# Linear Regression

Linear regression predicts a number by fitting a straight line through your data. A

## Introduction

Some predictions are numbers, not categories: the price of a house, tomorrow's temperature, the marks a student might score. When the number you want rises or falls in a fairly steady pattern with one input, the simplest model in machine learning can handle it — **linear regression**. It draws the straight line that best fits your data, and then uses that line to predict new values.

Think of houses in one small neighbourhood. Bigger houses usually cost more. If you plot each house — size on one axis, price on the other — the dots rise from left to right, roughly in a line, with some scatter. Linear regression finds the line that sits closest to all the dots at once. Once you have the line, predicting is easy: pick a size, go up to the line, read off the price.

The line has a simple formula you already know from school maths: price = slope × size + intercept. The **slope** says how much the price rises for each extra unit of size. The **intercept** says where the line crosses the axis — the predicted price when size is zero, which often has no real meaning on its own. The model does not "understand" houses; it just finds the slope and intercept that fit the examples best.

How does the model actually choose the line? It tries lines and keeps score. For each training point, it measures the gap between the real value and the line's prediction — that gap is called a **residual**. A point above the line has a positive residual, a point below it a negative one. The model adds up the squares of all the residuals (squaring stops positive and negative gaps from cancelling each other out) and picks the line with the smallest total. That is the whole fitting method, and it has a plain name: least squares, "least" because it minimises the total squared gap.

This also explains why one wild point can drag the line. A single house priced far above the rest creates a huge squared gap, so the line bends towards it to shrink that gap. If that point is a typing error, the whole line suffers. That is why plotting first, and checking for outliers, is not optional polish — it protects the line itself.

*(Roman Urdu: Linear regression seedhi lakeer khainch kar number ka andaza lagata hai — slope batata hai ke har izafi unit par qeemat kitni barhti hai.)*

## Key points

- **Regression predicts a number.** Price, temperature, marks — any answer that is a quantity, not a category, is a regression job.
- **Linear regression fits a straight line to the data.** The line is chosen to be as close as possible to all the training points at once.
- **The line's formula is: prediction = slope × feature + intercept.** With more than one feature, each feature gets its own slope, but the idea is the same.

- **Slope means "how much the prediction changes per unit of the feature".** A slope of 50 means each extra square foot adds 50 to the predicted price.
- **Intercept means "the prediction when the feature is zero".** It is often just a maths anchor, not a real-world house or student — do not over-interpret it.
- **It only works when the pattern is roughly straight.** If prices rise fast, then flatten, a straight line will miss the shape and the predictions will be off.
- **Linear regression is a great first model.** It is fast, easy to explain, and it gives you a baseline: any fancier model must beat this simple line to be worth it.
- **A residual is the gap between the real value and the prediction.** Residual = real value − predicted value. The model picks the line that makes the squared residuals smallest in total.
- **Outliers drag the line.** One extreme point creates a huge squared gap, so the fitted line bends towards it. Check for errors before trusting the line.
- **With several features, each gets its own slope.** Price might depend on size and on number of rooms; the model learns one slope per feature, and the reading stays the same: each slope is the change per unit of that feature.

## Examples

### Example 1 — Tiny house-price example, by hand

Five small houses, size in hundreds of square feet and price in thousands:

| Size (×100 sq ft) | Price (×1000) | |---|---| | 5 | 150 | | 7 | 200 | | 9 | 250 | | 11 | 300 | | 13 | 350 |

Here the pattern is perfectly steady: each extra 2 units of size adds 50 to the price. The fitted line is price = 25 × size + 25. Check: size 5 gives 25 × 5 + 25 = 150. Correct. For a new house of size 10, the line predicts 25 × 10 + 25 = 275 (that is, 275,000 in these units). Real data is never this clean — the points scatter around the line — but the reading works the same way.

### Example 2 — The same idea in code (scikit-learn)

This example needs scikit-learn installed first: `pip install scikit-learn`. It uses only the standard `LinearRegression` model on the tiny table above:

```
from sklearn.linear_model import LinearRegression

sizes = [[5], [7], [9], [11], [13]]
prices = [150, 200, 250, 300, 350]

model = LinearRegression()
model.fit(sizes, prices)
print("Slope:", model.coef_, "Intercept:", model.intercept_)
print("Prediction for size 10:", model.predict([[10]]))
```

What the result means: the model learns a slope of 25 and an intercept of 25, matching the hand calculation, and predicts 275 for a size-10 house. `.fit` is the training step, `.predict` is the guessing step, and `model.coef_` and `model.intercept_` let you read the line back in plain numbers.

### Example 3 — When a straight line is the wrong shape

Imagine ice-cream sales against temperature: sales rise as it warms up, but past a very hot point people stay indoors and sales flatten. A straight line through that curved pattern would over-predict at both ends and under-predict in the middle. The lesson: always plot your data first. If the dots curve, a straight line is the wrong tool, however easy it is.

### Example 4 — Worked example: study hours vs marks, with residuals

Six students, hours studied and marks scored:

| Hours | Real marks | Line prediction | Residual | |---|---|---|---| | 1 | 42 | 42.2 | −0.2 | | 2 | 48 | 48.8 | −0.8 | | 3 | 57 | 55.4 | +1.6 | | 4 | 61 | 62.0 | −1.0 | | 5 | 70 | 68.5 | +1.5 | | 6 | 74 | 75.1 | −1.1 |

The fitted line is roughly marks = 6.57 × hours + 35.67. Check one row: for 3 hours, 6.57 × 3 + 35.67 = 55.4, and the real mark was 57, so the residual is 57 − 55.4 = +1.6. The residuals are all small and mixed in sign — some above the line, some below — which is what a good fit looks like. For a new student who studied 7 hours, the line predicts 6.57 × 7 + 35.67 = 81.7, about 82 marks. Note what the intercept means here: at zero hours the line predicts about 36 marks, which might be the marks from guessing or from easier questions — but do not treat it as a real student who studied nothing.

### Example 5 — Worked example: the same data in code, and what to read from it

This example needs scikit-learn installed first: `pip install scikit-learn`. It fits the table from Example 4 and prints the line the model found:

```
from sklearn.linear_model import LinearRegression

hours = [[1], [2], [3], [4], [5], [6]]
marks = [42, 48, 57, 61, 70, 74]

model = LinearRegression()
model.fit(hours, marks)
print("Slope:", model.coef_, "Intercept:", model.intercept_)
print("Prediction for 7 hours:", model.predict([[7]]))
print("Training score:", model.score(hours, marks))
```

The slope comes out near 6.57 and the intercept near 35.67, matching the hand calculation, and the prediction for 7 hours is about 82. The training score is close to 1 here because this toy data is almost straight — on real, scattered data it will be lower, and that is normal. Try it: change the 6-hour student's marks from 74 to 40 (a wild outlier) and run again. Watch how far the slope moves from one changed row, and you will have felt, in code, why outliers matter.

**Real-world case — How property sites estimate a house price**

Property websites estimate prices with regression models trained on thousands of past sales: size, number of rooms, location and age of the building as features, sale price as the label. The model learns one slope per feature — for example, how much each extra room typically adds. When you type in a new house's details, the site reads off the fitted line's prediction. But the estimate is only as good as its features: it cannot see a leaking roof or a noisy road, so two houses with the same numbers can still sell for very different prices. That gap between the model's numbers and the real world is exactly the residual — and honest sites show a price range, not a single exact number, for that reason.

## Quick recap

- Linear regression predicts a number by fitting a straight line.
- Slope = change in prediction per unit of the feature. Intercept = prediction at zero.
- It is fast, explainable, and the baseline every fancier model must beat.
- It only fits when the real pattern is roughly straight — plot first, then fit.

## Common mistakes

1. **Reading the intercept as a real prediction** — "a zero-size house costs 25" is nonsense; the intercept is a maths anchor. Fix: interpret the slope, and treat the intercept with care.
2. **Fitting a line to curved data without looking** — the model runs fine and the answers are still wrong. Fix: plot the data before fitting, every time.
3. **Trusting the training fit alone** — a line can hug the training dots and still fail on new houses. Fix: split train/test (lesson ML.3) and report the test score.

1. **Using the line far outside the training range** — predicting for a 100-hour student from data that stops at 10 hours is guessing, not modelling. Fix: only predict inside, or near, the range your data actually covers.
2. **Treating a strong fit as proof of cause** — marks rising with hours does not prove hours caused the marks; motivated students may both study more and score more. Fix: say "linked with", not "caused by", unless you have evidence for cause.

## Practice

1. Using the table in Example 1, predict by hand the price of a size-8 house, then check it against the line formula.
2. Change one price in the table (make the size-9 house cost 290 instead of 250) and describe, in one sentence, what happens to the line.
3. Run the code in Example 2, then add a sixth house (size 15, price 400) and see how the slope changes.

1. Using the table in Example 4, predict the marks for a student who studied 8 hours, then work out the residual if that student actually scores 85. Is the model over- or under-predicting there?

2. Make up your own 5-row table (for example, daily practice minutes vs typing speed) where the pattern is roughly straight, fit the line roughly by eye, and predict one new value.
3. Run the code in Example 5 with the outlier change described there (74 changed to 40). Write down the new slope and one sentence about what the change teaches you.

## Practice quiz

**1. What does linear regression predict?**

- A category
- A number, by fitting a line to the pattern in the data
- A group
- A chart

**Answer:** A number, by fitting a line to the pattern in the data

**2. In a fitted line, what does the slope mean?**

- How many points there are
- How much the prediction rises when the input rises by one unit
- The starting value
- The error size

**Answer:** How much the prediction rises when the input rises by one unit

**3. What does fitting a line mean?**

- Drawing any line
- Finding the line that sits closest to the data points overall
- Connecting the first and last points
- Making the line straight

**Answer:** Finding the line that sits closest to the data points overall

---

**Watch the video lesson:** https://youtu.be/lwKCfLLqnIs

**Read online:** https://scopetreasure.com/notes/ml-linear-regression · More free tools and notes at https://scopetreasure.com