UNIT-2

Preparing the Model, Modelling, and Evaluation

2.1 Selecting a Model

- Input variables can be denoted by X, while individual input variables are represented as X1, X2, X3, ..., Xn, and output variables by symbol Y. The relationship between X and Y is represented in the general form:

  $Y = f(X) + e$,

  where ‘f ’ is the target function and ‘e’ is a random error term. Just like a target function concerning a machine learning model, some of these functions which are frequently tracked are
- A cost function (also called an error function) helps to measure the extent to which the model is going wrong in estimating the relationship between X and Y. In that sense, the cost function can tell how badly the model is performing. For example, R-squared (to be discussed later in this chapter) is a cost function of a regression model.
- The loss function is almost synonymous with the cost function – only the difference is that the loss function is usually a function defined by a data point, while the cost function is for the entire training data set.
- Machine learning is an optimization problem. We try to define a model and tune the parameters to find the most suitable solution to a problem. However, we need to have a way to evaluate the quality or optimality of a solution. This is done using an objective function. Objective means goal.
- The objective function takes in data and model (along with parameters) as input and returns a value. The target is to find values of model parameters to maximize or minimize the return value. When the objective is to minimize the value, it becomes synonymous with to cost function. Examples: maximize the reward function in reinforcement learning,

maximize the posterior probability in Naive Bayes, and minimize squared error in regression.

2.1.1 Predictive models

- Models for supervised learning or predictive models, as is understandable from the name itself, try to predict certain values using the values in an input data set. The learning model attempts to establish a relation between the target feature, i.e. the feature being predicted, and the predictor features. The predictive models have a clear focus on what they want to learn and how they want to learn. Predictive models, in turn, may need to predict the value of a category or class to which a data instance belongs. Below are some examples:

  1. Predicting a win/loss in a cricket match
  2. Predicting whether a transaction is fraud
  3. Predicting whether a customer may move to another product

  The models that are used for the prediction of target features of categorical value are known as classification models. The target feature is known as a class and the categories to which classes are divided are called levels. Some of the popular classification models include k-Nearest Neighbor (CNN), Naïve Bayes, and Decision Tree. Predictive models may also be used to predict numerical values of the target feature based on the predictor features. Below are some examples:

  1. Prediction of revenue growth in the succeeding year
  2. Prediction of rainfall amount in the coming monsoon
  3. Prediction of potential flu patients and demand for flu shots next winter

- The models that are used for the prediction of the numerical value of the target feature of a data instance are known as regression models. Linear Regression and Logistic Regression models are popular regression models.

2.1.2 Descriptive models

- Models for unsupervised learning or descriptive models are used to describe a data set or gain insight from a data set. There is no target feature or single feature of interest in the case of unsupervised learning. Based on the value of all features, interesting patterns or insights are derived about the data set.
- Descriptive models group together similar data instances, i.e. data instances having a similar value of the different features are called clustering models. Examples of clustering include

  1. Customer grouping or segmentation based on social, demographic, ethnic, etc. factors

  2. Grouping of music based on different aspects like genre, language, period, etc.

  3. Grouping of commodities in an inventory the most popular model for clustering is k-means.
- Descriptive models related to pattern discovery are used for market basket analysis of transactional data. In market basket analysis, based on the purchase pattern available in the transactional data, the possibility of purchasing one product based on the purchase of another product is determined. For example, transactional data may reveal a pattern that generally a customer who purchases milk also purchases biscuits at the same time. This can be useful for targeted promotions or in-store setups.
- Promotions related to biscuits can be sent to customers of milk products or vice versa. Also, in the store products related to milk can be placed close to biscuits.

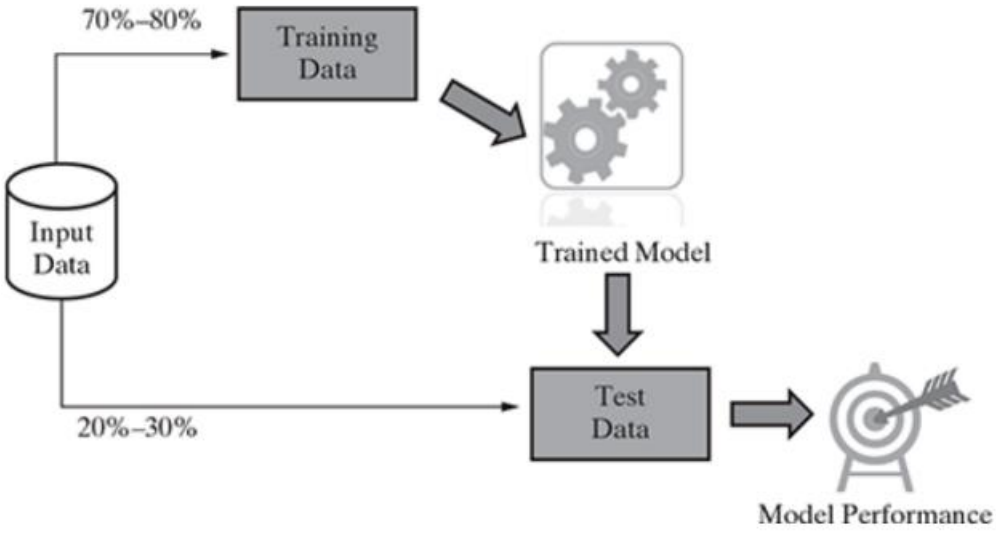
2.2 Training a Model for Supervised Learning

- Training a model simply means learning (determining) good values for all the weights and the bias from labeled examples. In supervised learning, a machine learning algorithm builds a model by examining many examples and attempting to find a model that minimizes loss; this process is called empirical risk minimization.

2.2.1 Cross-validation techniques

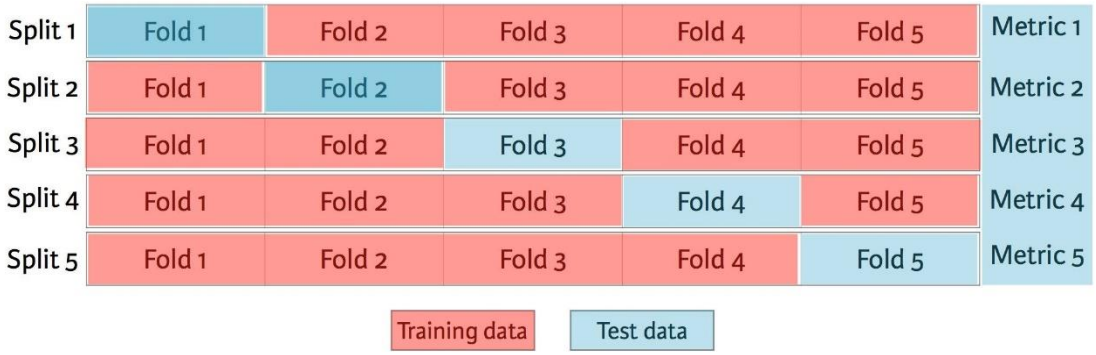
Holdout method

- In the case of supervised learning, a model is trained using the labeled input data. However, how can we understand the performance of the model? The test data may not be available immediately. Also, the label value of the test data is not known. That is the reason why a part of the input data is held back (that is how the name holdout originates) for evaluation of the model. This subset of the input data is used as the test data for evaluating the performance of a trained model.
- In general, 70%–80% of the input data (which is labeled) is used for model training. The remaining 20%–30% is used as test data for validation of the performance of the model. However, a different proportion of dividing the input data into training and test data is also acceptable. To make sure that the data in both buckets are similar, the division is done randomly.
- Random numbers are used to assign data items to the partitions. This method of partitioning the input data into two parts – training and test data, which is by holding back a part of the input data for validating the trained model is known as the holdout method.



K-fold Cross-validation Method

- Cross-validation or 'k-fold cross-validation' is when the dataset is randomly split up into 'k' groups. One of the groups is used as the test set and the rest are used as the training set. The model is trained on the training set and scored on the test set. Then the process is repeated until each unique group has been used as the test set.
- For example, for 5-fold cross-validation, the dataset would be split into 5 groups, and the model would be trained and tested 5 separate times so each group would get a chance to be the test set. This can be seen in the graph below.



2.3 Model Representation and Interpretability

- A machine learning model is interpretable if it's easy for humans to understand how it makes decisions. Interpretability requires a greater level of detail than explainability, which focuses on explaining the decisions made.

  Here are some types of interpretability in machine learning:

  - Model-agnostic: Uses tools that are applied after a model has been trained.
  - Surrogate models: Created by training a linear regression or decision tree on the original inputs and predictions of a complex model.
  - Modularity: A model is modular if a meaningful portion of its prediction-making process can be interpreted independently.

  Here are some types of machine learning models:

  - Descriptive: Helps understand what happened in the past.
  - Prescriptive: Automates business decisions and processes based on data.
  - Predictive: Predicts future business scenarios.

2.3 Evaluating the Performance of a Model

- Model evaluation is the process of using metrics to understand the performance of a machine learning model. It's important to assess the model's efficacy during the initial research phases.

2.4.1 Performance Metrics for Classification

- **Accuracy**

  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 the total number of predictions made by the classifiers.

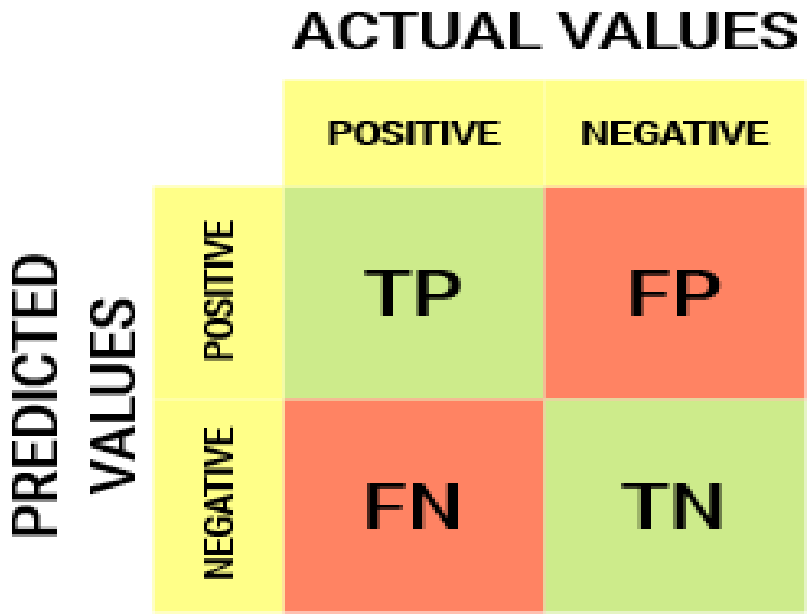
$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

- **Confusion Matrix**

  A confusion matrix is a tabular representation of prediction outcomes of any binary classifier, which is used to describe the performance of the classification model on a set of test data when true values are known.

  The confusion matrix is simple to implement, but the terminologies used in this matrix might be confusing for beginners.

  A typical confusion matrix for a binary classifier looks like the below image (However, it can be extended to use for classifiers with more than two classes).



1. True Positive (TP): In this case, the prediction outcome is true, and it is true in reality, too.
2. True Negative (TN): in this case, the prediction outcome is false, and it is false in reality, too.
3. False Positive (FP): In this case, prediction outcomes are true, but they are false in actuality.

4. False Negative (FN): In this case, predictions are false, and they are true in actuality.

- **Precision**

  The precision metric is used to overcome the limitation of Accuracy. The precision determines the proportion of positive prediction that was correct. It can be calculated as the True Positive or predictions that are true to the total positive predictions (True Positive and False Positive).

$$\text{Precision} = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

- **Recall**

  The recall is calculated as the ratio between the number of Positive samples correctly classified as Positive to the total number of Positive samples. The recall measures the model's ability to detect positive samples. The higher the recall, the more positive samples detected.

$$\text{Recall} = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

- **F-Score**

  F-score or F1 Score is a metric to evaluate a binary classification model based on predictions that are made for the positive class. It is calculated with the help of Precision and Recall So, the F1 Score can be calculated as the harmonic mean of both precision and Recall, assigning equal weight to each of them.

  The formula for calculating the F1 score is given below:

$$F1 - score = 2 * \frac{precision * recall}{precision + recall}$$

When to use F-Score?

As the F-score makes use of both precision and recall, it should be used if both of them are important for evaluation, but one (precision or recall) is slightly more important to consider than the other. For example, when False negatives are comparatively more important than false positives, or vice versa.

2.4.2 Evaluation Metrics for Regression Task

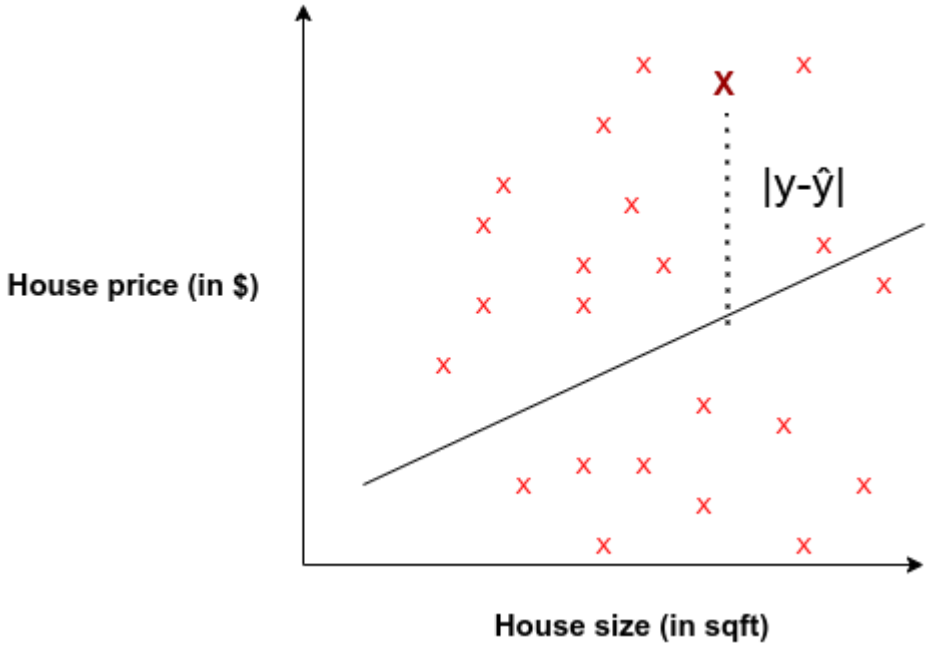
Mean Absolute Error (MAE)

Mean Absolute Error is the average of the difference between the ground truth and the predicted values. Mathematically, it is represented as :

$$MAE = \frac{1}{N}\sum_{j=1}^{N}|y_j - \check{y}_j|$$

Where:

- y_j: ground-truth value
- y_hat: predicted value from the regression model
- N: number of datums



A few key points for MAE

- It's more robust towards outliers than MAE since it doesn't exaggerate errors.
- It gives us a measure of how far the predictions were from the actual output. However, since MAE uses the absolute value of the residual, it doesn't give us an idea of the direction of the error, i.e. whether we're under-predicting or over-predicting the data.
- Error interpretation needs no second thoughts, as it perfectly aligns with the original degree of the variable.
- MAE is non-differentiable as opposed to MSE, which is differentiable.

Similar to MSE, this metric is also simple to implement.

```
mae = np.abs(y-y_hat)
print(f"MAE: {mae.mean():0.2f} (+/- {mae.std():0.2f})")
```

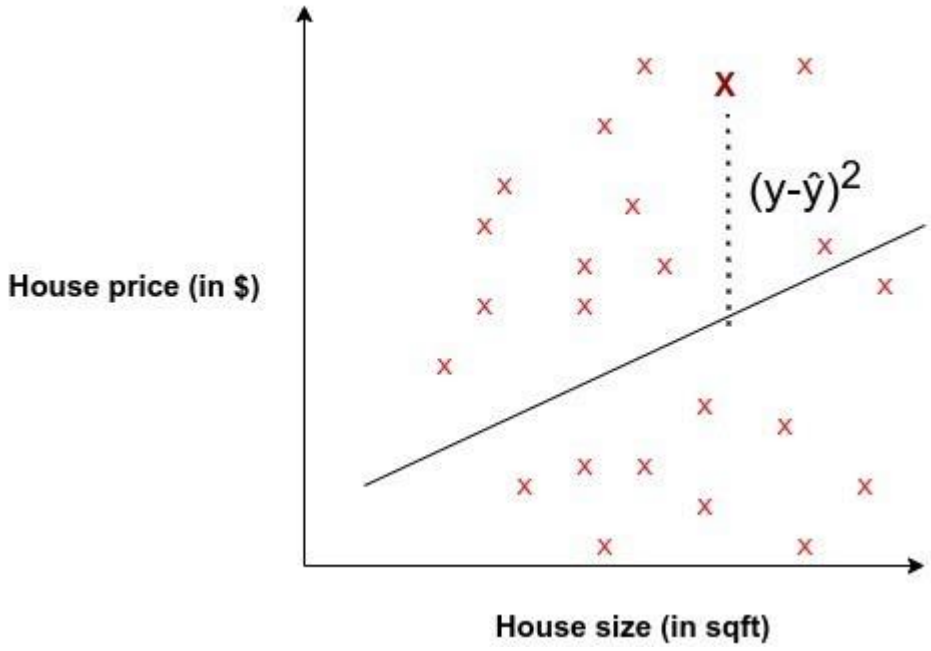
Mean Squared Error (MSE)

Mean squared error is perhaps the most popular metric used for regression problems. It essentially finds the average of the squared difference between the target value and the value predicted by the regression model.

$$MSE = \frac{1}{N}\sum_{j=1}^{N}(y_j - \check{y}_j)^2$$

Where:

- y_j: ground-truth value
- y_hat: predicted value from the regression model
- N: number of datums



A few key points related to MSE:

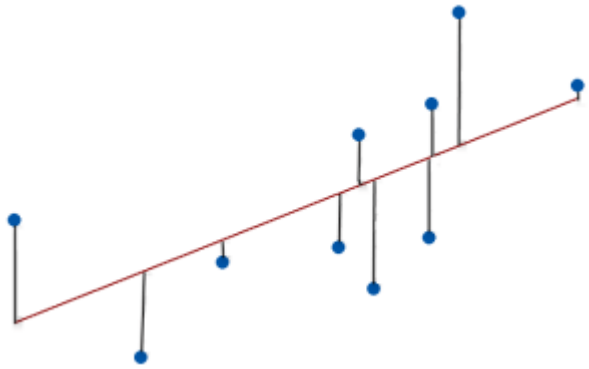
- It's differentiable, so it can be optimized better.
- It penalizes even small errors by squaring them, which essentially leads to an overestimation of how bad the model is.
- Error interpretation has to be done with the squaring factor(scale) in mind. For example, in our Boston Housing regression problem, we got MSE=21.89 which primarily corresponds to $(\text{Prices})^2$.
- Due to the squaring factor, it's fundamentally more prone to outliers than other metrics.

This can be implemented simply using NumPy arrays in Python.

```
me = (y-y_hat)**2
print(f"MSE: {mse.mean():0.2f} (+/- {mse.std():0.2f})")
```

Root Mean Squared Error (RMSE)

- The root mean square error (RMSE) measures the average difference between a statistical model's predicted values and the actual values. Mathematically, it is the standard deviation of the residuals. Residuals represent the distance between the regression line and the data points.
- RMSE quantifies how dispersed these residuals are, revealing how tightly the observed data clusters around the predicted values.



- As the data points move closer to the regression line, the model has less error, lowering the RMSE. A model with less error produces more precise predictions.
- RMSE values can range from zero to positive infinity and use the same units as the dependent (outcome) variable.
- Use the root mean square error to assess the amount of error in a regression or other statistical model. A value of 0 means that the predicted values perfectly match the actual values, but you'll never see that in practice. Low RMSE values indicate that the model fits the data well and has more precise predictions. Conversely, higher values suggest more error and less precise predictions.
- The root mean square error is a non-standardized goodness-of-fit assessment corresponding to its standardized counterpart—R-squared.
- The RSME formula should look familiar because it is essentially the standard deviation formula. That makes sense because the root mean

square error is the standard deviation of the residuals. It measures the scatter of the observed values around the predicted values.

- The RSME formula for a sample is the following:

$$RSME = \sqrt{\frac{\sum(y_i - \hat{y}_i)^2}{N - P}}$$

yi is the actual value for the ith observation.

ŷi is the predicted value for the ith observation.

N is the number of observations.

P is the number of parameter estimates, including the constant.

Mean Absolute Percentage Error (MAPE)

Mean absolute percentage error (MAPE) is a metric that measures the accuracy of a forecasting method. It's also known as mean absolute percentage deviation (MAPD).

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\frac{|A_i - F_i|}{A_i}$$

A_i is the actual value

F_i is the forecast value

n is total number of observations

- MAPE is calculated by averaging the absolute percentage errors of each entry in a dataset. It's expressed as a ratio, where At is the actual value and Ft is the forecast value.
- MAPE is one of the most commonly used KPIs to measure forecast accuracy. A MAPE value of 20% means that the average absolute percentage difference between the predictions and the actuals is 20%.

- MAPE has some limitations:
    - It can't be used when the actual values have instances of zero.
    - MAPE can sometimes favor models that under-forecast.

R2 Score

- The R2 score, also known as the coefficient of determination, is a statistical measure that evaluates how well a model fits a dataset. It is used to evaluate the performance of a linear regression model.
- The R2 score is calculated by:
    - Subtracting the average actual value from each of the actual values
    - Squaring the results
    - Summing the results
    - Dividing the first sum of errors (unexplained variance) by the second sum (total variance)
    - Subtracting the result from one
- The R2 score takes a value between 0 and 1. A value of 1 indicates that the model perfectly fits the data.

$$R^2 = 1 - \frac{SS_{RES}}{SS_{TOT}} = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \overline{y})^2}$$

- The R2 score can also be defined as:
    - The proportion of the variance in the dependent variable that is predictable from the independent variable(s)
    - (total variance explained by model) / total variance

2.4 Improving the Performance of a Model.

Here are some ways to improve the performance of a model:

- Training, testing, and data validation: These are essential steps in evaluating model performance.

- Choose a robust algorithm: Algorithms are the key factor used to train machine learning (ML) models.
- Improve data: Improving the quality and quantity of training data can provide a more robust model performance.
- Feature selection: Use the Correlation Feature Selection method to determine which features influence the output.
- Benchmarking: Compare models on common datasets and evaluation metrics to identify the most suitable model for a particular task.
- Interpret your model accuracy score: Identify what your model struggles to extract.

Other ways to improve model performance include:

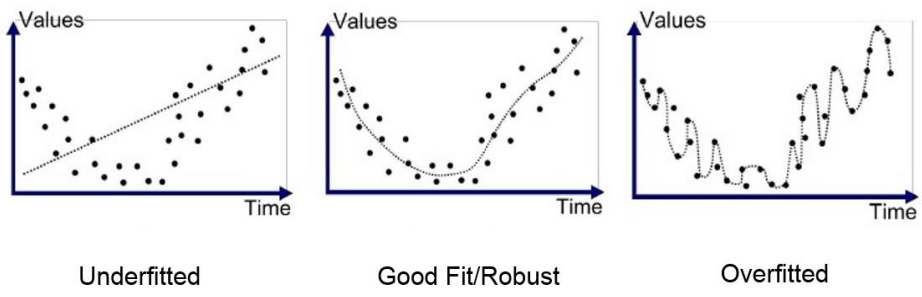
- Operations workflow
- Ensemble learning
- Supervised or unsupervised ML
- Anonymizing data
- Sampling data from large data sets
- Reducing dimensions
- Randomly shifting and rotating existing images

2.4.1 Under-fitting and Over-fitting

- **Under-fitting**

  A data model that can't accurately capture the relationship between input and output variables. Under-fitting causes a high error rate on both the training set and unseen data.

- **Over-fitting**

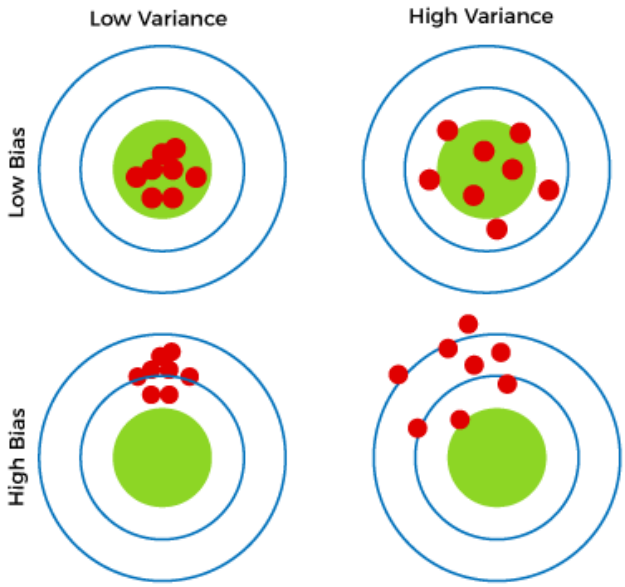


- A modeling error that occurs when a function is too closely aligned to a limited set of data points. Over-fitting causes the model to only be useful in reference to its initial data set.

2.6 Bias–variance tradeoff

What is bias?

- Bias is the difference between the average prediction of our model and the correct value which we are trying to predict. Model with high bias pays very little attention to the training data and oversimplifies the model. It always leads to high error on training and test data.



What is variance?

- Variance is the variability of model prediction for a given data point or a value which tells us spread of our data. Model with high variance pays a lot of attention to training data and does not generalize on the data which it hasn't seen before. As a result, such models perform very well on training data but has high error rates on test data.

Why is Bias Variance Tradeoff?

- If our model is too simple and has very few parameters then it may have high bias and low variance. On the other hand, if our model has large

number of parameters then it’s going to have high variance and low bias. So we need to find the right/good balance without over-fitting and under-fitting the data.

- This trade off in complexity is why there is a trade off between bias and variance. An algorithm can’t be more complex and less complex at the same time.
- Total Error

  To build a good model, we need to find a good balance between bias and variance such that it minimizes the total error.