How to get the most out of Bias-Variance Tradeoff?

BEGINNER BIAS AND VARIANCE MACHINE LEARNING PYTHON STRUCTURED DATA SUPERVISED TECHNIQUE

This article was published as part of the Data science Blogathon.

In this post, we will explain the bias-variance tradeoff in machine learning and how we can get the most out of it. We will follow up with some illustrative examples and practical implementation in the end.

First things first let us learn about what does bias, variance, and trade-off actually mean practically without learning some bookish definitions.

What is Bias?

So Bias is basically a part of a generalization error that comes across due to some wrong assumptions that we make during our development process i.e. assuming that the data is linear when it is actually quadratic.

A high-bias model is most likely to **underfit** our training data.

What is Variance?

So Variance comes into play whenever our model is extremely sensitive to minor variations in the training data. If the model is having many degrees of freedom like a high degree polynomial model is most likely to have high variance.

A high-variance model is most likely to **overfit** our training data.

Why trade-off?

So after keeping the above things in mind, we can come to a conclusion that increasing the complexity of our model will typically increase its variance and reduce its bias. Conversely, reducing the complexity of our model increases its bias and reduces its variance.

Hence what we will be aiming for are low bias and low variance, but really what we will be doing is trade-off bias for variance

Therefore it is called a **trade-off.**

Bias Variance trade-off:

Let's go ahead and discuss this topic by imagining a dartboard.

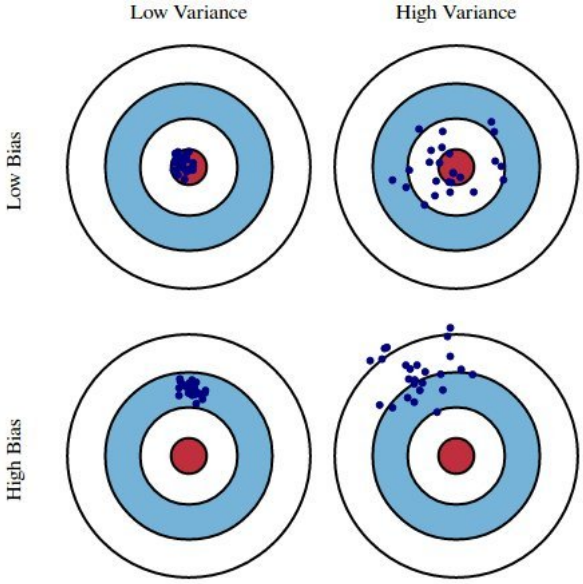


Image source

Imagine that the center of the target is a model that perfectly predicts the correct values. As we move away from the bulls-eye, our predictions get poor and poor. Here we can repeat our entire model building process to get a number of separate hits on the target. Each hit represents an individual realization of our model, assuming there is variability in the training data we gather.

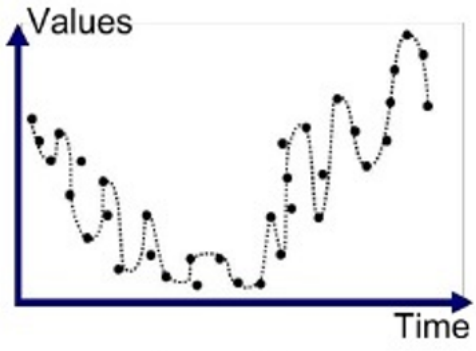
Sometimes we might be lucky enough to get a good distribution of training data so we predict well and we are close to bulls-eye, while sometimes that might not be the case when our training data is filled with outliers or some non-standard values resulting in poor predictions.

These different realizations result in scattering hits on the target.

Looking at the different realizations in the above image we see:

1. Low bias and low variance – This is predicting correct values on the bulls-eye.
2. Low bias and high variance – This will predict values around the bulls-eye with a high degree of variance.
3. High bias and low variance – This will have high bias around a certain location but low variance so all your model predictions are in a certain area.
4. High variance and high bias – This is the worst means predicted values tend to be all over the place.

We all know that there is a common temptation for beginners i.e. to continuously add complexity to a model until it fits the training set very well.



Overfitted

Image source

Here may be as a beginner you decide hey maybe I should make the model more and more complex or flexible so that it hits all those training points. However, if you are going to hit all those training points your model is going to fail to predict new test points which means it will overfit the training data.

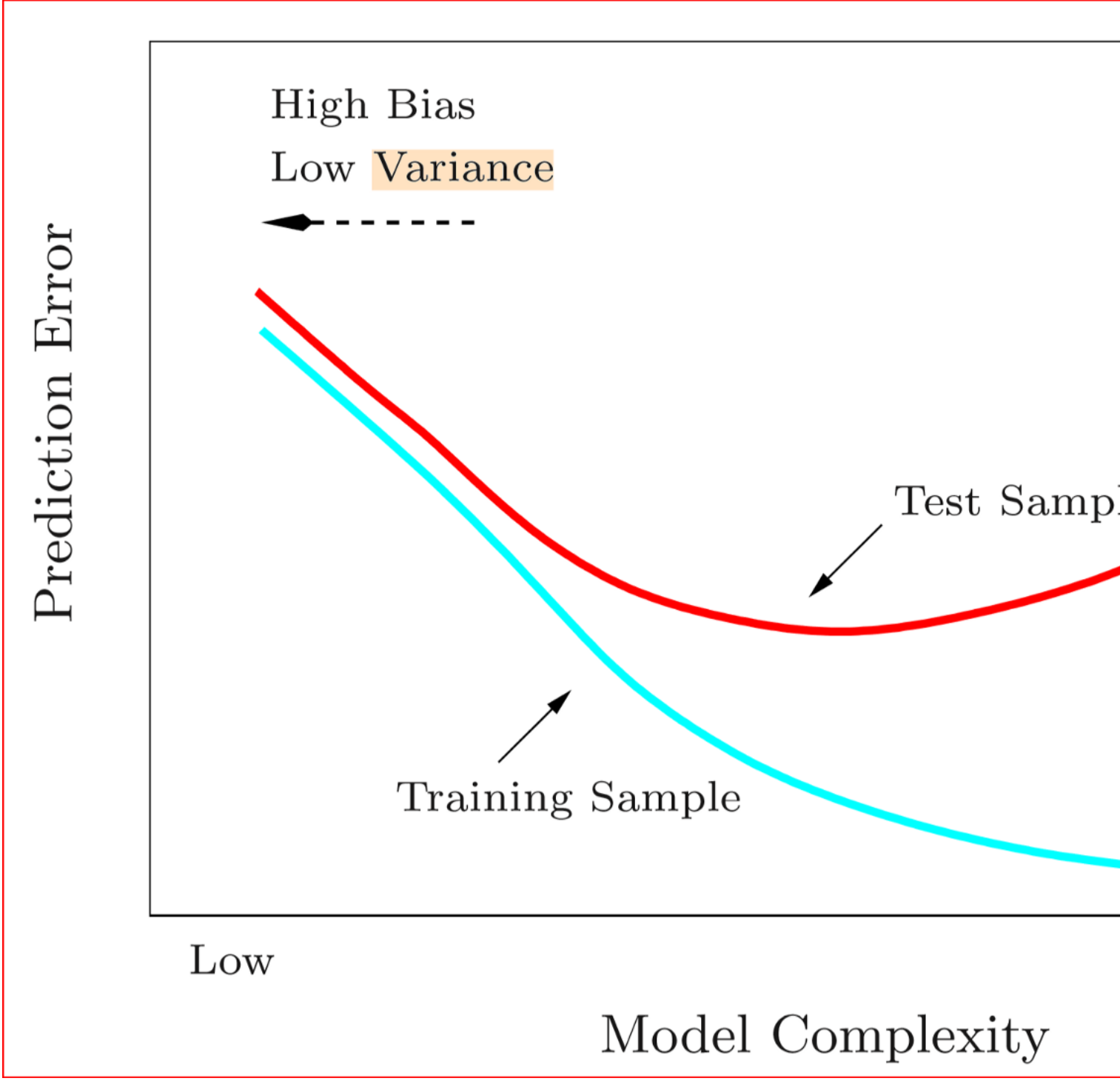


Image source

This is a classic plot to show this as a general stance where you have low versus high model complexity on the X-axis and some sort of prediction error on the Y-axis.

If you move to your left you get a higher bias but lower variance and as you move to your right to a higher complexity model you get a lower bias and higher variance.

Now what we need to do is to pick an optimal point(sweet spot) where you are comfortable with the bias trade-off. If you go to the left of it you will underfit the data and if you go to the right of it you will start to overfit the data.

As all of our explanation portions have come to an end, let's get started with our practical implementation.

Now comes the question of whether we can calculate the bias-variance trade-off for a particular algorithm?

And the answer is No. Technically we cannot calculate the actual values of bias-variance for a model.

Generally, we use bias, variance, irreducible error(noisiness of data), and the bias-variance trade-off to help us select and configure our model.

But we can estimate it in some cases.

The mlxtend library by Sebastian Raschka has a function bias_variance_decomp() function that can estimate the bias and variance for a model.

Firstly, you will need to install mlxtend library.

```
pip install mlxtend
```

After that, we will be using Boston housing dataset as our dataset.

```
from pandas import read_csv from sklearn.model_selection import train_test_split from sklearn.linear_model import LinearRegression from mlxtend.evaluate import bias_variance_decomp url = 'https://raw.githubusercontent.com/jbrownlee/Datasets/master/housing.csv' dataframe = read_csv(url, header=None) data = dataframe.values X, y = data[:, :-1], data[:, -1] X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=101) model = LinearRegression() mse, bias, var = bias_variance_decomp(model, X_train, y_train, X_test, y_test, loss='mse', num_rounds=200, random_seed=1) print('MSE: %.3f' % mse) print('Bias: %.3f' % bias) print('Variance: %.3f' % var)
```

At last, the output we will be getting would be:

```
MSE: 34.904 Bias: 33.438 Variance: 1.466
```

Here we can see that we have a high bias and low variance.

This is a topic you should keep in mind as you should start working on real-world data.

I hope this article made you help gain a better insight into this concept. Leave a comment below if you have any follow-up questions and I will try to answer them.

Thank you,

Karan Amal Pradhan.

The media shown in this article are not owned by Analytics Vidhya and are used at the Author's discretion.

Article Url - https://www.analyticsvidhya.com/blog/2021/06/how-to-get-the-most-out-of-bias-variance-tradeoff/

Karan Pradhan