# **What Is the Difference Between Bias and Variance?**

Understanding bias and variance, which have roots in statistics, is essential for data scientists involved in machine learning. Bias and variance are used in supervised machine learning, in which an algorithm learns from training data or a sample data set of known quantities. The correct balance of bias and variance is vital to building machine-learning algorithms that create accurate results from their models.

* **What is Bias**

Bias is the amount that a model’s prediction differs from the target value, compared to the training data. Bias error results from simplifying the assumptions used in a model so the target functions are easier to approximate. Bias can be introduced by model selection.

Data scientists conduct resampling to repeat the model building process and derive the average of prediction values. Resampling data is the process of extracting new samples from a data set in order to get more accurate results. There are a variety of ways to resample data, one of which is K fold resampling, in which a given data set is split into a K number of sections, or folds, where each fold is used as a testing set. Resampling can affect bias. If the average prediction values are significantly different from the true value based on the sample data, the model has a high level of bias.

A high level of bias can make the target function easier to learn, leading to underfitting, which occurs when the algorithm is unable to capture relevant relations between features and target outputs. A high bias model typically includes more assumptions about the target function or end result. A low bias model incorporates fewer assumptions about the target function.

A linear algorithm often has high bias, which makes them learn fast. In linear regression analysis, bias refers to the error that is introduced by approximating a real-life problem, which may be complicated, by a much simpler model. Though the linear algorithm can introduce bias, it also makes their output easier to understand. The simpler the algorithm, the more bias it has likely introduced. In contrast, nonlinear algorithms often have low bias.

* **What is Variance**

Variance shows how much the estimate of the target function will alter if different training data were used. Also known as *the error due to variance,* it is the amount by which the prediction, over one training set, differs from the expected predicted value, over all the training sets. Variance measures how inconsistent are the predictions from one another, over different training sets, not whether they are accurate or not. A model with low variance means sampled data is close to where the model predicted it would be. A model with high variance will result in significant changes to the projections of the target function.

Machine learning algorithms with low variance include linear regression, logistics regression, and linear discriminant analysis. Those with high variance include decision trees, support vector machines and k-nearest neighbors.

Models that exhibit small variance and high bias *underfit* the truth target. Models that exhibit high variance and low bias *overfit* the truth target. Note that if your target truth is highly nonlinear, and you select a linear model to approximate it, then you’re introducing a bias resulting from the linear model’s inability to capture nonlinearity. In fact, your linear model is *underfitting* the nonlinear target function over the training set. Likewise, if your target truth is linear, and you select a nonlinear model to approximate it, then you’re introducing a bias resulting from the nonlinear model’s inability to be linear where it needs to be. In fact, the nonlinear model is *overfitting* the linear target function over the training set.

* **The Bias-Variance Trade-off**

Data scientists building machine learning algorithms are forced to make decisions about the level of bias and variance in their models. Ultimately, the trade-off is well known: increasing bias decreases variance, and increasing variance decreases bias. Data scientists have to find the correct balance.

The “tradeoff” between bias and variance can be viewed in this manner – a learning algorithm with low bias must be “flexible” so that it can fit the data well. But if the learning algorithm is too flexible (for instance, too linear), it will fit each training data set differently, and hence have high variance. A key characteristic of many supervised learning methods is a built-in way to control the bias-variance tradeoff either automatically or by providing a special parameter that the data scientist can adjust.

When building a supervised machine-learning algorithm, the goal is to achieve low bias and variance for the most accurate predictions. Data scientists must do this while keeping underfitting and overfitting in mind. A model that exhibits small variance and high bias will underfit the target, while a model with high variance and little bias will overfit the target.

A model with high variance may represent the data set accurately but could lead to overfitting to noisy or otherwise unrepresentative training data. In comparison, a model with high bias may underfit the training data due to a simpler model that overlooks regularities in the data.

The trade-off challenge depends on the type of model under consideration. A linear machine-learning algorithm will exhibit high bias but low variance. On the other hand, a non-linear algorithm will exhibit low bias but high variance. Using a linear model with a data set that is non-linear will introduce bias into the model. The model will underfit the target functions compared to the training data set. The reverse is true as well — if you use a non-linear model on a linear dataset, the non-linear model will overfit the target function.

* **Total Error**

The total error of a machine-learning model is the sum of the bias error and variance error. The goal is to balance bias and variance, so the model does not underfit or overfit the data. As the complexity of the model rises, the variance will increase and bias will decrease. In a simple model, there tends to be a higher level of bias and less variance. To build an accurate model, a data scientist must find the balance between bias and variance so that the model minimizes total error.

**Concept of Overfitting and How to Avoid it**

Overfitting is a modeling error that occurs when a function is too closely fit to a limited set of data points. Overfitting the model generally takes the form of making an overly complex model to explain idiosyncrasies in the data under study. Overfitting is "the production of an analysis that corresponds too closely or exactly to a particular set of data, and may therefore fail to fit additional data or predict future observations reliably".

In contrast, Underfitting occurs when a statistical model cannot adequately capture the underlying structure of the data. An under-fitted model is a model where some parameters or terms that would appear in a correctly specified model are missing. Under-fitting would occur, for example, when fitting a linear model to non-linear data. Such a model will tend to have poor predictive performance.

With reference to regression models, when there’s a large set of explanatory variables that actually have no relation to the dependent variable being predicted, some variables will in general be falsely found to be statistically significant and the researcher may thus retain them in the model, thereby overfitting the model. This is known as Freedman's paradox.

In reality, the data often studied has some degree of error or random noise within it. Thus, attempting to make the model conform too closely to slightly inaccurate data can infect the model with substantial errors and reduce its predictive power.

## **Understanding Overfitting**

For instance, a common problem is using computer algorithms to search extensive databases of historical market data in order to find patterns. Given enough study, it is often possible to develop elaborate theorems which appear to predict things such as returns in the stock market with close accuracy.

However, when applied to data outside of the sample, such theorems may likely prove to be merely the overfitting of a model to what were in reality just chance.

Another example is, let’s say we want to predict if a student will land a job interview based on her resume.

Now, assume we train a model from a dataset of 10,000 resumes and their outcomes.

Next, we try the model out on the original dataset, and it predicts outcomes with 99% accuracy, but then when we run the model on a new (“unseen”) dataset of resumes, we only get 50% accuracy. We can see that **Our model doesn’t *generalize* well from our training data to unseen data.**

## **How to Detect Overfitting**

A key challenge with overfitting, and with machine learning in general, is that we can’t know how well our model will perform on new data until we actually test it.

To address this, we can split our initial dataset into separate *training* and *test* subsets. This method can approximate how well our model will perform on new data.

If our model does much better on the training set than on the test set, then we’re likely overfitting. For example, it would be a big red flag if our model saw 99% accuracy on the training set but only 55% accuracy on the test set.

**Another tip is to start with a very simple model to serve as a benchmark.**

Then, as you try more complex algorithms, you’ll have a reference point to see if the additional complexity is worth it.

## **How to Prevent Overfitting**

Detecting overfitting is useful, but it doesn’t solve the problem. Fortunately, you have several options to try.

**Cross-validation**. This is a powerful preventative measure against overfitting.

The idea is clever: Use your initial training data to generate multiple mini train-test splits. Use these splits to tune your model.

In standard k-fold cross-validation, we partition the data into k subsets, called folds. Then, we iteratively train the algorithm on k-1 folds while using the remaining fold as the test set (called the “holdout fold”).

Cross-validation allows you to tune hyperparameters with only your original training set. This allows you to keep your test set as a truly unseen dataset for selecting your final model.

#### **Train with more data.** It won’t work every time, but training with more data can help algorithms detect the signal better. In the earlier example of modeling height vs. age in children, it’s clear how sampling more schools will help your model.

Of course, that’s not always the case. If we just add more noisy data, this technique won’t help. That’s why you should always ensure your data is clean and relevant.

#### **Remove features.** Some algorithms have built-in feature selection. For those that don’t, you can manually improve their generalizability by removing irrelevant input features. An interesting way to do so is to tell a story about how each feature fits into the model. This is like the data scientist's spin on software engineer’s rubber duck debugging technique, where they debug their code by explaining it, line-by-line, to a rubber duck.

If anything doesn't make sense, or if it’s hard to justify certain features, this is a good way to identify them.

#### **Early stopping.** When training a learning algorithm iteratively, you can measure how well each iteration of the model performs.

Up until a certain number of iterations, new iterations improve the model. After that point, however, the model’s ability to generalize can weaken as it begins to overfit the training data.

Early stopping refers to stopping the training process before the learner passes that point. Today, this technique is mostly used in deep learning while other techniques (e.g. regularization) are preferred for classical machine learning.

#### **Regularization.** Regularization refers to a broad range of techniques for artificially forcing your model to be simpler. A regularization is a form of regression, which constrains/ regularizes or shrinks the coefficient estimates towards zero. In other words, it discourages learning a more complex or flexible model to avoid the risk of overfitting.

Oftentimes, the regularization method is a hyperparameter as well, which means it can be tuned through cross-validation.

The most obvious consequence of overfitting is poor performance on the validation dataset. In all cases, it is important to test a model against data that is outside of the sample used to develop it.

**When to Use Classification over Regression**

Regression and Classification algorithms are Supervised Learning algorithms. Both the algorithms are used for prediction in Machine learning and work with the labeled datasets. But the difference between both is how they are used for different machine learning problems.

The main difference between Regression and Classification algorithms that Regression algorithms are used to predict the continuous values such as price, salary, age, etc. and Classification algorithms are used to predict/Classify the discrete values such as Male or Female, True or False, Spam or Not Spam, etc.

**Differences between regression and classification**

|  |  |
| --- | --- |
| **Regression** | **Classification** |
| In Regression, the output variable must be of continuous nature or real value. | In Classification, the output variable must be a discrete value. |
| The task of the regression algorithm is to map the input value (x) with the continuous output variable(y). | The task of the classification algorithm is to map the input value(x) with the discrete output variable(y). |
| Regression Algorithms are used with continuous data. | Classification Algorithms are used with discrete data. |
| In Regression, we try to find the best fit line, which can predict the output more accurately. | In Classification, we try to find the decision boundary, which can divide the dataset into different classes. |
| Regression algorithms can be used to solve the regression problems such as Weather Prediction, House price prediction, etc. | Classification Algorithms can be used to solve classification problems such as Identification of spam emails, Speech Recognition, Identification of cancer cells, etc. |
| The regression Algorithm can be further divided into Linear and Non-linear Regression. | The Classification algorithms can be divided into Binary Classifier and Multi-class Classifier. |

**References**

[**Bias-Variance Trade off**](https://insidebigdata.com/2014/10/22/ask-data-scientist-bias-vs-variance-tradeoff/)

[**Concept of overfitting**](https://www.investopedia.com/terms/o/overfitting.asp#:~:text=Overfitting%20is%20a%20modeling%20error,limited%20set%20of%20data%20points.&text=Thus%2C%20attempting%20to%20make%20the,and%20reduce%20its%20predictive%20power.)

[**Concept of Overfitting**](https://en.wikipedia.org/wiki/Overfitting) **(wikipedia)**

[**Difference between bias and variance**](https://www.mastersindatascience.org/learning/difference-between-bias-and-variance/)

[**Preventing Overfitting**](https://elitedatascience.com/overfitting-in-machine-learning)

[**Regression over classification**](https://www.javatpoint.com/regression-vs-classification-in-machine-learning#:~:text=The%20main%20difference%20between%20Regression,Spam%20or%20Not%20Spam%2C%20etc.)