There are two main task in Supervised ML that are:

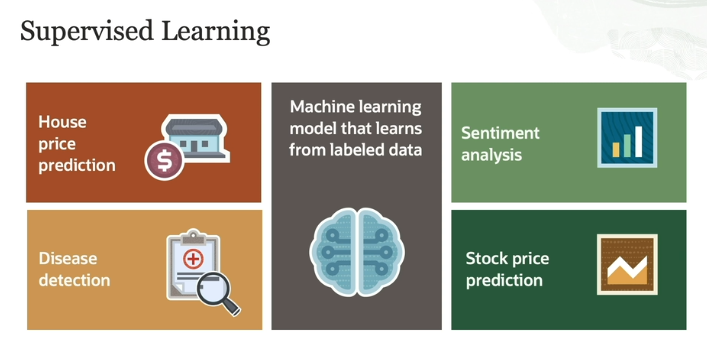
1. Classification
2. Prediction / Regression / Inference

**inference** refers to the process of using a trained model to make predictions on new, unseen data.

1- **Training Phase**: First, during training, the model learns from a set of labeled data. For example, if you're predicting house prices, the training data includes input features like house size, location, and number of rooms, along with the actual prices (the labeled data).

2- **Inference Phase**: After the model has been trained, it's ready to be used to predict outcomes for new data that it hasn’t seen before. This is what we call **inference**. Inference is the stage where the model is used to apply the learned patterns to make predictions.





The image is explaining **Supervised Learning**, a type of machine learning model that uses **labeled data** to train a model. In supervised learning, the model learns by mapping input features (data) to known output labels (results or predictions). The goal is to teach the model to predict outputs for new, unseen data by learning from past examples (iska mtlb yehi haka jo main goal hota hai wo yehi hota hai kay model ko pehlay exact input features and uskay corresponding output pay train krna and then on the basis of that past learned relationship and data , jo new input data aye usko classify krna and then predict krna). Let’s break it down with examples, as seen in the image:

**Key Idea:**

Supervised learning works by training the model on data where both the input and output are known (its mean kay during the process of training jo input data/feature with their corresponding output label provide kiya jatay hain wo exact relationship hotay hain which means that they are not unknown). The input is data the model is trained on, and the output is what we want the model to predict. After enough training, the model can predict the output for new, unseen inputs.

**Applications (in the image):**

1. **House Price Prediction:**
   * **Input**: House size in square feet, number of bedrooms, etc.
   * **Output**: The price of the house.
   * **Explanation**: The model is trained on historical data about house prices, learning how certain factors like square footage or location affect the price. Once trained, the model can predict the price of a house based on its features.
   * Now i have a question in this example that ab jo house ki price hoti hai wo toh different size , location etc pa depend krti hai toh uskay liya hum kasay ek exact output labeled data use kreinga ?

In your example of predicting house prices, you're right to point out that house prices depend on many factors, such as size, location, number of bedrooms, nearby amenities, and more. Here’s how this is handled in supervised learning:

1. **Input Features**: We don’t just use one input (like square feet). Instead, we create a set of **input features**, which can include the house size, location, number of bedrooms, age of the house, etc. Each of these features will have some influence on the final house price.
2. **Training Data**: We need labeled data where we already know the prices of houses with similar features. This data would include houses of different sizes, in different locations, and so on. The price for each house is the **output label** in the training data.
3. **Model Learning**: During training, the model learns how each feature affects the price. For instance, it might learn that houses in certain locations tend to be more expensive, or that a larger square footage adds a certain amount to the price. The model is essentially learning the relationships between the input features (size, location, etc.) and the output (price).
4. **Making Predictions**: Once trained, the model can use the learned relationships to predict the price of a new house, based on its features. It may not be exact (because many factors influence house prices), but it will give a reasonable estimate based on the patterns it has learned.

So iska mtlb yeh hua kay jab hum Model ko kuch different input features dedengay and unkay corresponding may output labels be dengay toh wo unkay relationships ko learn krega or then jab wo yeh training kr rha hoga toh wo yeh bi learn krega kay different relations may jo price fluctuation ho rhi hai wo kin features ki base pa ho rhi hai like size, location etc. So isi tarah say input features with their corresponding output labels ka through model ko train krsktay hain.

1. **Disease Detection (Medical Diagnosis):**
   * **Input**: A patient’s medical data, such as scans, age, or symptoms.
   * **Output**: Whether the person has a certain disease, like cancer, or whether the tumor is benign (non-cancerous) or malignant (cancerous).
   * **Explanation**: The model is trained using data from past patients, where it knows both their medical details and whether they had a disease. After training, the model can predict the likelihood of a disease in new patients.
   * Or iss example mabi yehi ho rha hoga kay first we will train model on different input disease features with their corresponding output label. Then it will learn the relation between them and predict the exact disease.
2. **Sentiment Analysis:**
   * **Input**: Customer reviews of products, text from social media, etc.
   * **Output**: Positive, negative, or neutral sentiment.
   * **Explanation**: The model is trained on a large set of reviews that are already labeled as positive, negative, or neutral. Based on the language patterns, the model learns to identify sentiment in new reviews.
   * Or yebi isi tarah possible hai kay by providing the reviews and language patterns as input features with their corresponding output label. Like:

|  |  |
| --- | --- |
| Input Feature | Output label |
| Good, excellent, or similar words | Positive |
| Bad, not good, or similar words | Negative |
| Ok , better, or similar words | Neutral |

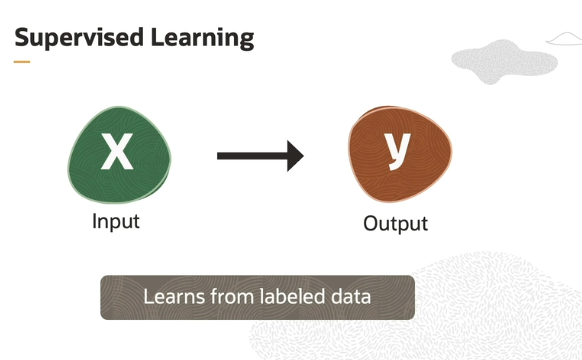
So now jo above ek table create kia hai wo zarori nhi kay isi tarah say input feature provide kr rhay hongay But just for imagining it , we can consider it. Agar jo review language pattern using positive words / sentence so its mean it will predict the output label as Positive.

1. **Stock Price Prediction:**
   * **Input**: Information such as the stock’s opening price, closing price, trading volume, etc.
   * **Output**: The future price of the stock.
   * **Explanation**: The model learns from past stock data how factors like price fluctuations and volume affect future prices. It then uses this knowledge to predict future prices based on current data.

**Summary of the Learning Process:**

* **Labeled Data**: The training data has both the input and the correct output (label). For example, house prices for given features, or stock prices based on historical data.
* **Training**: The model uses this data to learn the relationships between the input and the output.
* **Prediction**: After training, the model can predict the output (e.g., house price, disease status, sentiment) for new inputs.

Supervised learning is highly effective for tasks where you have a lot of examples to train on and a clear output to predict.



In the context of supervised learning, the main idea is to teach a machine learning model how to map inputs to outputs by showing it examples of correct answers during the training process. Let’s break down the concept:

**Mapping between Input and Output:**

* **Input**: These are the features or data points you feed into the model. For example, in a house price prediction scenario, the inputs could be things like the house's size, number of bedrooms, location, etc.
* **Output**: This is the result or the value that the model is trying to predict. In the same house price prediction example, the output would be the actual price of the house.

The **goal** of supervised learning is to learn a **mapping** or **relationship** between these inputs and outputs. This mapping is essentially the function or algorithm the model learns from the data.

**Teaching a Model:**

Supervised learning is often compared to a **teacher-student** relationship. Here’s why:

1. **Training with Examples**: Just like a teacher shows students many examples to help them understand a concept, the model is shown many labeled examples during training. These examples include both the input and the correct output.
   * For example, if you’re predicting house prices, the model would be trained on past data that shows house sizes, locations, and their respective prices.
2. **Learning the Mapping**: As the model sees more and more examples, it starts to understand the relationship between inputs and outputs. This is the core of **learning**. For instance, the model might learn that houses with more square feet tend to have higher prices, or houses in certain locations tend to be more expensive.
3. **Feedback on Errors**: Just like a teacher corrects a student’s mistakes, the model adjusts itself based on errors it makes during training. If the model predicts a price that’s too high or too low compared to the actual price, it will adjust its internal parameters (using techniques like gradient descent) to make better predictions in the future.

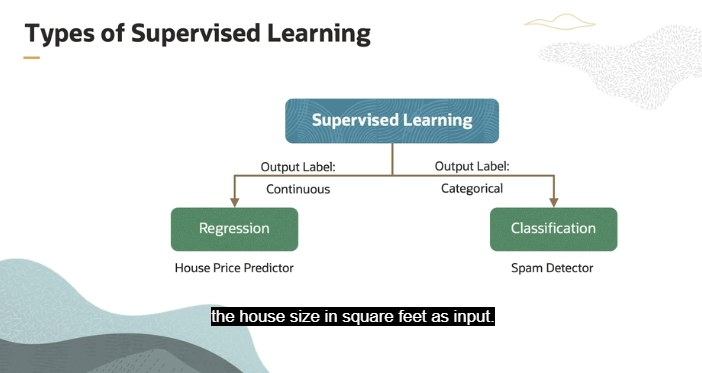
**Final Goal:**

Once the model has learned the mapping between inputs and outputs during training, it can then be used to predict the output (e.g., house prices) for new, unseen inputs (e.g., new house features).

In summary:

* **Supervised learning** involves training a model using labeled data where both inputs and their corresponding outputs are known.
* The model **learns** the relationship (mapping) between the input and output data.
* This is like a student learning from examples, adjusting based on errors, and finally being able to make predictions when presented with new information.

Agar isko In simple term smjhay , toh yeh bilkul usi tarah hai like In school jab teacher humay Mathematics may Ek **prove that** walay questions deta tha jisme smjh lein jo output hota tha wo given hota tha humay just input ka through classify krna hota tha (so here in term of ML we can consider that jo output hai wo labeled data hai jo decide hoga on the basis of input , kay like jo input hoga uskay corresponding hi labeled output decide hoga like customer review example jo customer say review input hoga uski base pa hi decide hoga kay output may jo labeled data hai wo positive, negative or neutral hoga), toh uss learning may yehi hota tha kay pehla teacher humay train krta hai by solving different example and then phr later on at the time of exam wo agar koi unseen question be deday toh we can solve it . yeh srf is ilia bcuz we use the data and examples from past learning and also learned the relationships b/w the input and output. So isi tarah say Supervise ML be hoti hai.



Now in simple terms , its mean that we have a two type of output some are category base or we can say constant values, But some are continuous value that can vary on the basis of input . So in that situation agar toh output categorial hoga like we are creating a model that will detect the customer review , toh waha par humara pass 3 category hai which are positive, negative and neutral, So jo be output ayega wo inhi may say koi hoga therefore in that type of situation we will do **Classification model.** BUT if we are creating a model for house price prediction or weather prediction so in that case the price of house can vary on the basis some feature like size, location ad same goes for weather prediction .

In supervised learning, the outputs or "labels" that the model is trying to predict can come in two different types: **continuous** or **categorical**.

**Explanation from the above Picture :**

The image divides supervised learning into two types based on the nature of the output label:

1. **Continuous Output (Regression)**:
   * If the output you're predicting is a continuous value, meaning it can take on any number within a range, you would use a **regression model**.
   * Example: **House Price Prediction**:
     + If you're predicting the price of a house based on features like its size (square feet), location, and number of rooms, the output (house price) is a continuous number.
     + You might get a price like $250,000 or $400,500, which can be any value within a certain range.
2. **Categorical Output (Classification)**:
   * If the output you're predicting belongs to a set of distinct categories (for example, "spam" or "not spam"), you would use a **classification model**.
   * Example: **Spam Detection**:
     + If you're building a model to detect spam emails, the output is categorical. The model will classify each email into one of two categories: either "spam" or "not spam".

**Regression Example (House Price Prediction):**

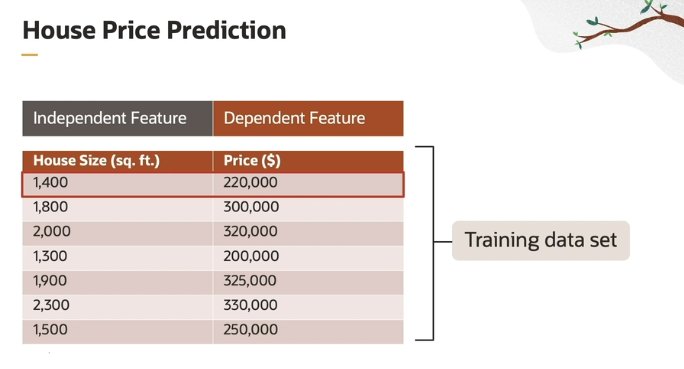
In the house price prediction example, **regression** is used because the goal is to predict a **continuous output**—the exact price of a house. The model takes the **house size in square feet** as one of the input features (along with possibly other features like location or number of bedrooms). It learns to map these inputs to an estimated **house price** (the continuous output).

In regression, the model learns to identify relationships between the input features and the output (the house price). After training, the model can predict the price of a house based on the given inputs, like size, in real-world scenarios.

In summary:

* **Regression** is for continuous outputs, like predicting house prices.
* **Classification** is for categorical outputs, like detecting whether an email is spam or not.

Let us see an example below of house price prediction. In this scenario, we want to predict the price of a house based on a single feature, which is the size of the house in square feet :



**Table Explanation:**

* The table contains two columns:
  1. **House Size (sq. ft.)**: This represents the size of the house in square feet, which is the **independent feature** (or input).
  2. **Price ($)**: This is the price of the house in dollars, which is the **dependent feature** (or output).

**Concepts:**

* **Independent Feature (Input)**:
  + In this case, the **house size** (measured in square feet) is considered independent because it influences the price but is not influenced by anything in this dataset.
  + It's called an **input feature** because this is what you provide to the machine learning model as input for predictions.
* **Dependent Feature (Output)**:
  + The **price** of the house depends on the size of the house and is influenced by the house size.
  + It's called the **output label** because this is the value the model is trying to predict based on the input feature.

**Example:**

Each row in the table represents a single **training example**, also known as a **tuple**. For example:

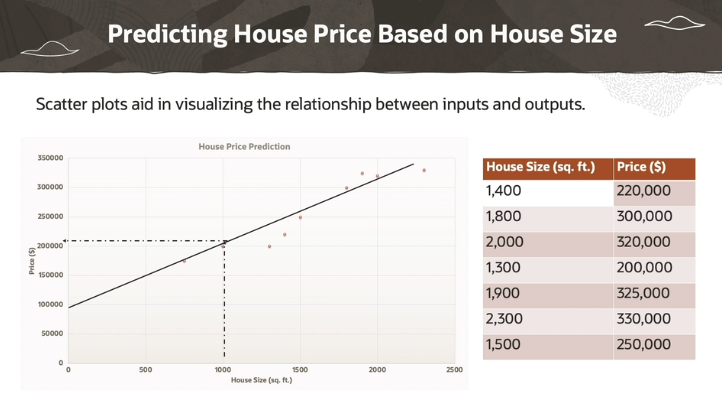
* For a house that is 1,400 square feet, the price is $220,000.
  + In this example, 1,400 is the input (independent feature), and 220,000 is the output (dependent feature).

**Training Data Set:**

* This entire table is the **training data set**, which consists of multiple **training examples** (rows).
* The goal of the machine learning model is to learn from this data set. It analyzes the relationship between the **house size** (input) and the **price** (output) to predict prices for new houses that are not in the training data.

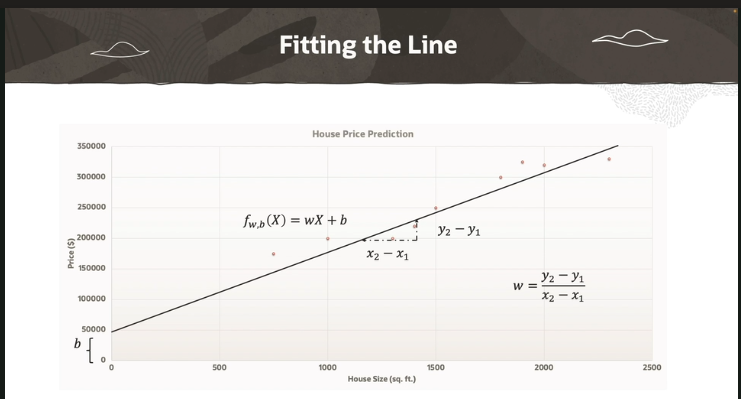
In summary, the data set in the table helps the model learn the relationship between house size and price. After training, the model will be able to predict house prices based on new inputs (house sizes) by using this learned mapping from the training data.

So in simple terms, it means that the output label is **dependent** on the input data/feature . like in above example right we are predicting the house price only on one feature which is **house size ,**  So basically jo **price**(output label) hai it is dependent on house size kay jitna big house hoga utna hi zeada price hongay . But right now in our case jo **house size** (input data) hai wo **independent** bcuz table may or koi asi field nhi hai jispay house size depend kray.



The input and output is plotted in a scatter plot and this visualization helps in understanding the relationship between them. The graph shows that when the size of the house increases, the price also increases. If we fit a straight line  that passes through these points, we can use this line for predicting the price of a house given its size.

See, for example, we want to find the price of a property with size as 1,100. We can just find out the corresponding value from the scatter plot If we know the line passing through the points.



This explanation refers to how a **linear regression model** works in machine learning when predicting house prices based on the size of a house. The relationship between the **house size** and **price** is modeled as a straight line on a graph, where:

* The **x-axis** represents the input feature (house size in square feet).
* The **y-axis** represents the output (house price in dollars).

**Key Concepts:**

1. **Linear Relationship**:
   * The model assumes that there's a linear (straight-line) relationship between the size of the house and the price. In simpler terms, as the size of a house increases, the price also increases in a predictable manner.
2. **Equation of the Line**:  
   The relationship between house size (**x**) and house price (**f(x)**) is represented by a linear equation:

**f(x)=w.x + b**

Where:

* + **f(x)** is the **predicted price** (output) for a given house size **x**.
  + **w** is the **slope** of the line, representing the **rate of change** of house prices based on size (how much the price increases for every square foot increase).
  + **b** is the **bias** (or **y-intercept**), which determines where the line crosses the y-axis (price when house size is 0).

**Understanding Slope (w) and Bias (b):**

* **Slope** (w):
  + The slope controls the **tilt** of the line.
  + If the slope is steep (rising or falling), the house price increases rapidly as the size increases. If the slope is flat, the price increases slowly as the size increases.
  + In this context, the slope represents how much the house price changes for each square foot of house size. For instance, if the slope is 200, it means for every additional square foot, the house price increases by $200.
* **Bias** (b):
  + The bias controls the **vertical position** of the line on the graph.
  + By adjusting the bias, we can move the line **up** or **down** on the graph, which helps to better fit the line to the data.
  + In terms of house prices, the bias is the predicted price when the house size is zero. While this might not make sense physically (since house sizes can’t be zero), it's a mathematical tool to help position the line for better accuracy.
  + And basically the more accurate fitting of Bias result in more accurate predictions.

In simple terms, **bias** in machine learning is like the **starting point** or the **baseline** for predictions. It helps shift the predictions up or down, depending on where we think the line (or model) should be positioned to best fit the data.

Imagine you're guessing the price of a house. Even if you don’t know the size of the house yet, you might have a general idea that houses in a certain area usually start at around, say, $200,000. This "starting point" is the **bias**.

So, in a model:

* **Bias** is what the model predicts when the input (like house size) is **zero**. It’s the base value from which the model starts before it adjusts based on the input.
* The model uses bias to ensure that its predictions aren't too low or too high by default. By adjusting the bias, the model can fit the data better.

**Real-world analogy:**

Think of it like guessing the age of a person. If you know that most people you meet are adults, you might start with a base guess of 25 years old (even if you don’t know any details). This base guess is your bias. After that, you’ll adjust your guess based on additional details like appearance, mannerisms, or how they talk, which would be the other features in the prediction.

**Best-Fitting Line:**

* The goal of the model is to find the **best-fitting line** that passes as close as possible to all the data points in the training set.
* The model does this by **iteratively adjusting** both the slope (w) and the bias (b) to minimize the difference between the predicted prices and the actual prices in the training data.
  + This process is called **training** the model, and it usually involves an algorithm like **gradient descent** to adjust **w** and **b** in small steps until the best fit is found.

**Iteratively adjusting** in machine learning, or in any problem-solving context, refers to the process of making repeated adjustments or changes, step by step, to gradually improve the outcome.

Imagine you're trying to take a picture with your camera to get the perfect shot. You might take a photo, look at it, and then decide to adjust the focus or change the angle slightly and then take another picture. You'll repeat this process—adjusting and taking new pictures—until you're satisfied with the result. Each adjustment gets you closer to the perfect photo.

**In machine learning:**

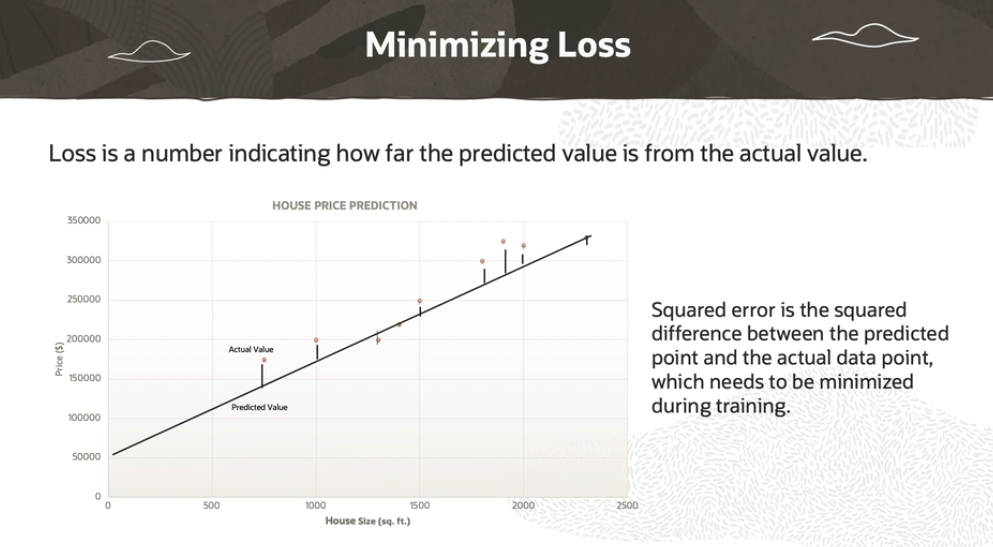
When training a model like the one for predicting house prices:

* You start with a guess for the slope and the bias (parameters).
* You use these parameters to predict prices based on house sizes.
* You compare these predictions to the actual prices and see how far off you are.
* Based on this, you adjust the slope and bias a little bit to reduce the error.
* You repeat this process many times. Each iteration adjusts the slope and bias based on the error observed in the last prediction.
* Gradually, these adjustments make your model's predictions more accurate.

This step-by-step refinement is what we mean by "iteratively adjusting" the model. Each step improves the prediction a bit more, just like adjusting your camera settings bit by bit improves the photo.

**Summary:**

* The equation f(x)=w⋅x+b represents the relationship between **house size** and **house price** as a straight line.
* The slope (w) controls how steep (rising or falling) the line is, representing how much price changes with house size.
* The bias (b) shifts the line up or down, helping to fine-tune the predictions.
* The model adjusts these values iteratively to find the line that best fits the training data, allowing it to make accurate predictions for new data points.



**Linear Regression in Machine Learning:**

In this case, **linear regression** is an algorithm that tries to draw the "best-fitting line" between your input data (like house size) and output data (like house price). The line is positioned in such a way that it closely follows the actual data points.

**Adjusting the Line (Weights and Bias):**

The line is described by an equation:  
**f(x) = w \* x + b**  
Here:

* **w** is the "weight" or the slope of the line (how steep the line is).
* **b** is the "bias" or y-intercept (where the line crosses the y-axis).

Initially, the machine learning algorithm picks random values for the weight and bias. These values are **not perfect** at first, so the line might not fit the data well.

The algorithm then **adjusts** these values over and over (through iterations) to improve how well the line fits the data. This process continues until the line predicts house prices as accurately as possible.

**Error (Difference Between Actual and Predicted):**

* **Predicted value** is the price the model thinks a house should be based on its size.
* **Actual value** is the real price of the house from the training data.
* The difference between these two values is called the **error**.

For example:

* If the actual house price is $300,000 but the model predicts $280,000, the error is $20,000.

**Loss (Penalty for Bad Predictions):**

The **loss** is a measure of how far off the model’s predictions are from the actual values. It's like a score that tells the algorithm how "bad" or "good" the prediction was:

* **Perfect prediction = zero loss** (no penalty).
* **Bad prediction = high loss** (bigger penalty).

**Squared Loss:**

One way to measure how bad the model's predictions are is by calculating **squared loss**.

**Why Squared Loss?**

When we calculate the **error** (difference between the actual price and predicted price), it could be positive or negative. But we want a way to measure how far off the predictions are **without worrying about direction** (whether the prediction is too high or too low, which means kay humay srf jo actual value say predicted value tak ka difference hai wo chaiya baki jo uski direction hai like down or up wo nhi chaiya ).

So, we use **squared loss**:

1. Take the difference between the actual value and the predicted value.
2. **Square** this difference. Squaring makes sure the result is always positive (since negative errors would mess up the calculation).

Example:

* + If the error is $20,000, squaring it gives:  
    $20,000 \* $20,000 = 400,000,000.

**Minimizing Loss:**

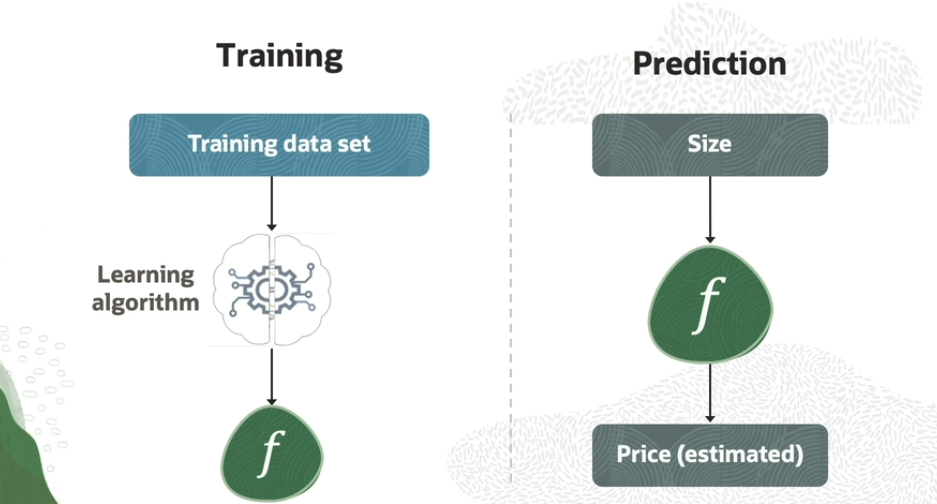
The goal of the algorithm is to **reduce** or **minimize** this loss. It does this by adjusting the weight (slope) and bias (y-intercept) through an iterative process. With each adjustment, the algorithm checks if the loss is getting smaller. If it is, it keeps going until it finds the optimal values for weight and bias.

Once the **optimal weight and bias** are found, the model is ready. At this point, the line fits the data as well as possible, and the model can be used to predict house prices based on new input (house sizes).

**Summary:**

The algorithm "learns" by adjusting the slope and bias to make its predictions more accurate, which reduces the **error**. It keeps adjusting these values to minimize the **loss**, or penalty for bad predictions, and once the loss is minimized, the model is ready to predict new outcomes.

Summary:



To summarize, we train a regression model by giving input and output value  pairs, which we call as data set. The algorithm learns a function f, which is the mapping function. This represents a trained model. This function is used for prediction. If we give house size as input to the trained model, the learned function can predict a house price.

In simple words,, above pic is describing that a process when the model is learning data is known as Training , and after that t become **Trained model**. So after that when it is used for inputting new data and generating output is known as **prediction or inference**.