ML Model Evaluation, Overfitting, Underfitting, Bias, Variance

By

Md. Zubair
Lecturer Department of CSE, Uttara University

Email: md.zubair@uttarauniversity.edu.bd
Blog: https://medium.com/@mzh706
Quora Space: https://abcofdatascienceandml.quora.com/

What is Evaluation Metrics and Why It is Needed?

Evaluation metrics are the metrics by which we can evaluate the performance of a ML model.

Most beginners and practitioners most of the time do not bother about the model performance. Machine learning must be a generalized model, machine learning model cannot have 100 per cent efficiency otherwise the model is known as a biased model. which further includes the concept of overfitting and underfitting.

It is necessary to obtain the accuracy on training data, But it is also important to get a genuine and approximate result on unseen data otherwise Model is of no use. So to build and deploy a generalized model we require to Evaluate the model on different metrics which helps us to better optimize the performance, fine-tune it, and obtain a better result.

If one metric is perfect, there is no need for multiple metrics. To understand the benefits and disadvantages of Evaluation metrics because different evaluation metric fits on a different set of a dataset.

ML Model Evaluation

There are different types of evaluation metrics for evaluating the classification and regression algorithms.

For Classification models

- Confusion matrix
- Precision
- Recall
- F1-Score
- Accuracy
- ROC (Receiver Operating Characteristic) curve and AUC (Area under Curve)

For regression models

- Mean Absolute Error(MAE)
- Mean Squared Error(MSE)
- RMSE
- R squared

Let’s start

Classification Model's Evaluation Metrics

- **Confusion Matrix**

The confusion matrix is a matrix used to determine the performance of the ***classification models*** for a given set of test data. It can only be determined if the true values for test data are known. Since it shows the errors in the model performance in the form of a matrix, hence also known as an **error matrix**.

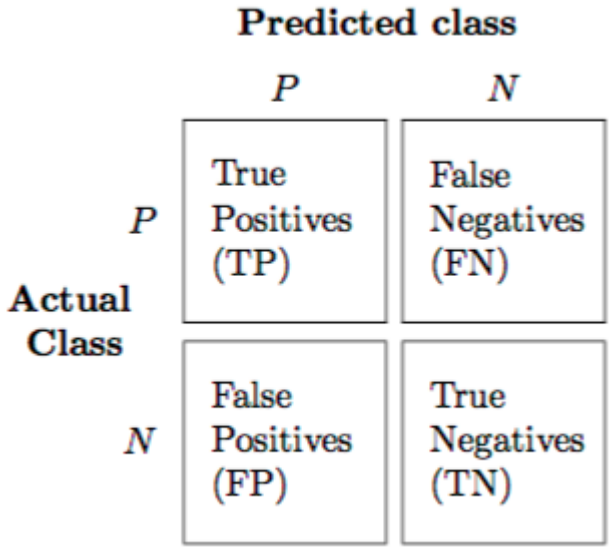


Figure 1- Confusion matrix for binary classification

- **True Negative:** Model has given prediction *No*, and the real or actual value was also *No*.
- **True Positive:** The model has predicted *Yes*, and the actual value was also *True*.
- **False Negative:** The model has predicted *No*, but the actual value was *Yes*.
- **False Positive:** The model has predicted *Yes*, but the actual value was *No*.

		Predicted		
		Yes	**No**	
Actual	Yes	24 (TP)	8 (FN)	32
	No	3 (FP)	65 (TN)	68
		27	73	

Figure 2- Confusion matrix with an example

From the above example, we can conclude that:

•The table is given for the two-class classifier, which has two predictions "Yes" and "NO." Here, Yes defines that patient has the disease, and No defines that patient does not has that disease.

•The classifier has made a total of **100 predictions**. Out of 100 predictions, **89 are true predictions**, and **11 are incorrect predictions**.

•The model has given prediction "yes" for 32 times, and "No" for 68 times. Whereas the actual "Yes" was 27, and actual "No" was 73 times.

- **Precision**

It can be defined as the number of *correct positive outputs provided by the model or out of all positive classes that have predicted correctly by the model*, how many of them were actually true. It can be calculated using the below formula:

$$\text{Precision} = \frac{TP}{TP+FP}$$

As the evaluation metric indicates the accuracy of positive outcomes out of the predicted positive classes, it plays a significant role where positive outcome is very crucial.

Task: Calculate the precision for the confusion matrix of figure 2.

- **Recall**

It is defined as the out of total positive classes, how our model predicted correctly. The recall must be as high as possible.

$$\text{Recall} = \frac{TP}{TP+FN}$$

The metric is very crucial in case of medical diagnosis and so on.

Task: Calculate the recall for the confusion matrix of figure 2.

- **F1-Score**

If two models have low precision and high recall or vice versa, it is difficult to compare these models. So, for this purpose, we can use F-score. This score helps us to evaluate the recall and precision at the same time. The F-score is maximum if the recall is equal to the precision. It can be calculated using the below formula:

$$\text{F-measure} = \frac{2*Recall*Precision}{Recall+Precision}$$

In a nutshell, f1-score is the trade-off between precision and recall score.

Task: Calculate the f1-score for the confusion matrix of figure 2.

- **Classification Accuracy**

It is one of the important parameters to determine the accuracy of the classification problems. It defines how often the model predicts the correct output. It can be calculated as the ratio of the number of correct predictions made by the classifier to all number of predictions made by the classifiers. The formula is given below:

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN}$$

Task: Calculate the accuracy for the confusion matrix of figure 2.

- **Precision Vs Recall**

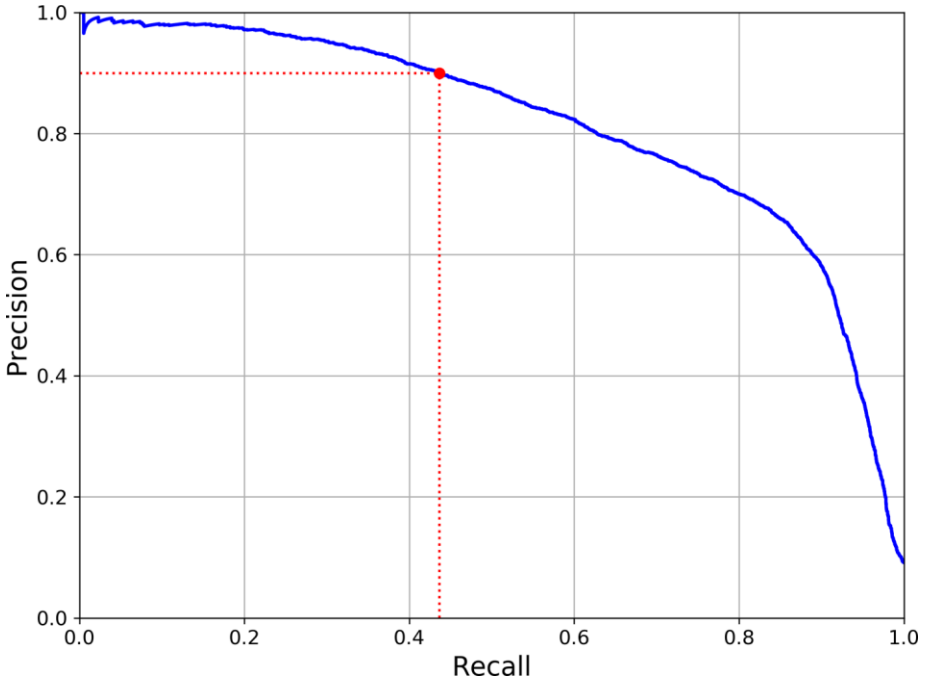


Figure- Precision versus recall

A good precision/recall trade-off is to plot precision directly against recall, as shown in Figure.

You can see that precision really starts to fall sharply around 80% recall. You will probably want to select a precision/recall trade-off just before that drop—for example, at around 60% recall. But of course, the choice depends on your project.

- **ROC Curve**

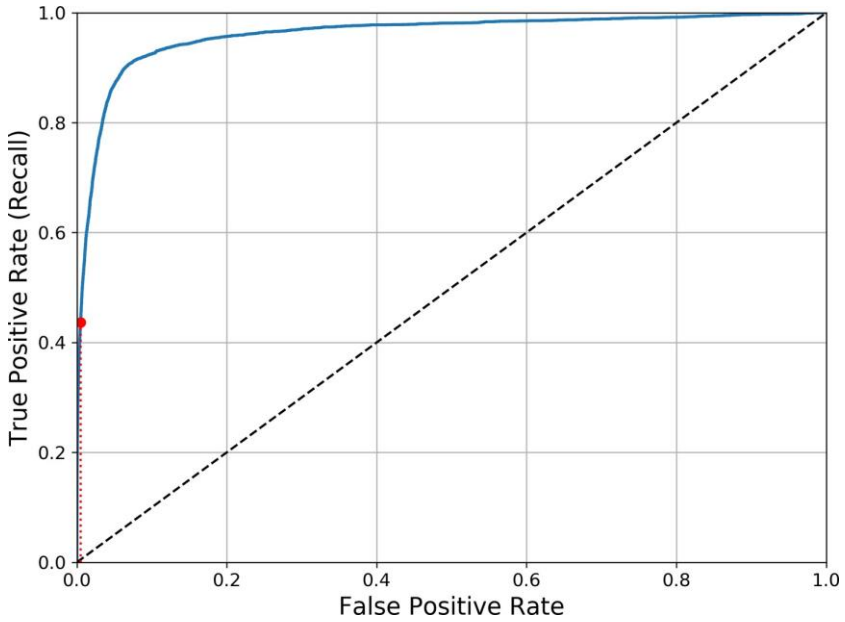


Figure- This ROC curve plots the false positive rate against the true positive rate for all possible thresholds; the red circle highlights the chosen ratio.

The *receiver operating characteristic* (ROC) curve is another common tool used with binary classifiers. It is very similar to the precision/recall curve, but instead of plotting precision versus recall, the ROC curve plots the *true positive rate* (another name for recall) against the *false positive rate* (FPR). **The FPR is the ratio of negative instances that are incorrectly classified as positive.** It is equal to 1 – the *true negative rate* (TNR), which is the ratio of negative instances that are correctly classified as negative.

The TNR is also called *specificity*. Hence, the ROC curve plots *sensitivity* (recall) versus 1 – *specificity*.

The dotted line represents the ROC curve of a purely random classifier; a good classifier stays as far away from that line as possible (toward the top-left corner).

Once again there is a trade-off: the higher the recall (TPR), the more false positives (FPR) the classifier produces.

- **AUC**

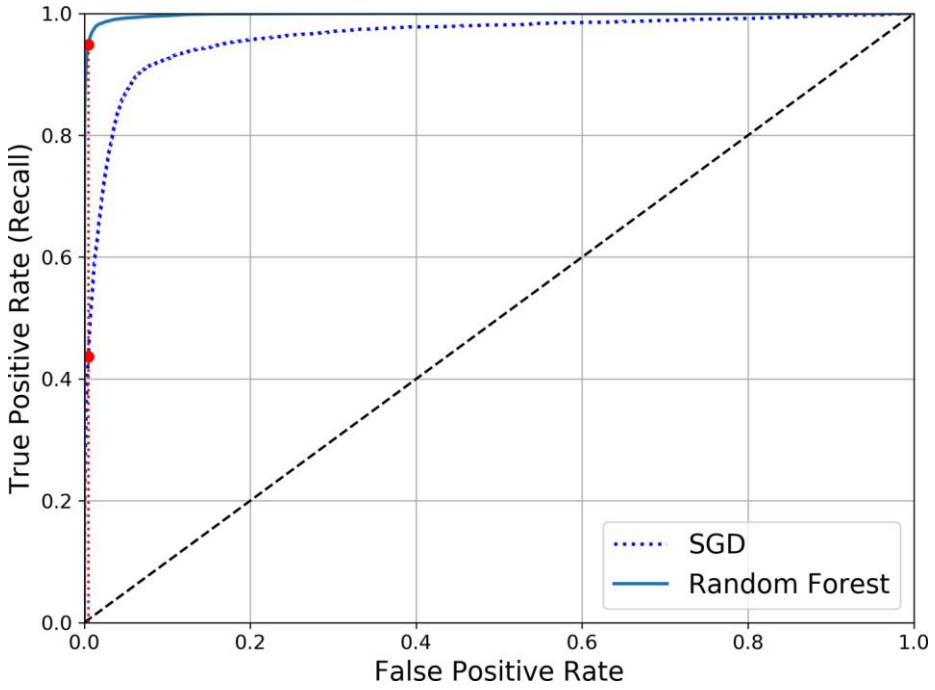


Figure *- Comparing ROC curves: the Random Forest classifier is superior to the SGD classifier because its ROC curve is much closer to the top-left corner, and it has a greater AUC*

One way to compare classifiers is to measure the *area under the curve* (AUC). A perfect classifier will have a ROC AUC equal to 1, whereas a purely random classifier will have a ROC AUC equal to 0.5.

When to Use Which Evaluation Graph

Since the ROC curve is so similar to the precision/recall (PR) curve, you may wonder how to decide which one to use. As a rule of thumb, you should prefer the PR curve whenever the positive class is rare or when you care more about the false positives than the false negatives. Otherwise, use the ROC curve. For example, looking at the previous ROC curve (and the ROC AUC score), you may think that the classifier is really good. But this is mostly because there are few positives (5s) compared to the negatives (non-5s). In contrast, the PR curve makes it clear that the classifier has room for improvement (the curve could be closer to the top-left corner).

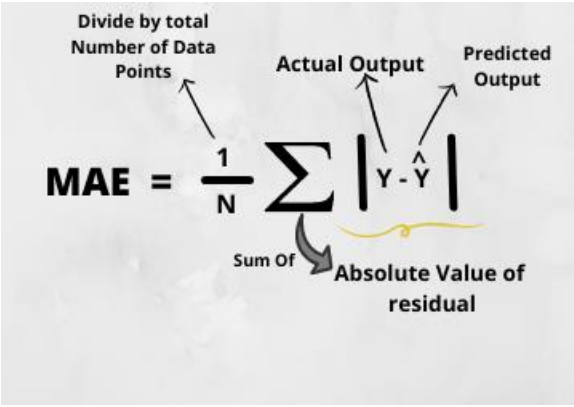
Regression Model's Evaluation Metrics

- **Mean Absolute Error(MAE)**

MAE is a very simple metric which calculates the **absolute difference between actual and predicted values.** To better understand, let's take an example you have input data and output data and use Linear Regression, which draws a best-fit line.

Now you have to find the MAE of your model which is basically a mistake made by the model known as an error. Now find the difference between the actual value and predicted value that is an absolute error but we have to find the mean absolute of the complete dataset.

so, sum all the errors and divide them by a total number of observations And this is MAE. And we aim to get a minimum MAE because this is a loss.



Advantages of MAE

•The MAE you get is in the same unit as the output variable.

•It is most Robust to outliers.

Disadvantages of MAE

•The graph of MAE is not differentiable so we have to apply various optimizers like Gradient descent which can be differentiable.

- **Mean Square Error**

MSE is a most used and very simple metric with a little bit of change in mean absolute error. Mean squared error states that finding the squared difference between actual and predicted value. So, above we are finding the absolute difference and here we are finding the squared difference.

What actually the MSE represents? It represents the squared distance between actual and predicted values. We perform squared to avoid the cancellation of negative terms and it is the benefit of MSE.

$$MSE = \frac{1}{n}\Sigma\underbrace{\left(y - \hat{y}\right)}^{2}$$

The square of the difference between actual and predicted

Advantages of MSE

The graph of MSE is differentiable, so you can easily use it as a loss function.

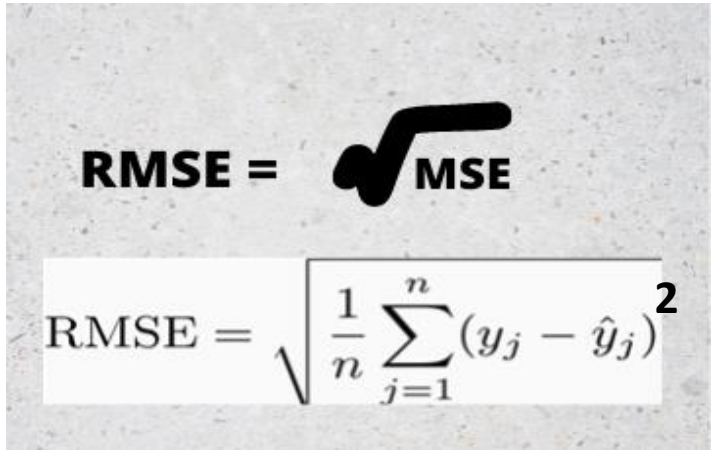
Disadvantages of MSE

•The value you get after calculating MSE is a squared unit of output. for example, the output variable is in meter(m) then after calculating MSE the output we get is in meter squared.

•If you have outliers in the dataset then it penalizes the outliers most and the calculated MSE is bigger. So, in short, It is not Robust to outliers which were an advantage in MAE.

- **Root Mean Square Error**

As RMSE is clear by the name itself, that it is a simple square root of mean squared error.



Advantages of RMSE

• The output value you get is in the same unit as the required output variable which makes interpretation of loss easy.

Disadvantages of RMSE

•It is not that robust to outliers as compared to MAE.

Most of the time people use RMSE as an evaluation metric and mostly when you are working with deep learning techniques the most preferred metric is RMSE.

- **R Squared (R2)**

R2 score is a metric that tells the performance of your model, not the loss in an absolute sense that how many wells did your model perform.

In contrast, MAE and MSE depend on the context as we have seen whereas the R2 score is independent of context.

$$\text{R2 Squared} = 1 - \frac{SSr}{SSm}$$

SSr = Squared sum error of regression line

SSm = Squared sum error of mean line

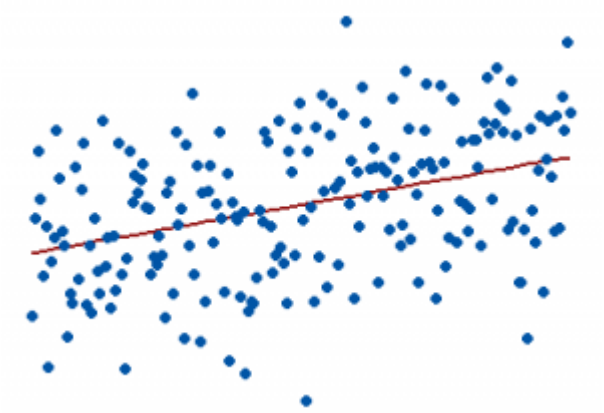


Figure- Not good fit model

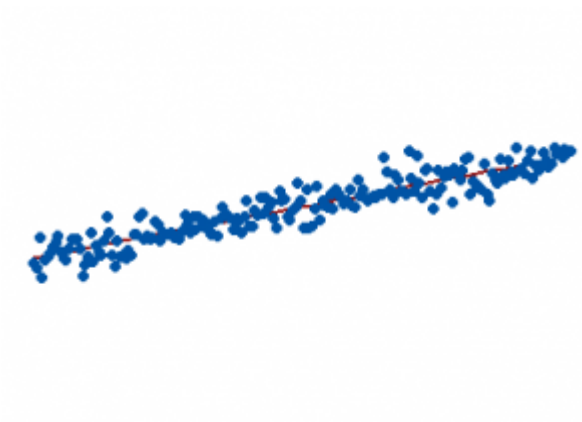


Figure- Good fit model

R-squared is always between 0 and 100%:

•0% represents a model that does not explain any of the variation in the response variable around its mean. The mean of the dependent variable predicts the dependent variable as well as the regression model.

•100% represents a model that explains all the variation in the response variable around its mean.

Overfitting and Underfitting In Machine Learning

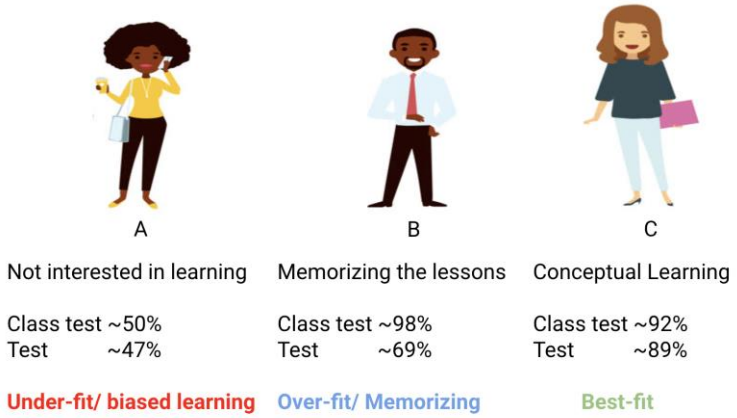


Figure- Real life example of Overfitting and Underfitting

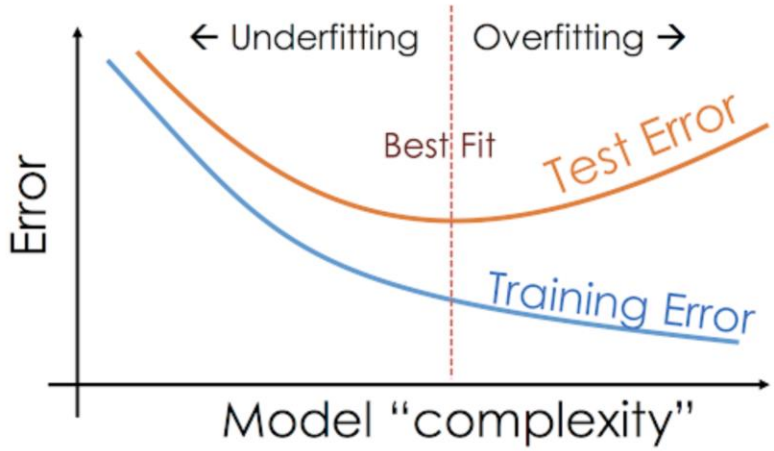


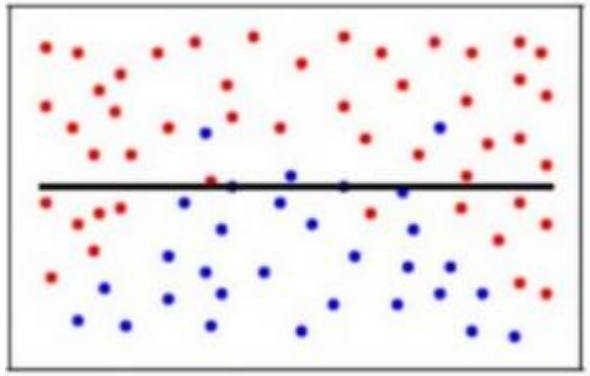
Figure- ML example of Overfitting and Underfitting

- Overfitting is a concept in data science, which occurs **when a statistical model fits exactly against its training data**. When this happens, the algorithm unfortunately cannot perform accurately against unseen data, defeating its purpose.

- Underfitting is a scenario in data science where **a data model is unable to capture the relationship between the input and output variables accurately**, generating a high error rate on both the training set and unseen data.

Bias In ML

In the simplest terms, Bias is the difference between the Predicted Value and the Expected Value. To explain further, the model makes certain assumptions when it trains on the data provided. When it is introduced to the testing/validation data, these assumptions may not always be correct.



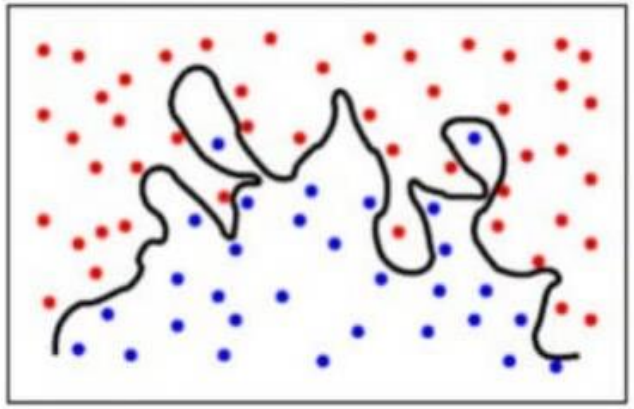
High bias mainly occurs due to a much simple model. Below are some ways to reduce the high bias:

- Increase the input features as the model is underfitted.
- Increase the training data.
- Decrease the regularization term.
- Use more complex models, such as including some polynomial features.

Variance in ML

The Variance is when the model takes into account the fluctuations in the data i.e. the noise as well. So, what happens when our model has a high variance?

The model will still consider the variance as something to learn from. That is, the model learns too much from the training data, so much so, that when confronted with new (testing) data, it is unable to predict accurately based on it.



- A high variance model leads to overfitting.
- Increase model complexities.

Ways to Reduce High Variance:

- Reduce the input features or number of parameters as a model is overfitted.
- Do not use a much complex model.
- Increase the training data.
- Increase the Regularization term.

Bias Variance Trade-off

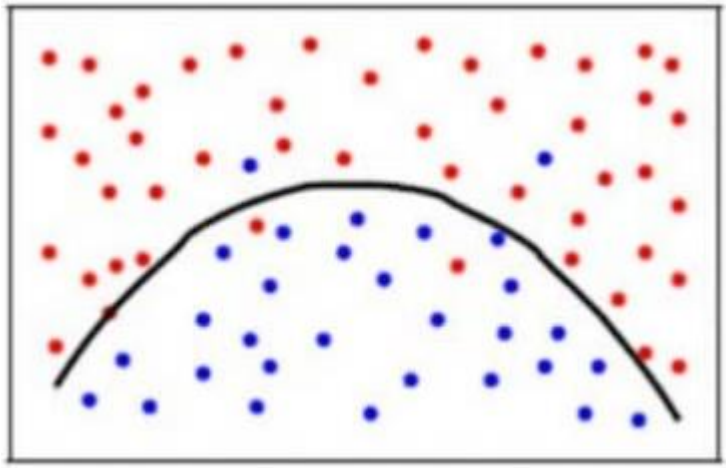


Figure-1: Bias Variance Trade-off

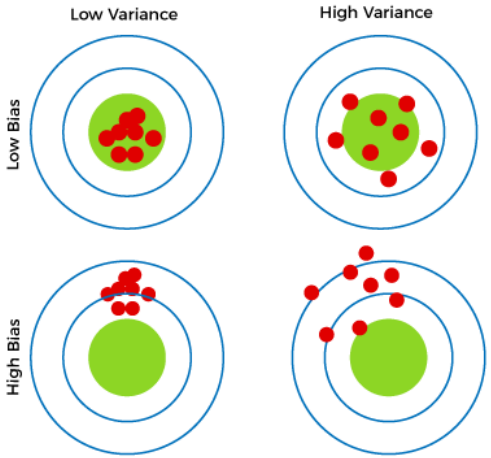


Figure-2: Bias Variance Trade-off

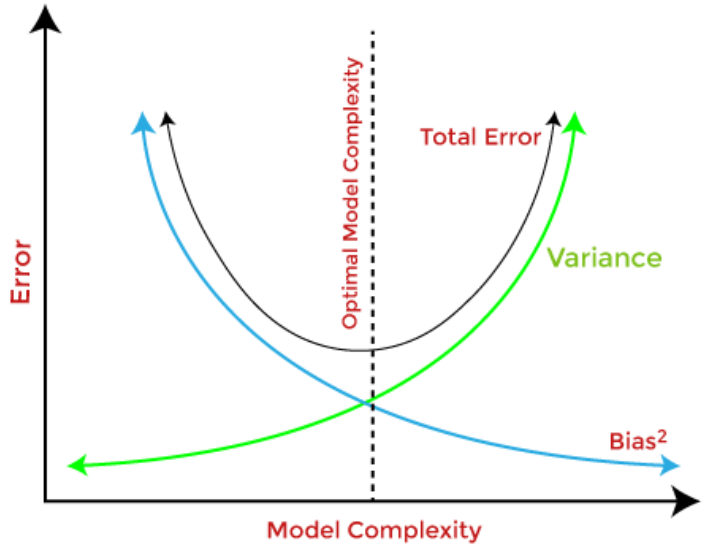


Figure-3: Bias Variance Trade-off

Bias-Variance trade-off is about finding the sweet spot to make a balance between bias and variance errors

The End