EXPLORE AI
ACADEMY

Accuracy

Trend lines for model accuracy

Trend lines for model accuracy

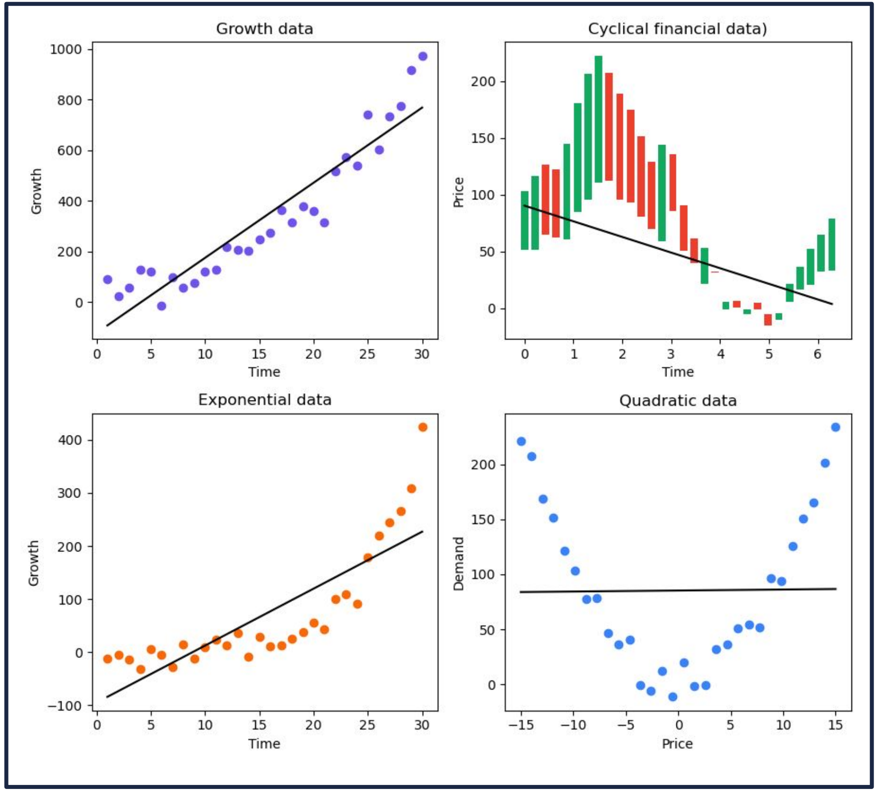
Non-linear data

In **real-world** scenarios, **data** don't always follow a straight line. Instead, it **bends and curves**. We call this **non-linear data**.

In linear analysis we **assume a linear relationship** between the **x** and **y**. However, linear lines often fail to model complex data.

Non-linear data is common in fields like physics, biology, and engineering where many natural phenomena are inherently non-linear, and in finance and economics, data are often cyclical in nature.

We need a **more complex** model to fit to non-linear data, and **poly-n trend lines** are a good example.



Trend lines for model accuracy

What is a poly-n trend line?

> A **polynomial** is an equation that consists of independent **variables** (x) **raised** to **powers**(x^n), **multiplied** by **coefficients** ($\beta_n x^n$), and **summed** together to give **y**.

Polynomial equations have the following format:

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \dots + \beta_n x^n$$

x is a variable and the coefficients (β_0, β_1, β_2... β_n) are numbers multiplied by **x**. The overall shape of the polynomial curve is controlled by the coefficients, β_n. Terms of increasing powers of **x** are added together up to **n** and are called the degree of the polynomial. The degree of the polynomial indicates how many terms we have, for example, a second-degree polynomial:

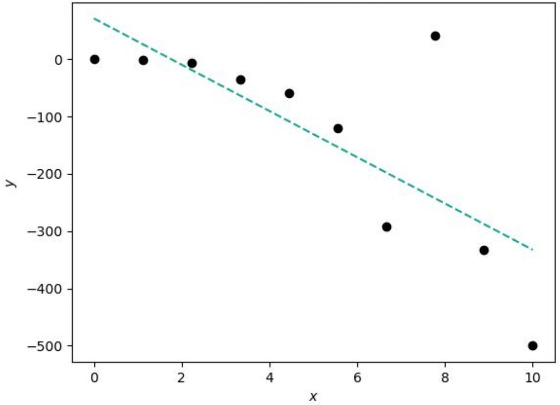
$y = \beta_0 + \beta_1 x + \beta_2 x^2$

The polynomial equation can be used to fit a line of best fit to data in providing a 'poly-n' trend line. By adjusting the degree, we can create more flexible curves for non-linear data, though this may occasionally lead to inaccurate results due to overfitting.

Extending linear equations

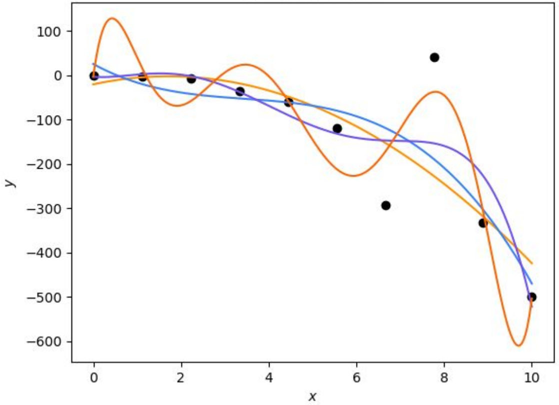
> A **linear** model will have a **straight** trend line, while **poly-n trend lines** will be **curved**. As we **add terms** we can make **more complex** curves.

Linear



$y = \beta_0 + \beta_1 x$

Poly-n

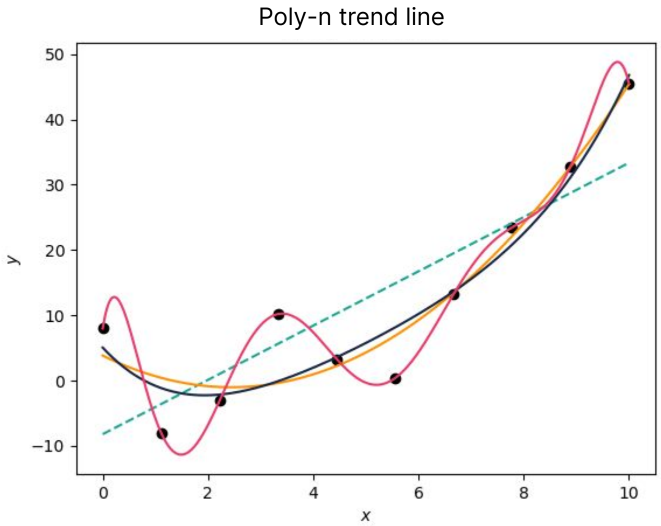


$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \ldots + \beta_n x^n$

Trend lines for model accuracy

Poly-n trend line

> The **higher** the **degree** of the polynomial, the more the trend line can curve to fit the data. However, choosing the **right** degree is crucial to avoid **overfitting** or **underfitting** the data.



Poly-1 trend line is just a straight line (x):

n = 1: $y = \beta_0 + \beta_1 x$ **(underfit)**

Poly-2 trend line has a square term (x^2):

n = 2: $y = \beta_0 + \beta_1 x + \beta_2 x^2$

Poly-3 trend line has a cube term (x^3):

n = 3: $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3$

Poly-8 trend line has an (x^8) term:

n = 8: $y = \beta_0 + \beta_1 x + \beta_2 x^2 + ... + \beta_8 x^8$ **(overfit)**

Trend lines for model accuracy

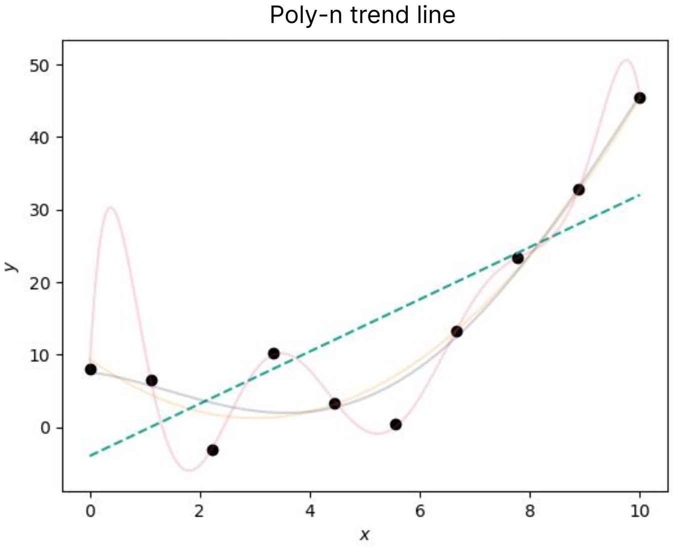
Underfitting: the risk of oversimplification

> **Underfitting** occurs when our model is **too simple** to capture the underlying **pattern** in the data. It has a **high bias**, so it will model data with **low accuracy**.

For example, using a **linear equation** or a **low-degree** polynomial for data that exhibit a **complex**, non-linear relationship can lead to **underfitting**.

An **under-fitted** model will **perform poorly** on current data and any new, unseen data. This is because it hasn't captured the **true underlying pattern**, but rather an **oversimplified** version of it.

This model will struggle to accurately predict values above x = 10, because it hasn’t captured the upward trend in the data well.



Trend lines for model accuracy

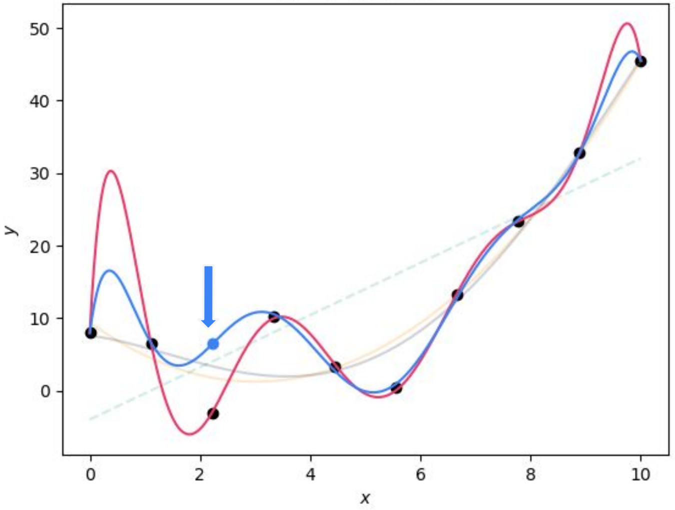
Overfitting: a double-edged sword of flexibility

> As we increase the degree of our polynomial, we **add flexibility** to our model. This can be beneficial, allowing us to **capture complex**, **non-linear patterns** in the data, but can lead to **overfitting**. Overfitting has a **low bias** and a **high variance**.

When a model becomes **too flexible**, it starts to capture **noise**. This is **overfitting**. The **over-fitted** line matches the data perfectly, so it has a perfect fit, and **low bias** since it captures the relationships in the data well.

But the model has a **high variance**, so using another data point like (●) in the plot will result in a totally **different model** that will have different predictions. How do we avoid this?

Poly-n trend line



Trend lines for model accuracy

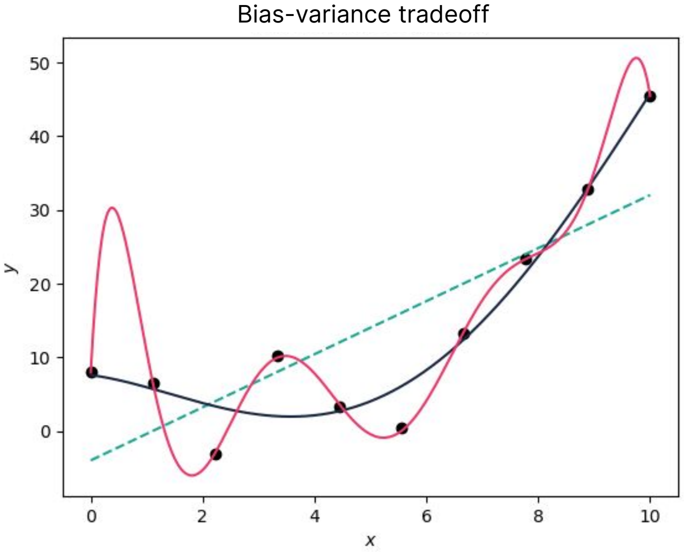
Bias-variance tradeoff

> There are **under-fitted**, **well-fitted** and **over-fitted** models. A **good model** has a **low** mean squared error (**MSE**) and is achieved by choosing a model that is **complex enough** to have **low bias** but **not too complex** to have too much **variance**.

MSE is **high** for the **under-fitted** model due to **too much bias**. ✗

MSE is **high** for the **over-fitted** model because there is **too much variance**. ✗

MSE is **low** for the **well-fitted** model because it is not over- or under-fitted. ✓

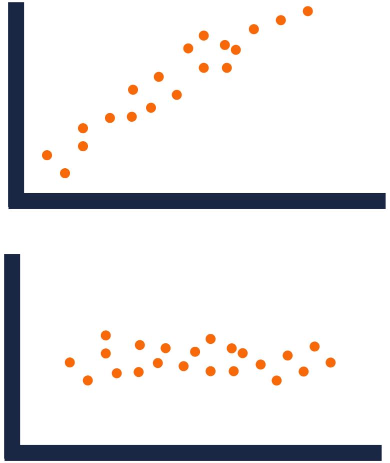


Trend lines for model accuracy

Choosing between linear and polynomial trend lines

> The choice between using a **linear trend** line or a **polynomial trend** line depends on the nature of the relationship between variables.

Linear trend lines



Polynomial trend lines