DS JULY 2022 Batch
Module 13 – Machine learning Fundamentals

Topics

- Types of Machine Learning Methods
- Classification problem in general
- Validation Techniques: CV,OOB
- Curse of dimensionality
- Feature Selection
- Imbalanced Dataset and its effect on Classification
- Different types of metrics for Classification
- Bias Variance Trade-off

Types of Machine learning Methods

Types of Machine learning methods

Machine learning algorithms

Machine Learning algorithms are the programs that can learn the hidden patterns from the data, predict the output, and improve the performance from experiences on their own.

Types of Machine Learning Algorithms

Machine Learning Algorithm can be broadly classified into three types:

1.Supervised Learning Algorithms

2.Unsupervised Learning Algorithms

3.Reinforcement Learning algorithms

Supervised learning algorithms

Supervised learning algorithms

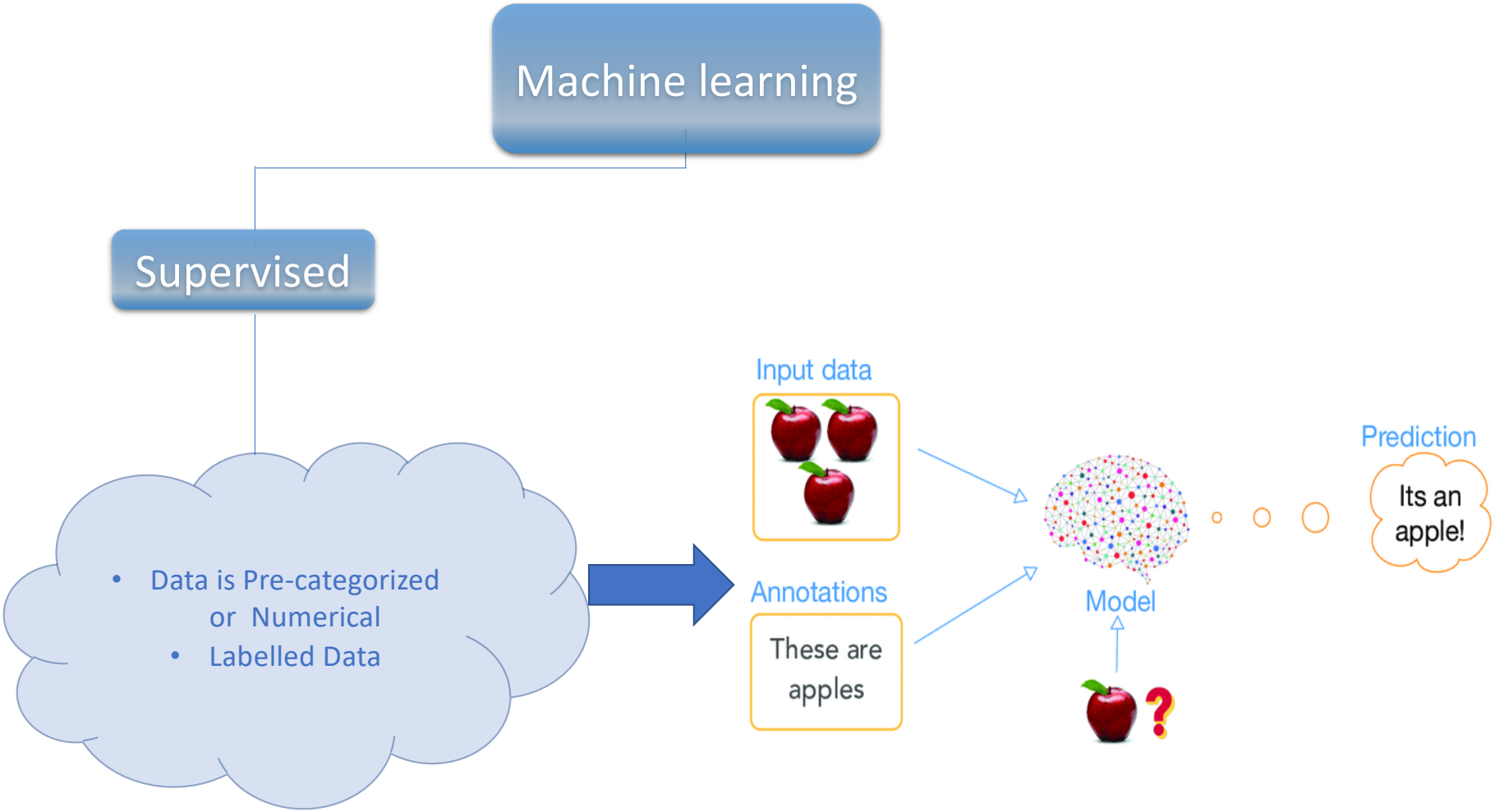
- Supervised learning is a type of Machine learning in which the machine needs external supervision to learn.
- The supervised learning models are trained using the labelled dataset.
- Once the training and processing are done, the model is tested by providing a sample test data to check whether it predicts the correct output.

Supervised learning can be divided further into two categories of problem:

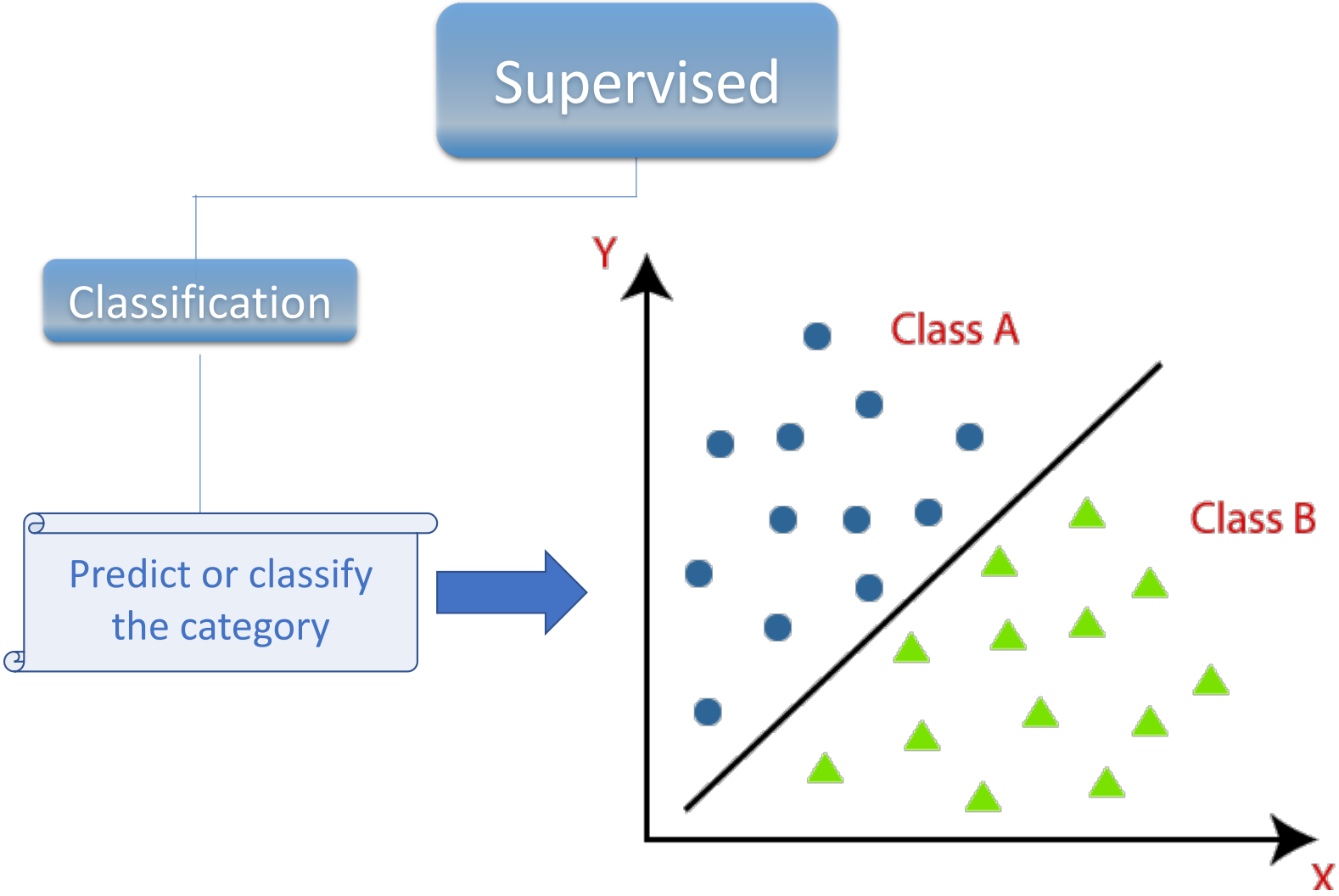
- **Classification problem** : The Classification algorithm is a Supervised Learning technique that is used to identify the category of new observations on the basis of training data. In Classification, a program learns from the given dataset or observations and then classifies new observation into a number of classes or groups. Such as, Yes or No, 0 or 1, Spam or Not Spam, cat or dog, etc. Classes can be called as targets/labels or categories.

- **Regression problem :** Regression is a supervised machine learning algorithm used to predict the continuous values of output based on the input.

Supervised learning algorithms



Supervised learning algorithms

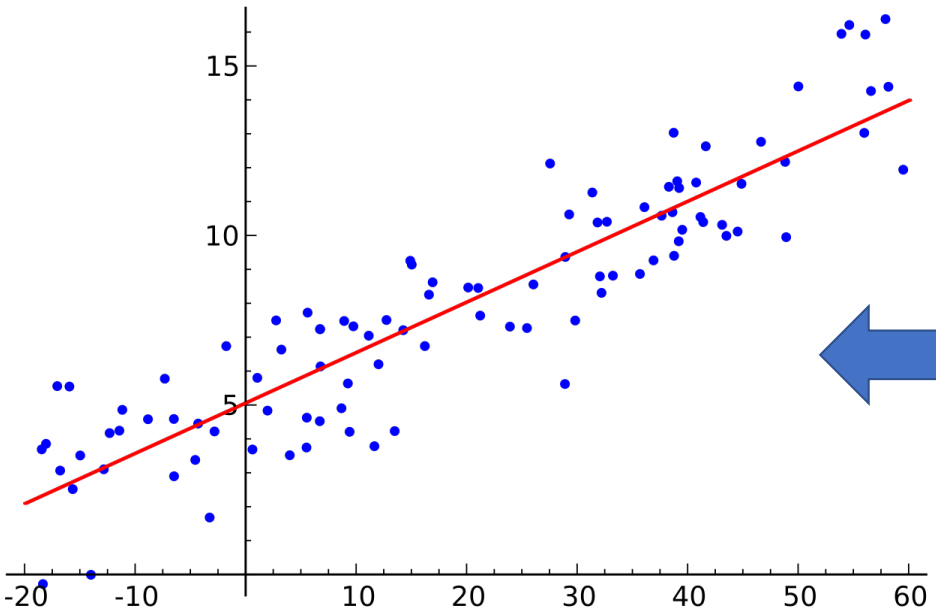


Supervised learning algorithms

Supervised

Regression

Predict a number



Unsupervised learning algorithms

Unsupervised learning algorithms

- It is a type of machine learning in which the machine does not need any external supervision to learn from the data, hence called unsupervised learning.
- The unsupervised models can be trained using the unlabelled dataset that is not classified, nor categorized, and the algorithm needs to act on that data without any supervision.
- In unsupervised learning, the model doesn't have a predefined output, and it tries to find useful insights from the huge amount of data. These are used to solve the Association and Clustering problems.

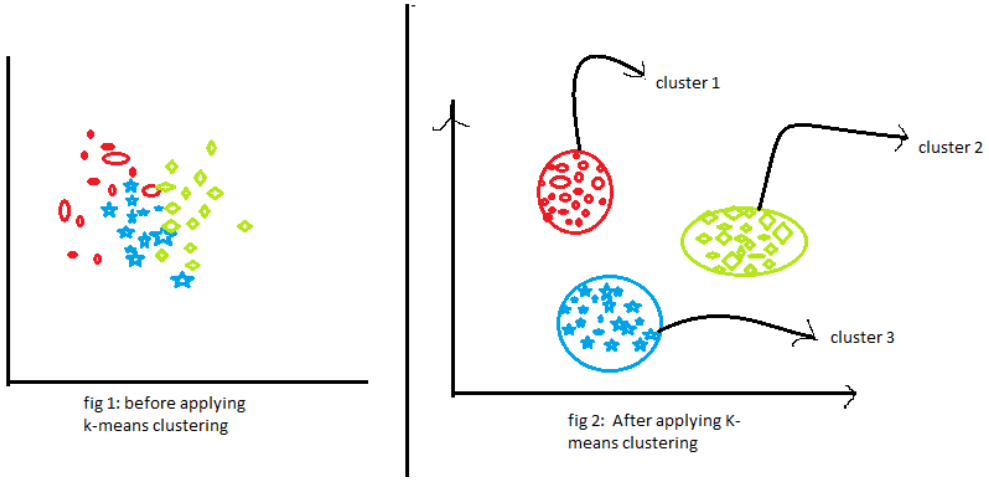
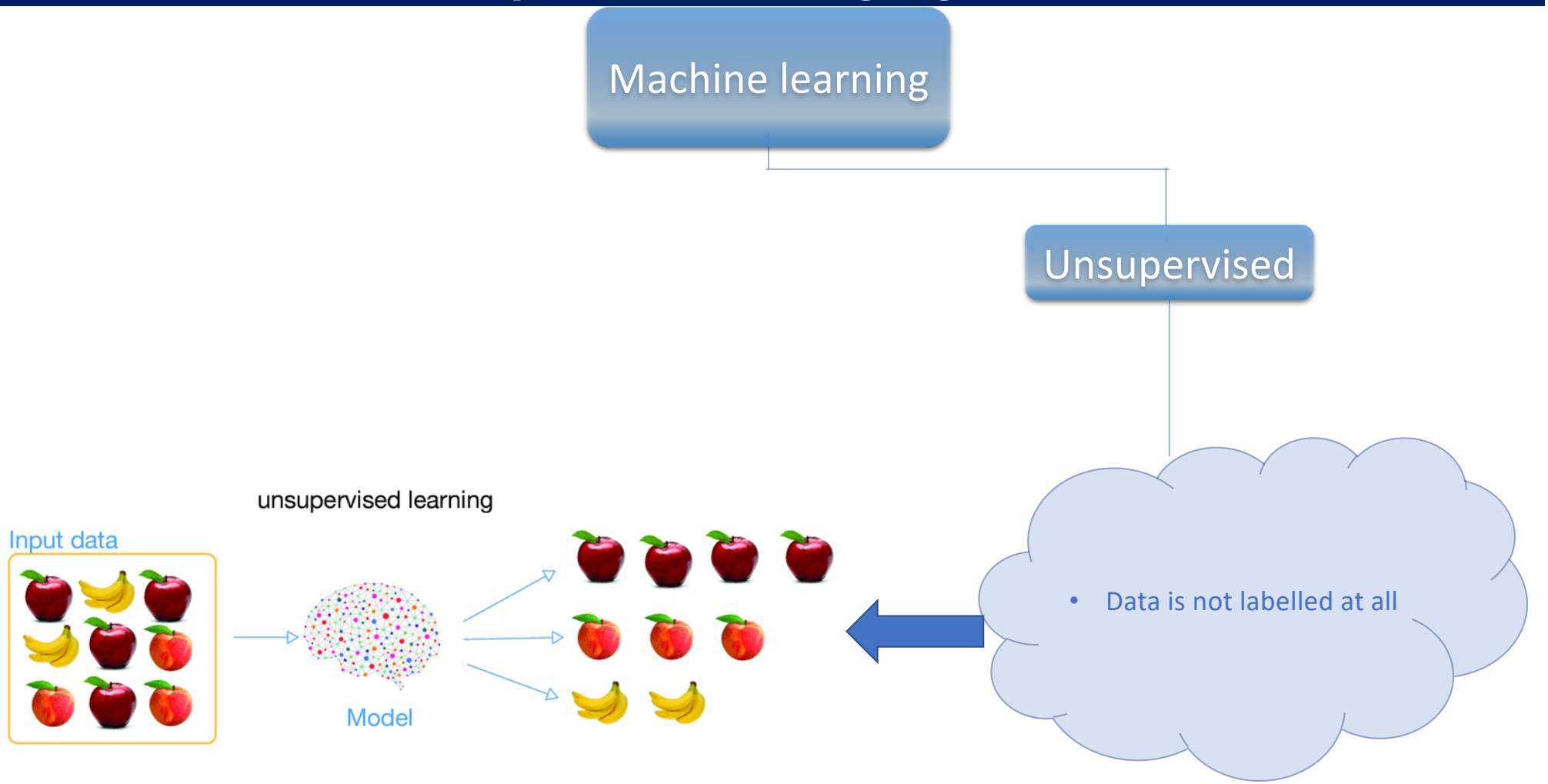


fig 1: before applying k-means clustering

fig 2: After applying K-means clustering

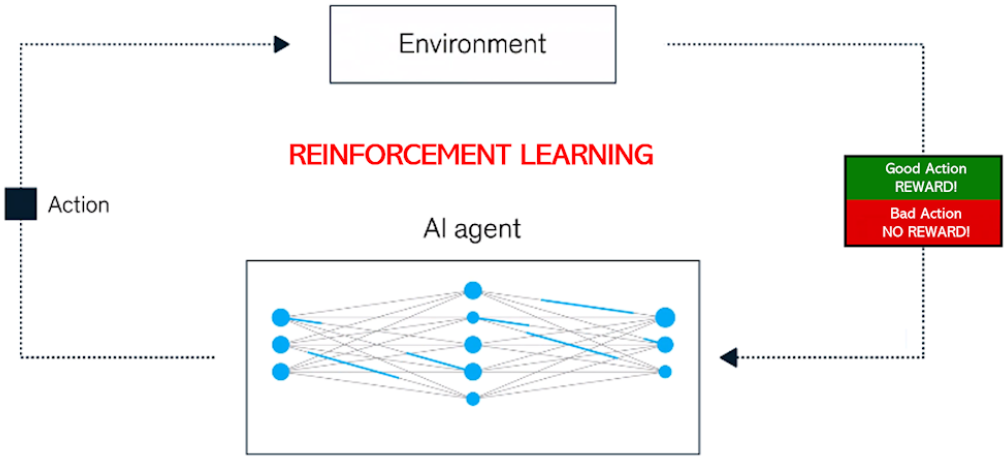
Unsupervised learning algorithms



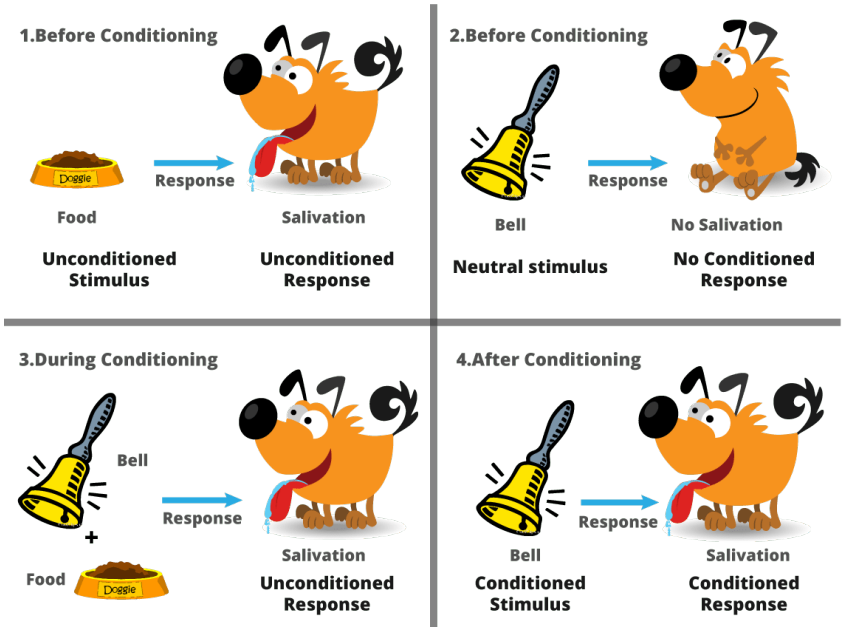
Reinforcement Learning algorithms

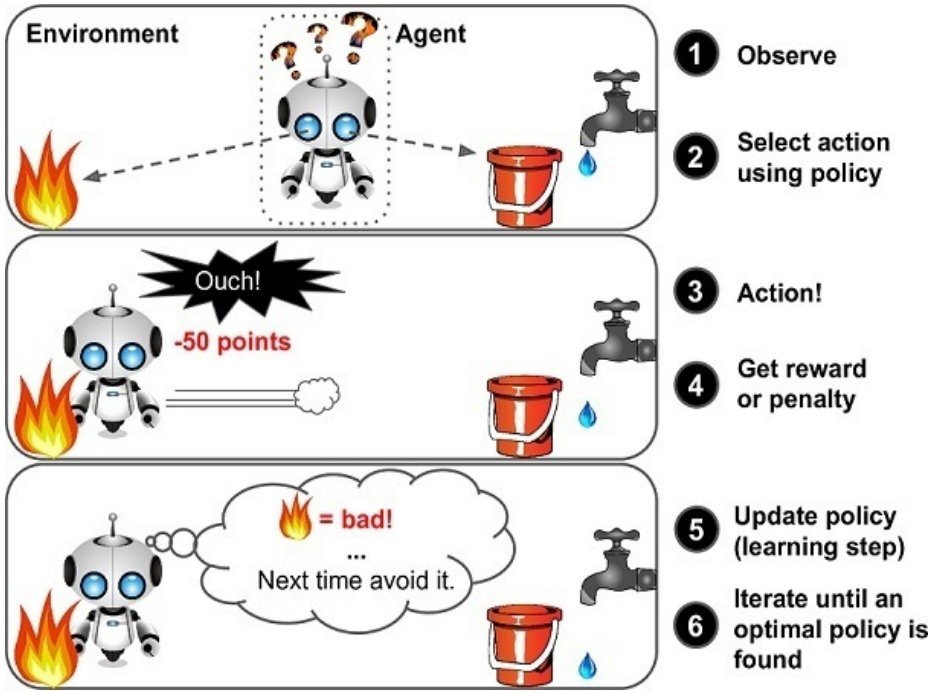
Reinforcement learning algorithms

- In Reinforcement learning, an agent interacts with its environment by producing actions, and learn with the help of feedback.
- The feedback is given to the agent in the form of rewards, such as for each good action, he gets a positive reward, and for each bad action, he gets a negative reward.
- There is no supervision provided to the agent.

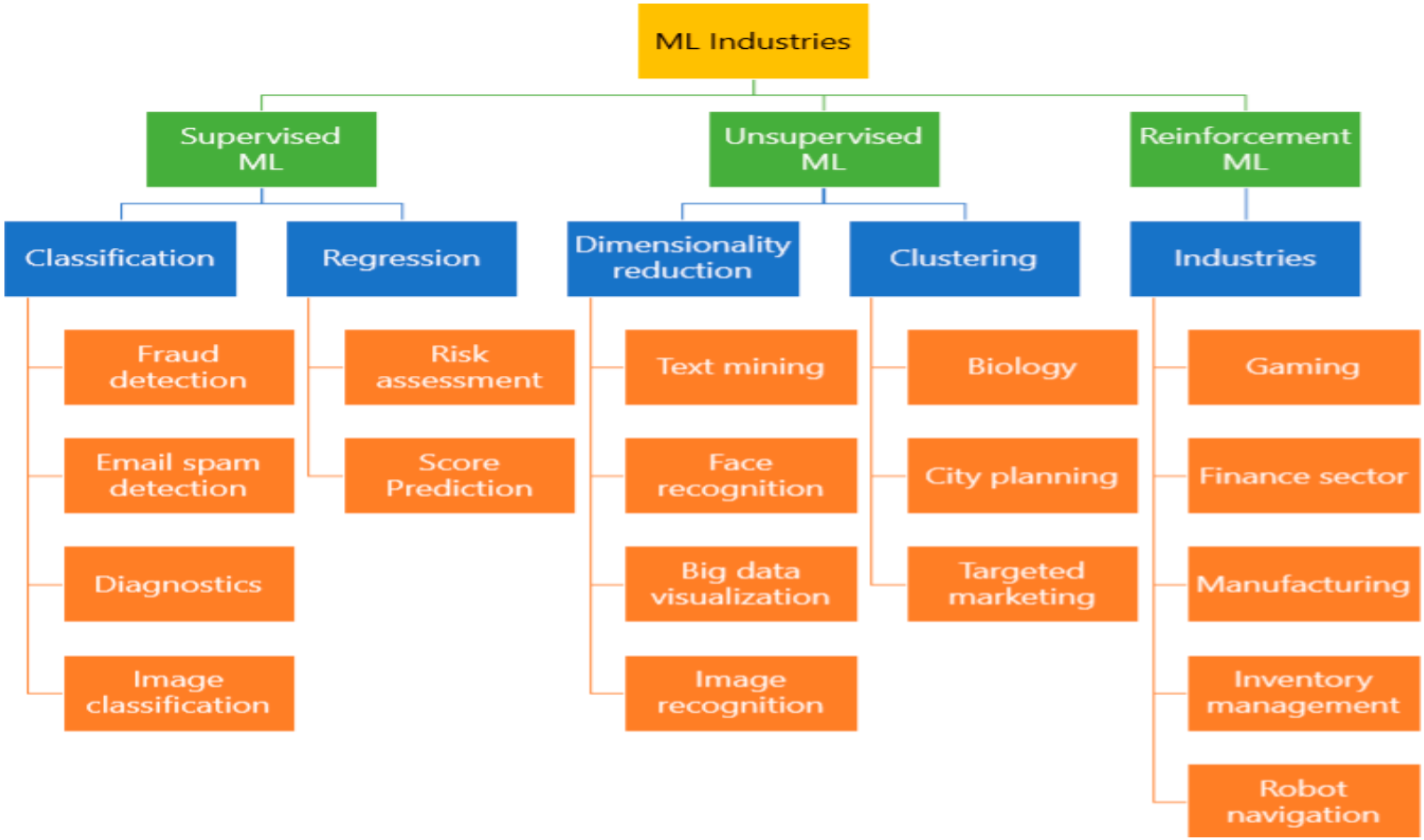


Reinforcement Learning algorithms





Summary


ML Industries
Supervised ML
Unsupervised ML
Reinforcement ML
Classification
Regression
Dimensionality reduction
Clustering
Industries
Fraud detection
Email spam detection
Diagnostics
Image classification
Risk assessment
Score Prediction
Text mining
Face recognition
Big data visualization
Image recognition
Biology
City planning
Targeted marketing
Gaming
Finance sector
Manufacturing
Inventory management
Robot navigation

Classification Problem

Classification problem

- Classification is defined as the process of recognition, understanding, and grouping of objects and ideas into preset categories.
- Based on training data, the Classification algorithm is a Supervised Learning technique used to categorize new observations. In classification, a program uses the dataset or observations provided to learn how to categorize new observations into various classes or groups.
- For instance, 0 or 1, red or blue, yes or no, spam or not spam, etc. Targets, labels, or categories can all be used to describe classes.
- The Classification algorithm uses labelled input data because it is a supervised learning technique and comprises input and output information. A discrete output function (y) is transferred to an input variable in the classification process (x).

Validation Techniques

Validation Techniques

- Whenever we build any machine learning model, we feed it with initial data to train the model. Then we feed some unknown data (test data) to understand how well the model performs and generalized over unseen data.
- If the model performs well on the unseen data, it's consistent and is able to predict with good accuracy on a wide range of input data; then this model is stable.
- But machine learning models are not always stable and we have to evaluate the stability of the machine learning model. That is where Cross Validation comes into the picture.

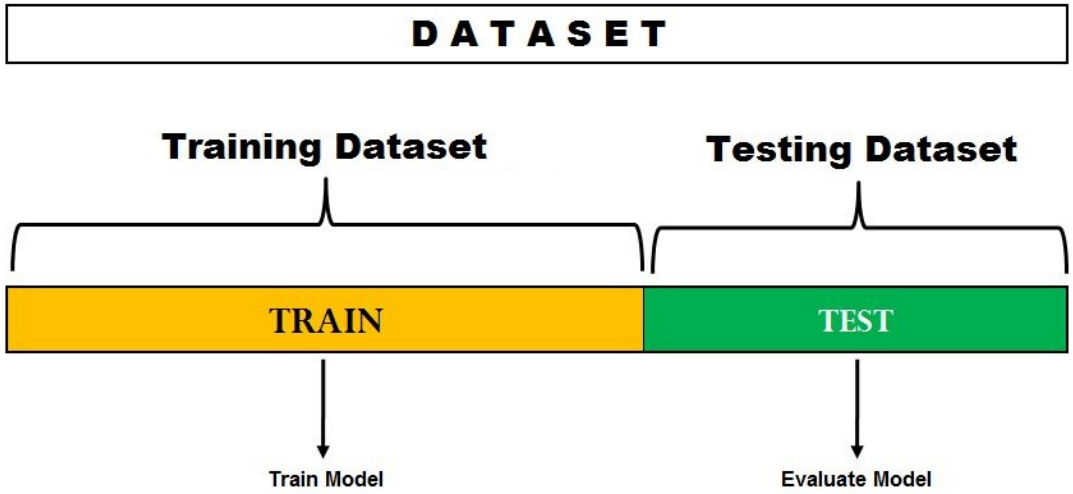
Cross Validation

Cross-Validation is a resampling technique with the **fundamental idea of splitting the dataset into 2 parts- training data and test data**. **Train data is used to train the model and the unseen test data is used for prediction**. If the **model performs well over the test data** and gives good accuracy, it means the model hasn't overfitted the training data and **can be used for prediction.**

Cross Validation Methods

Hold Out Method

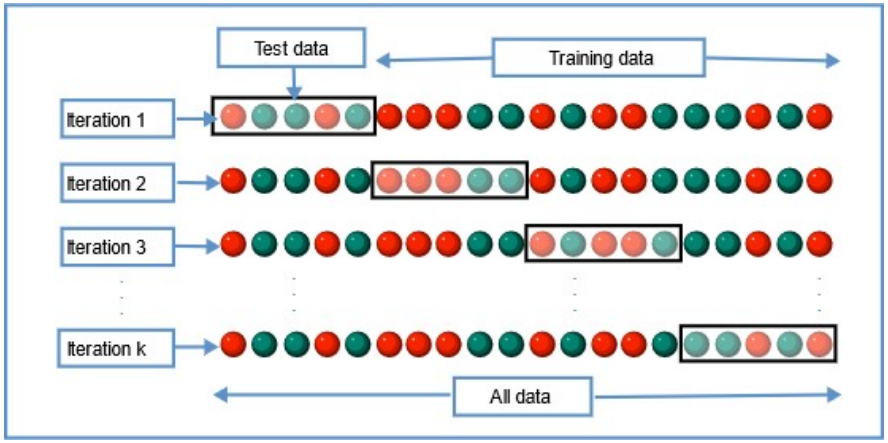
- This is the simplest evaluation method and is widely used in Machine Learning projects.
- Here the entire dataset(population) is **divided into 2 sets – train set and test set.**
- The data can be divided into 70-30 or 60-40, 75-25 or 80-20, or even 50-50 depending on the use case.
- As a rule, **the proportion of training data has to be larger than the test data.**
- The **data split happens randomly**, and we can't be sure which data ends up in the train and test bucket during the split **unless we specify random_state.**



Cross Validation Methods

K-Fold Cross-Validation

- In this resampling technique, the whole **data is divided into k sets** of almost equal sizes.
- The **first set is selected as the test** set and the **model is trained on the remaining k-1 sets.** The test error rate is then calculated after fitting the model to the test data.
- In the **second iteration**, the **2nd set is selected as a test set** and the **remaining k-1 sets are used to train** the data and the error is calculated. This **process continues for all the k sets.**
- Typically, K-fold Cross Validation is performed using **k=5 or k=10** as these values have been empirically shown to yield test error estimates that neither have high bias nor high variance.



$$\mathrm{CV}_{(k)} = \frac{1}{k}\sum_{i=1}^{k}\mathrm{MSE}_i.$$

Out of Bag Sample and Score

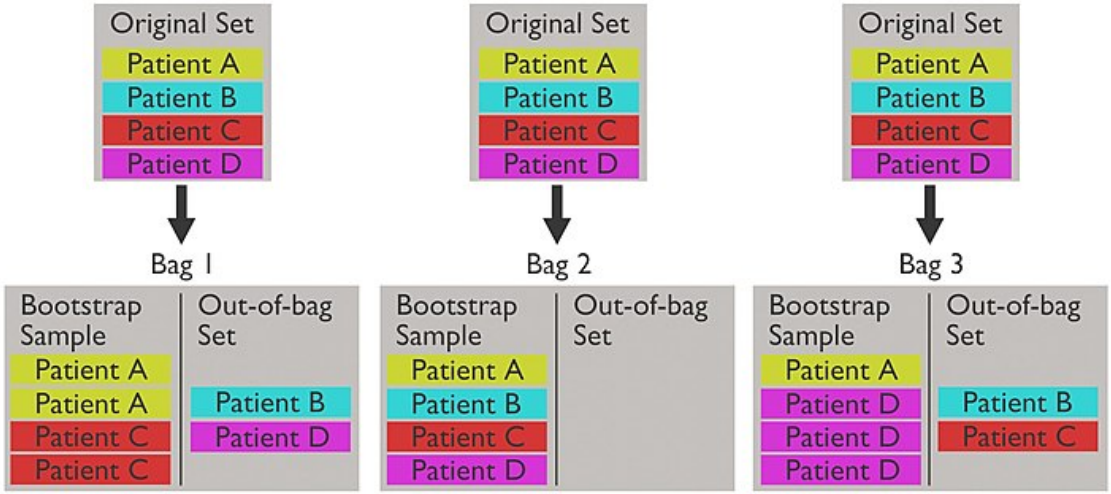
Bootstrap Sampling

- In statistics, Bootstrap Sampling is a method that involves drawing of sample data repeatedly with replacement from a data source to estimate a population parameter.

Out of Bag Sample

- In Cross Validation we **divide our dataset into training and testing data** and we validate our model on testing data but the testing data is still the part of model training process.
- In bagging technique such as Random forest we first divide our dataset into 2 parts : **train, test.**
- Among the datapoints in the train data some sample points are kept aside which is called as **OOB sample.**
- Model is trained on the train data and then tested on test data. Once the model is trained and tested then our model is tested on **OOB sample.**
- The **OOB score** is computed as the number of correctly predicted rows from the out-of-bag sample.
- **OOB Error** is the number of wrongly classifying the OOB Sample.

Out of Bag Sample and Score



Note

- In random forest we built multiple decision tree. Each of the above bags are representing bootstrap samples of respective decision trees.
- Generally about 1/3rd of the training data points are selected in OOB sample.

Curse of Dimensionality

Curse of Dimensionality

- The curse of dimensionality is a problem that arises when we are **working with a lot of data** having multiple features or we can say it as high dimensional data. The **dimension of the data** means the **no. of features or columns** in our dataset.

Drawbacks of high dimensional dataset:

- Becomes very challenging to identify meaningful patterns
- Visualizing the data
- Degrades the Machine Learning model's accuracy
- Training the model will become much slower

Need for Data Points with Increase in Dimensions

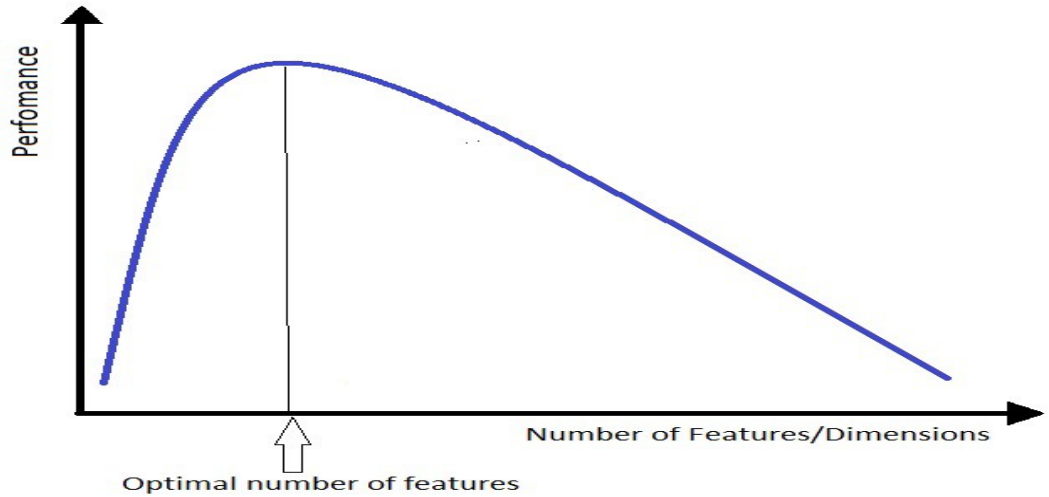
1 Binary feature →	2^1 unique values →	$2^1 \times 10 = 20$ data points
2 Binary features →	2^2 unique values →	$2^2 \times 10 = 40$ data points
3 Binary features →	2^3 unique values →	$2^3 \times 10 = 80$ data points
.	.	.
.	.	.
.	.	.
k Binary features →	2^k unique values →	$2^k \times 10$ data points

Model 1		
Sr No	f1_Yes	f1_No
1	0	1
2	0	1
3	1	0
4	1	0
5	0	1
6	0	1
7	1	0
8	0	1
9	1	0
10	1	0

Model 2				
Sr No	f1_Yes	f1_No	f2_Yes	f2_No
1	0	1	0	1
2	0	1	0	1
3	1	0	1	0
4	1	0	1	0
5	0	1	0	1
6	0	1	0	1
7	1	0	1	0
8	0	1	0	1
9	1	0	1	0
10	1	0	1	0

Hughes phenomenon

- Hughes (1968) in his study concluded that with a fixed number of training samples, the predictive power of any classifier first increases as the number of dimensions increase, but after a certain value of number of dimensions, the performance deteriorates.
- Thus, the phenomenon of curse of dimensionality is also known as Hughes phenomenon.



Remedies for high dimensional data

- With the increase in the no. of dimensions, it becomes very cumbersome to calculate distance between observations as well
- All the Machine Learning algorithm that relies on calculating distance between observations find it very troublesome to work with high dimensional data, such as segmentation and clustering algorithms like KNN, K-Means etc.

Dimensionality Reduction

Dimensionality reduction is the transformation of data from a high-dimensional space into a low-dimensional space so that the low-dimensional representation retains some meaningful properties of the original data, ideally close to its natural dimension or in simple words, it **means reducing the dimensions of our data set.**

Dimensionality Reduction can be divided into two types, **Feature Selection** and **Feature Extraction.**

Feature Extraction

There are two popular methods of feature extraction : PCA and t-SNE

PCA — Principal Component Analysis

PCA is a **linear dimensionality reduction algorithm** that helps us in extracting a new set of variables from an existing large set of variables which are called **Principal Components**.

It finds the best linear combinations of the original variables so that the variance or spread along the new variable is maximum.

We will see PCA in details in module (Unsupervised learning Part -2)

t-SNE t-Distributed Stochastic Neighbor Embedding

It is a **non-linear dimensionality reduction algorithm** used for exploring high-dimensional data.

It finds patterns in the data based on the similarity of data points with features.

The similarity of points is calculated as the conditional probability that a point A would choose point B as its neighbour.

Feature Selection

Feature Selection is selecting only the important features or eliminating the less useful features from our dataset.

Remove those feature which has a huge no. of missing values: (more than 50–60%) because it will not have sufficient information to provide us. We can set a particular threshold value, and if the missing values are more than this threshold then we will remove all those features.

Remove the features having a high correlation : A high correlation between 2 variables means that both are carrying similar information and dropping either of the two will not have that much impact on our outcome. We can find the correlation of our data frame using the corr() function of pandas. And we can set a threshold, and if the correlation coefficient crosses that threshold, we can drop it.

Feature Selection using Random Forest — Random Forrest has an inbuilt feature of providing feature importance metric but before that, we need to convert our data into numerical form as Random Forrest takes only numeric inputs. (Note- Drop the ID variables and Target variables as well)

Feature Selection

Feature Selection

The goal of feature selection in machine learning is to find the best set of features that allows one to build useful models of studied phenomena.

The techniques for feature selection in machine learning can be broadly classified into the following categories:

Supervised Techniques: These techniques can be used for labelled data, and are used to identify the relevant features for increasing the efficiency of supervised models like classification and regression.

Unsupervised Techniques: These techniques can be used for unlabelled data.

From a taxonomic point of view, these techniques are classified as under:

A. Filter methods
B. Wrapper methods
C. Embedded methods
D. Hybrid methods

Filter Methods

Filter methods pick up the intrinsic properties of the features measured via univariate statistics instead of cross-validation performance. When dealing with high-dimensional data, it is computationally cheaper to use filter methods.

1) Information Gain :

Information gain calculates the reduction in entropy from the transformation of a dataset. It can be used for feature selection by evaluating the Information gain of each variable in the context of the target variable.

2) Chi-square Test

The Chi-square test is used for categorical features in a dataset. We calculate Chi-square between each feature and the target and select the desired number of features with the best Chi-square scores.

3) Fisher's Score

Fisher score is one of the most widely used supervised feature selection methods. The algorithm which we will use returns the ranks of the variables based on the fisher's score in descending order. We can then select the variables as per the case.

4) Correlation Coefficient

The logic behind using correlation for feature selection is that the good variables are highly correlated with the target.

Furthermore, variables should be correlated with the target but should be uncorrelated among themselves.

Filter Methods

5) Variance Threshold:

Variance Threshold method removes all features which variance doesn't meet some threshold. By default, it removes all zero-variance features, i.e., features that have the same value in all samples. We assume that features with a higher variance may contain more useful information

Wrapper Methods

- Wrappers require some method to search the space of all possible subsets of features, assessing their quality by learning and evaluating a classifier with that feature subset.
- The feature selection process is based on a specific machine learning algorithm that we are trying to fit on a given dataset.
- It follows a greedy search approach by evaluating all the possible combinations of features against the evaluation criterion. The wrapper methods usually result in better predictive accuracy than filter methods.

1) Forward Feature Selection

This is an iterative method wherein we start with the best performing variable against the target. Next, we select another variable that gives the best performance in combination with the first selected variable. This process continues until the preset criterion is achieved.

2) Backward Feature Elimination

This method works exactly opposite to the Forward Feature Selection method. Here, we start with all the features available and build a model. Next, we remove the variable from the model which deteriorates evaluation measure value. This process is continued until the preset criterion is achieved.

Wrapper Methods

3) Exhaustive Feature Selection

This is the most robust feature selection method covered so far. This is a brute-force evaluation of each feature subset. This means that it tries every possible combination of the variables and returns the best performing subset.

4) Recursive Feature Elimination

'Given an external estimator that assigns weights to features (e.g., the coefficients of a linear model), the goal of recursive feature elimination (RFE) is to select features by recursively considering smaller and smaller sets of features. First, the estimator is trained on the initial set of features and the importance of each feature is obtained either through a coef_ attribute or through a feature_importances_ attribute.

Then, the least important features are pruned from the current set of features. That procedure is recursively repeated on the pruned set until the desired number of features to select is eventually reached.'

Embedded Methods

- These methods encompass the benefits of both the wrapper and filter methods, by including interactions of features but also maintaining reasonable computational cost.
- Embedded methods are iterative in the sense that takes care of each iteration of the model training process and carefully extracts those features which contribute the most to the training for a particular iteration.

1) LASSO Regularization (L1)

Regularization consists of adding a penalty to the different parameters of the machine learning model to reduce the freedom of the model, i.e. to avoid over-fitting.
In linear model regularization, the penalty is applied over the coefficients that multiply each of the predictors.
From the different types of regularization, Lasso or L1 has the property that is able to shrink some of the coefficients to zero. Therefore, that feature can be removed from the model.

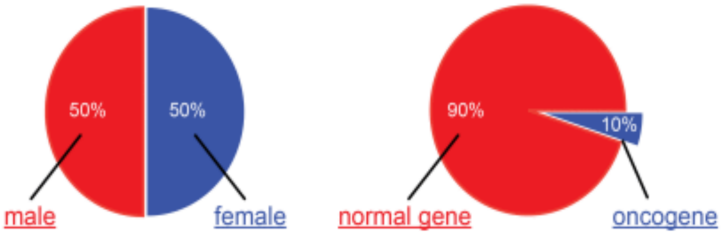
2) Random Forest Importance

Random Forests is a kind of a Bagging Algorithm that aggregates a specified number of decision trees. The tree-based strategies used by random forests naturally rank by how well they improve the purity of the node, or in other words a decrease in the impurity (Gini impurity) over all trees.
Nodes with the greatest decrease in impurity happen at the start of the trees, while notes with the least decrease in impurity occur at the end of trees. Thus, by pruning trees below a particular node, we can create a subset of the most important features.

Imbalanced Dataset and its effect on Classification

Imbalanced Dataset

- Imbalanced classification is defined by a dataset with a **skewed target class distribution**.
- This is often exemplified by a binary (two-class) classification task where most of the examples belong to class 0 with only a few examples in class 1. The distribution may range in severity from 1:2, 1:10, 1:100, or even 1:1000.



Class Imbalance appear in many domains, including:

- Fraud detection
- Spam filtering
- Disease screening

Effect of imbalanced dataset on classification

- Most machine learning algorithms work best when the number of samples in each class are about equal. This is because most algorithms are designed to maximize accuracy and reduce errors.
- when the class imbalance is high, e.g. 90% points for one class and 10% for the other, standard optimization criteria or performance measures may not be as effective and would need modification.
- In such cases, you get a pretty high accuracy just by predicting the majority class, but you fail to capture the minority class, which is most often the point of creating the model in the first place..

Ways to deal with Imbalanced Dataset

Resampling Technique

A widely adopted technique for dealing with highly unbalanced datasets is called resampling. It consists of removing samples from the majority class (under-sampling) and/or adding more examples from the minority class (over-sampling).

1. Random Under-Sampling

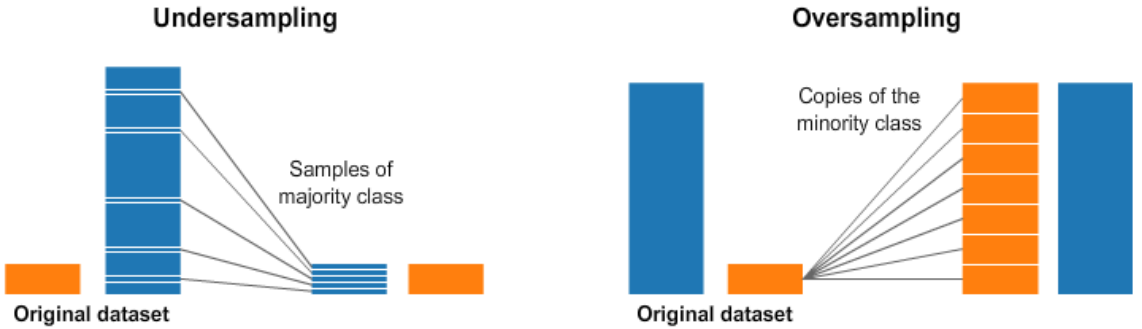
Undersampling can be defined as **removing some observations of the majority class.** This is done until the majority and minority class is balanced out.

Undersampling can be a good choice when you have a ton of data -think millions of rows. But a drawback to undersampling is that we are removing information that may be valuable.

2. Random Over-Sampling

Oversampling can be defined as **adding more copies to the minority class.** Oversampling can be a good choice when you don't have a ton of data to work with.

A con to consider when oversampling is that it can cause overfitting and poor generalization to your test set.



Ways to deal with Imbalanced Dataset

Balance data with the imbalanced-learn python module

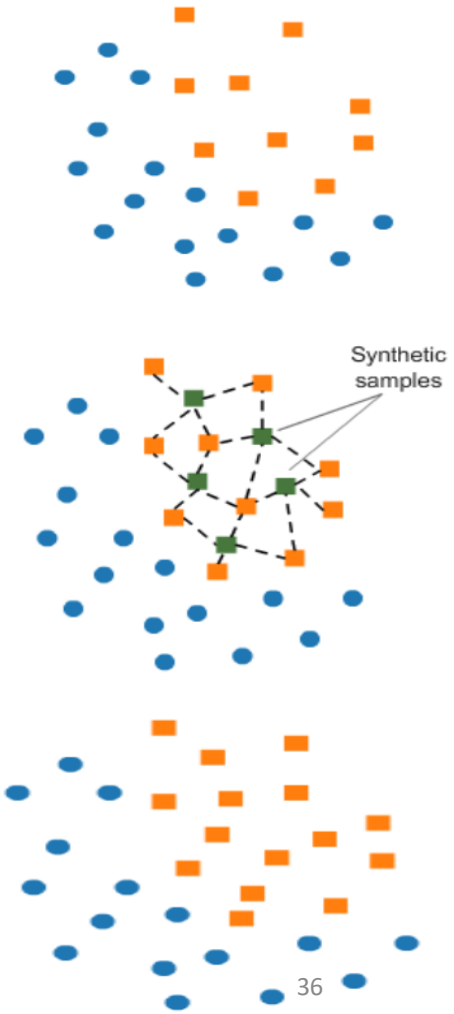
Imbalanced-Learn is a Python module that helps in balancing the datasets which are highly skewed or biased towards some classes.
Thus, it helps in resampling the classes which are otherwise oversampled or undesampled.

Synthetic Minority Oversampling Technique (SMOTE)

This technique generates synthetic data for the minority class.
SMOTE (Synthetic Minority Oversampling Technique) works by randomly picking a point from the minority class and computing the k-nearest neighbors for this point.
The **synthetic points are added** between the chosen point and its neighbors.

SMOTE algorithm works in 4 simple steps:

1.Choose a minority class as the input vector
2.Find its k nearest neighbors (**k_neighbors** is specified as an argument in the **SMOTE()** function)
3. Choose one of these neighbors and place a synthetic point anywhere on the line joining the point under consideration and its chosen neighbor
4. Repeat the steps until data is balanced



Different types of metrics for Classification

Metrics for classification

The most important task in building any machine learning model is to evaluate its performance.

the question arises that how would one measure the success of a machine learning model?

- Evaluation metrics helps us to asses the performance of machine learning algorithms.
- The metrics that you choose to evaluate your machine learning model is very important.
- Choice of metrics influences how the performance of machine learning algorithms is measured and compared.

The following metrics will help us to evaluate a classification model

- **Confusion Matrix**
- **Accuracy**
- **Precision**
- **Recall or Sensitivity**
- **Specificity**
- **F1 Score**
- **AUC-ROC**

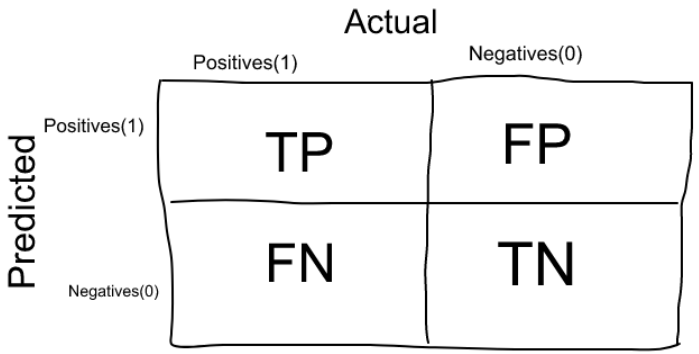
Confusion matrix

- The Confusion matrix is one of the most intuitive and easiest (unless of course, you are not confused)metrics used for finding the correctness and accuracy of the model.
- It is used for Classification problem where the output can be of two or more types of classes.

Terms associated with Confusion matrix:

1.True Positives (TP): True positives are the cases when the actual class of the data point was 1(True) and the predicted is also 1(True)

Ex: The case where a person is actually having cancer(1) and the model classifying his case as cancer(1) comes under True positive.



2. True Negatives (TN): True negatives are the cases when the actual class of the data point was 0(False) and the predicted is also 0(False)

Ex: The case where a person NOT having cancer and the model classifying his case as Not cancer comes under True Negatives.

3. False Positives (FP): False positives are the cases when the actual class of the data point was 0(False) and the predicted is 1(True). False is because the model has predicted incorrectly and positive because the class predicted was a positive one. (1)

Ex: A person NOT having cancer and the model classifying his case as cancer comes under False Positives.

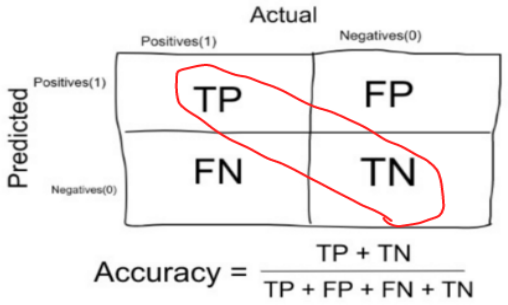
4. False Negatives (FN): False negatives are the cases when the actual class of the data point was 1(True) and the predicted is 0(False). False is because the model has predicted incorrectly and negative because the class predicted was a negative one. (0)

Ex: A person having cancer and the model classifying his case as No-cancer comes under False Negatives.

Accuracy, Precision and Recall

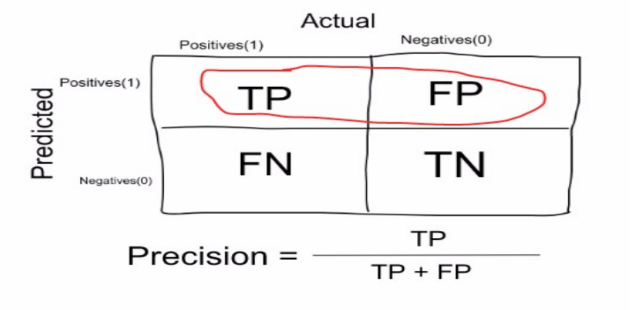
Accuracy

- Accuracy in classification problems is the number of correct predictions made by the model over all kinds predictions made.
- Accuracy is a good measure when the target variable classes in the data are nearly balanced.
- Accuracy should NEVER be used as a measure when the target variable classes in the data are a majority of one class.



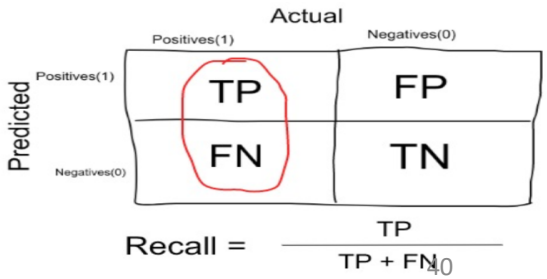
Precision

- The precision determines the proportion of positive prediction that was actually correct.
- It can be calculated as the True Positive or predictions that are actually true to the total positive predictions (True Positive and False Positive).



Recall or Sensitivity

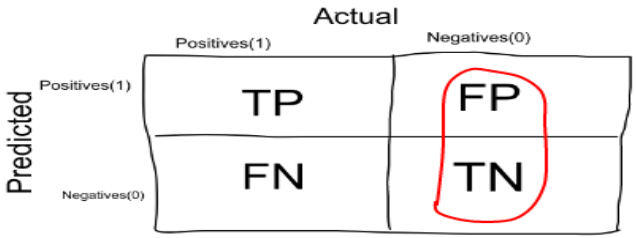
- It is also similar to the Precision metric; however, it aims to calculate the proportion of actual positive that was identified incorrectly.
- It can be calculated as True Positive or predictions that are actually true to the total number of positives, either correctly predicted as positive or incorrectly predicted as negative (true Positive and false negative).



Specificity and F1 Score

Specificity

- It can be calculated as True Negatives or predictions that are actually False to the total number of negatives, either correctly predicted as negatives or incorrectly predicted as positives (true negative and false positive).
- Specificity is the exact opposite of Recall.



$$\text{Specificity} = \frac{TN}{TN + FP}$$

F1 Score

- F1 Score is a metric to evaluate a binary classification model on the basis of predictions that are made for the positive class.
- It is calculated with the help of Precision and Recall. It is a type of single score that represents both 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.

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

AUC-ROC

- AUC (Area Under Curve)-ROC (Receiver Operating Characteristic) is a performance metric, based on varying threshold values, for classification problems. As name suggests, ROC is a probability curve and AUC measure the separability. In simple words, AUC-ROC metric will tell us about the capability of model in distinguishing the classes. Higher the AUC, better the model.
- Mathematically, it can be created by plotting TPR (True Positive Rate) i.e. Sensitivity or recall vs FPR (False Positive Rate) i.e. 1-Specificity, at various threshold values.
- AUC calculates the performance across all the thresholds and provides an aggregate measure.
- The value of AUC ranges from 0 to 1.
- It means a model with 100% wrong prediction will have an AUC of 0.0, whereas models with 100% correct predictions will have an AUC of 1.0.

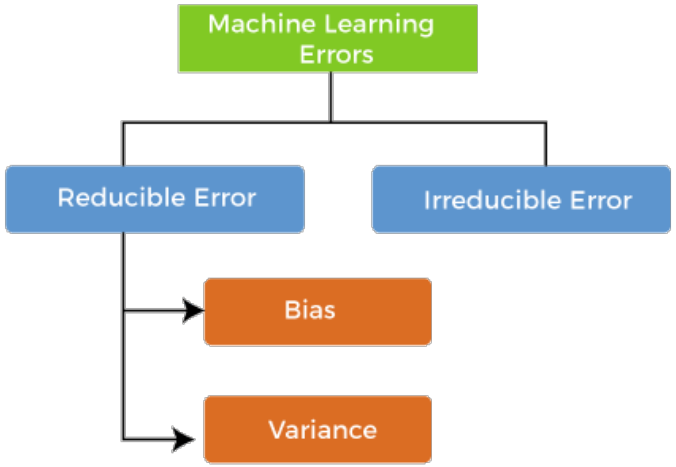
Bias Variance Trade-off

Errors in Machine learning

In machine learning, an error is a measure of how accurately an algorithm can make predictions for the previously unknown dataset. On the basis of these errors, the machine learning model is selected that can perform best on the particular dataset. There are mainly two types of errors in machine learning, which are:

- **Reducible errors:** These errors can be reduced to improve the model accuracy. Such errors can further be classified into **bias and Variance**.

- **Irreducible errors:** These errors will always be present in the model regardless of which algorithm has been used. The cause of these errors is unknown variables whose value can't be reduced.



Bias

- While training, the model learns the patterns in the dataset and applies them to test data for prediction.
- While making predictions, a difference occurs between prediction values made by the model and actual values/expected values, and this difference is known as bias errors or Errors due to bias.
- It can be defined as an inability of machine learning algorithms such as Linear Regression to capture the true relationship between the data points.

A model has either:

- **Low Bias:** A low bias model will make fewer assumptions about the form of the target function.
- **High Bias:** A model with a high bias makes more assumptions, and the model becomes unable to capture the important features of our dataset. **A high bias model also cannot perform well on new data.**
  When the model has high bias we call that model as underfitted model

Ways to reduce High Bias:

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.
- Decrease the regularization term.
- Use more complex models, such as including some polynomial features.

Variance Error

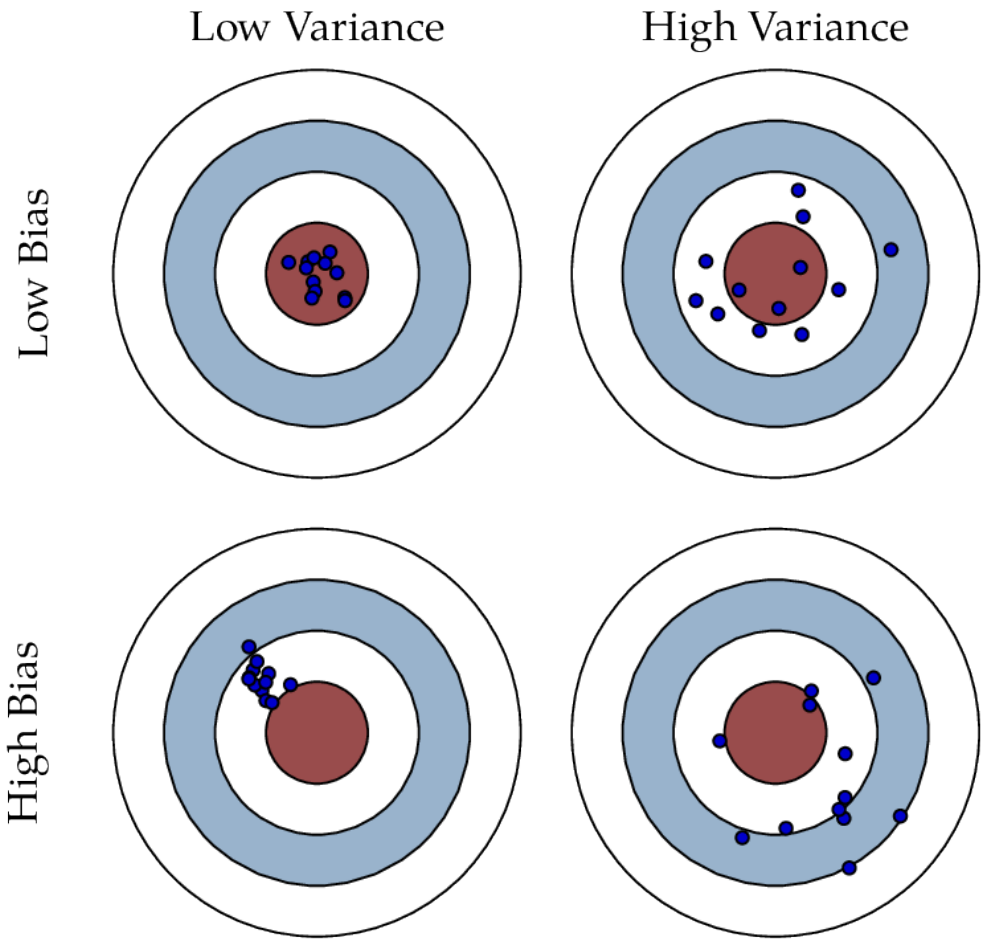
- The variance would specify the amount of variation in the prediction if the different training data was used. In simple words, variance tells that how much a random variable is different from its expected value.
- Ideally, a model should not vary too much from one training dataset to another, which means the algorithm should be good in understanding the hidden mapping between inputs and output variables. Variance errors are either of low variance or high variance.
- **Low variance** means there is a small variation in the prediction of the target function with changes in the training data set

  High variance shows a large variation in the prediction of the target function with changes in the training dataset.

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



Bias Variance Trade-Off

There are four possible combinations of bias and variances

1.Low-Bias, Low-Variance:

The combination of low bias and low variance shows an ideal machine learning model. However, it is rarely possible practically.

2.Low-Bias, High-Variance: With low bias and high variance, model predictions are inconsistent and accurate on average. This case occurs when the model learns with a large number of parameters and hence leads to an **overfitting**

3.High-Bias, Low-Variance: With High bias and low variance, predictions are consistent but inaccurate on average. This case occurs when a model does not learn well with the training dataset or uses few numbers of the parameter. **It leads to underfitting problems in the model.**

4.High-Bias,High-Variance:

With high bias and high variance, predictions are inconsistent and also inaccurate on average.

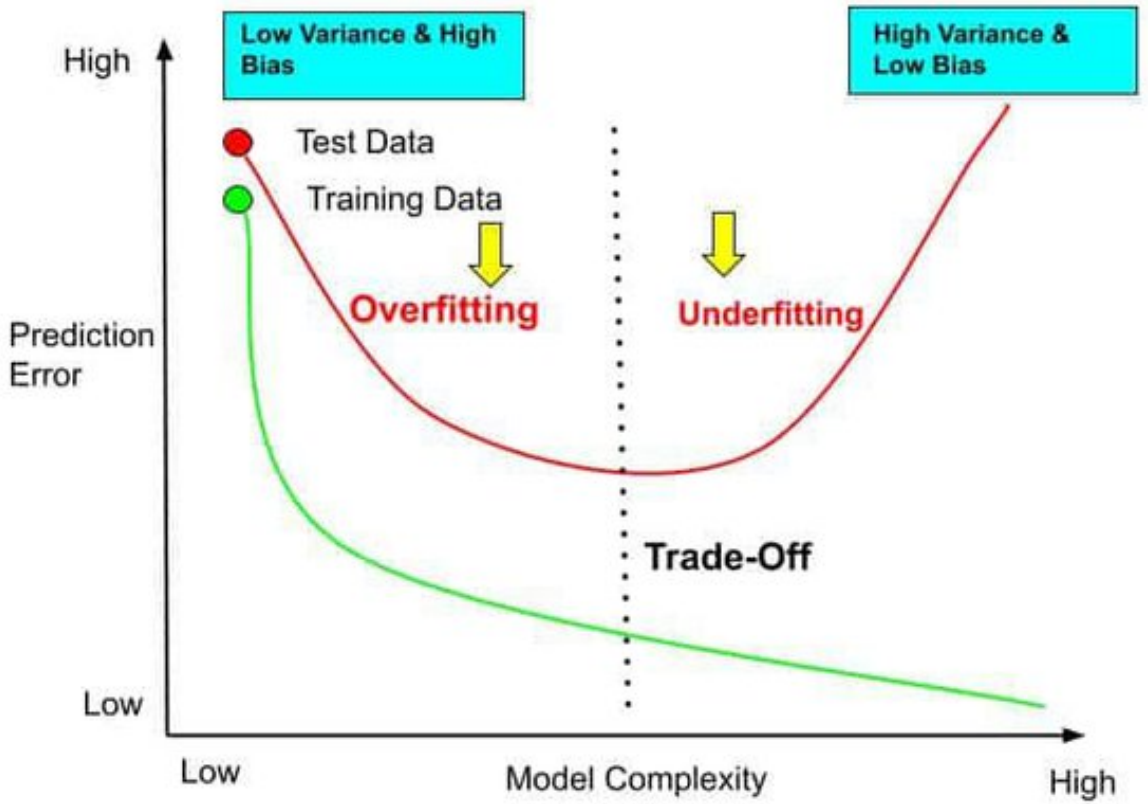
Bias-Variance Trade-Off

While building the machine learning model, it is really important to take care of bias and variance in order to avoid overfitting and underfitting in the model.

If the model is very simple with fewer parameters, it may have low variance and high bias. Whereas, if the model has a large number of parameters, it will have high variance and low bias.

So, it is required to make a balance between bias and variance errors, and this balance between the bias error and variance error is known as **the Bias-Variance trade-off.**

Bias Variance Trade-Off




Thank you!