit a curved line by adding polynomial terms (squared, cubic, etc.) of the features.
* Example: Instead of just

*y=β0+β1x*

we can fit:

*y=β0+β1x+β2x2+β3x3+…*

**Why do we use Polynomial Regression:**

* Simple linear regression can only fit straight lines.
* Polynomial regression captures **non-linear relationships** between the independent variable(s) and dependent variable.
* If we add higher powers of *x*, like *x2, x3*, the regression line can bend and curve:
* With *x2x^2*x2: parabola (U-shape or inverted U).
* With *x3x^3*x3: S-shaped curve.
* With more degrees: even more complex curves.

**Formula:**



**When it is useful:**

* Predicting house prices (sometimes price rises quickly with area, then slows down).
* Modelling physics problems (e.g., distance vs. time under acceleration).
* Capturing trends that clearly aren’t straight lines.

**Regularization**

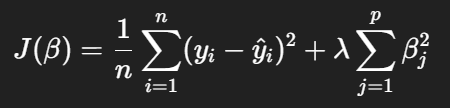
**Regularization:**

Regularization is a technique in machine learning used to prevent overfitting by adding a penalty to the cost (loss) function.

**Ridge Regression (L2 Regularization):**

* Adds a penalty proportional to the square of the coefficients.

**Cost function:**

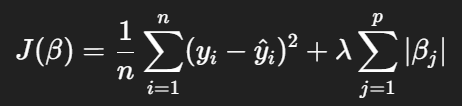


* Shrinks coefficients but never makes them exactly zero.
* Useful when many features contribute a little bit.

**Lasso Regression (L1 Regularization):**

* Adds a penalty proportional to the absolute value of coefficients.

**Cost function:**



* Can shrink some coefficients exactly to zero → performs feature selection.