# Core Intuition behind Supervised Machine Learning

Typically, in any machine learning problem, we want to find the values of $y$ (the output) from those of $x$ (the input). The relationship between them is governed by some true function, which we call $f$. This function actually describes the underlying rule that links input and output.

However, in most cases, we do not know what this true function $f$ looks like. This is why the goal of a machine learning model is to find (derive) an estimated function, which we call $\hat{f}$, that behaves as close to the real one as possible. If we can find such an approximation, we can use it to make predictions or understand the relationship of interest.

So, the first step would be to collect some data, where both the $x$ and $y$ are known. This sample of $x$'s and their corresponding $y$'s is the **training set** for the model. From this, there exists another function, which we'll call $f_{\text{data}}$, that describes the relationship within the observed data. In practice, $f_{\text{data}}$ is only a "snapshot" of the true function $f$. By using this $f_{\text{data}}$, we will try to learn an estimated function $\hat{f}$ that will generalize beyond this sample dataset and ultimately approximate the true function.

Essentially, now that we have the training data, the next step is to find an estimated function $\hat{f}$ that will be similar to $f_{\text{data}}$ and which will be defined in terms of some parameters (often written as $\Theta$ and $\Theta_0$). The process of learning then becomes a problem of **optimization**, where we want to find the values of $\Theta$ and $\Theta_0$ that make our model's predictions as close to the actual outputs as possible by defining an **objective function** (usually denoted by $J$).

This objective function measures how far our model's predictions are from the true data values. In mathematical terms, this equals a loss term, which describes the difference between the predictions from our estimated function $\hat{f}$ and the observed outputs from $f_{\text{data}}$. The loss term depends on the parameters $\Theta$ and $\Theta_0$, as well as on the inputs $x$ and their corresponding $y$s. Thus, the learning process aims to minimize this loss, which means finding the parameters that produce the smallest difference between the prediction and the real data.

$$J = L(\hat{f}, f_{\text{data}}) = L(\Theta, \Theta_0, x_{\text{data}}, y_{\text{data}}) \tag{1}$$

However, $f_{\text{data}}$ is not necessarily identical to the true function $f$ because the data we collected can contain noise or errors that do not truly represent the real relationship. Graph 1 visualizes the core concept of approximating the true function. The estimated function $\hat{f}$ (the dashed yellow line) tries to approximate the true underlying function $f$ (the blue line). The difference between them represents the error our model tries to minimize.

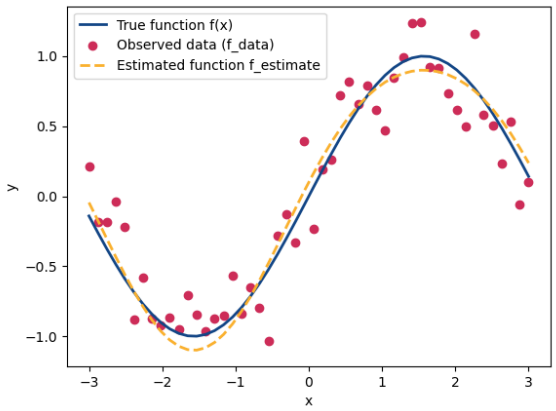


Figure 1: Approximating the true function from data

Therefore, to "discourage" the model from becoming too similar to $f_{\text{data}}$, we need to add another component to our objective function called the **regularization term**, which depends on $\Theta$ and serves as a constraint on the function.

The loss term pushes in the direction of making the model fit the training data too well (overfitting), while the regularization term pushes in the direction of generalization and avoiding memorization of the training data. To balance these terms, a new parameter is introduced to the objective function, called $\lambda$ (sometimes written as $\alpha$). It determines how much weight is given to each term and is not determined through training but rather chosen beforehand or tuned through experimentation.

$$J = L(\Theta, \Theta_0, x_{\text{data}}, y_{\text{data}}) + \alpha R(\Theta) \tag{2}$$

This formulation tries to keep a balance between overfitting and underfitting the model, as depicted in Graph 2.

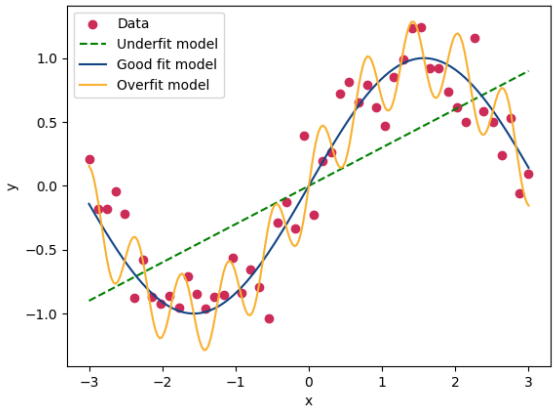


Figure 2: Underfitting vs overfitting vs good fit