Assignment 11

1. What are the different validation techniques in Machine Learning?

Ans:

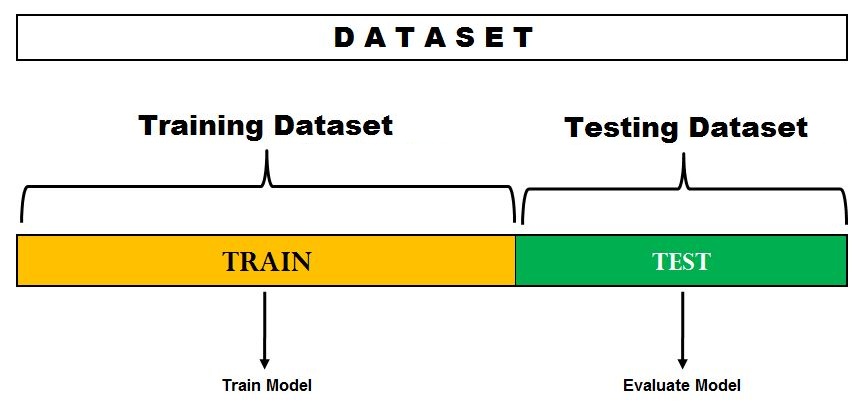
Validation : In machine learning, model validation is referred to as **the process where a trained model is evaluated with a testing data set**. The testing data set is a separate portion of the same data set from which the training set is derived.

**Different Validation Techniques :**

## 

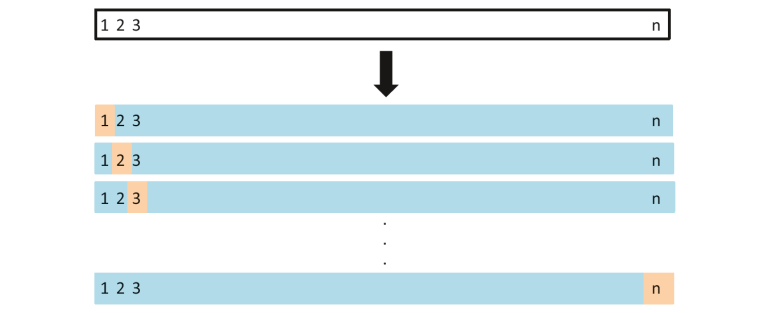
## 1. Hold Out method

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. This can lead to extremely high variance and every time, the split changes, the accuracy will also change.

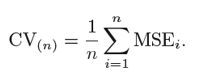


## 2. Leave One Out Cross-Validation

In this method, we divide the data into train and test sets – but with a twist. Instead of dividing the data into 2 subsets, we select a single observation as test data, and everything else is labeled as training data and the model is trained.



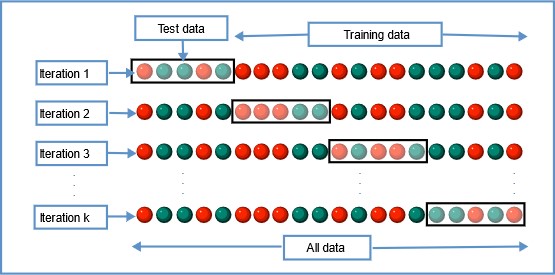
Now the 2nd observation is selected as test data and the model is trained on the remaining data. This process continues ‘n’ times and the average of all these iterations is calculated and estimated as the test set error.



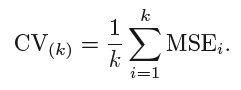
## 3. K-Fold Cross-Validation

In this re-sampling 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.

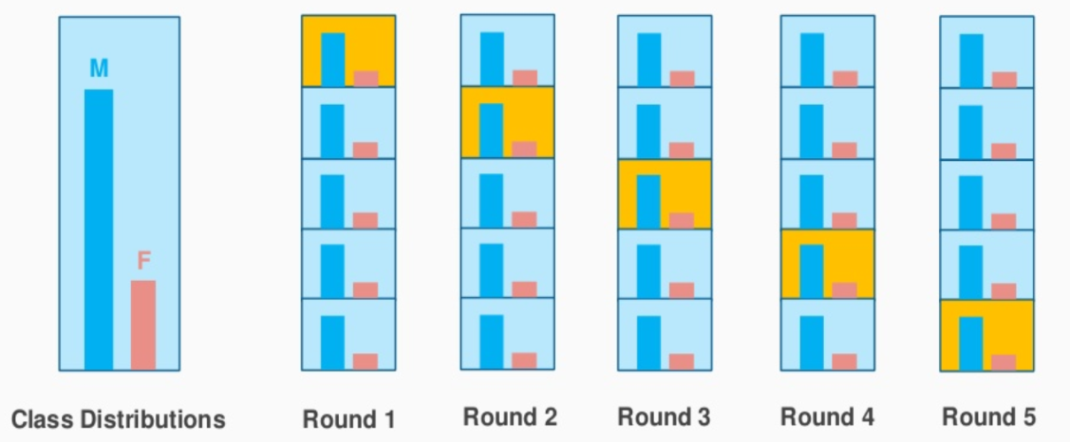


The mean of errors from all the iterations is calculated as the CV test error estimate.



## 4. Stratified K-Fold Cross-Validation

This is a slight variation from K-Fold Cross Validation, which uses **‘stratified sampling’** instead of ‘random sampling.’



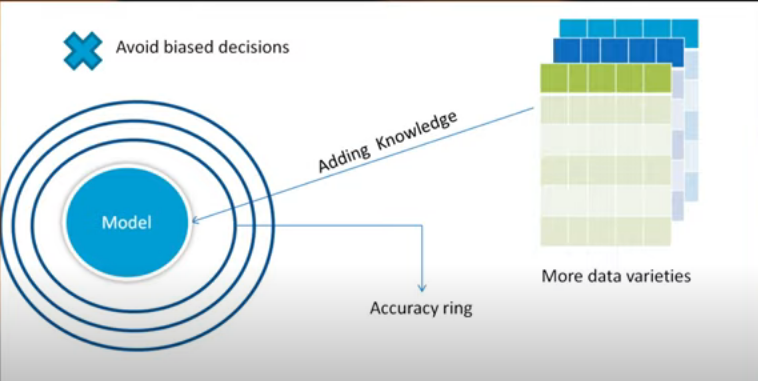
1. What are the different ways of improving the accuracy of a Machine Learning Model?

# Ans:

# Method 1: Add more data samples

Data tells a story only if you have enough of it. Every data sample provides some input and perspective to your data's overall story is trying to tell you. Perhaps the easiest and most straightforward way to improve your model's performance and increase its accuracy is to add more data samples to the training data.

Doing so will add more details to your data and finetune your model resulting in a more accurate performance. Rember after all, the more information you give your model, the more it will learn and the more cases it will be able to identify correctly.



**Method 2 : Feature Selection :**

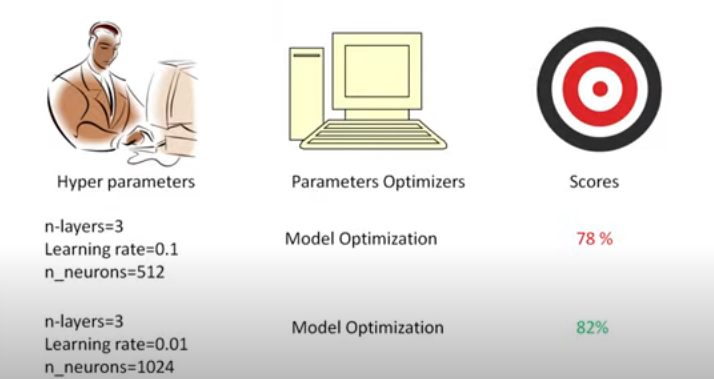
Feature selection is very important part in machine learning. In this part you have to select the features that will create impact on your accuracy of model . Adding more data is good but in the data , you have to select those features which will give less variance in the predictive model. If your variance estimation is high then you have to drop some features from the training part. By reducing the non-impacted features in your data matrix it will help you to reduce the variance and it is most advisable step to drop some unwanted features from your dataset. And one more thing , if you select the more relevant features and fit them into the model you can avoid over fitting results also . So in this step I will give some two important methods to follow while selecting the features.



1. Feature Importance : Some algorithms like random forest or xgboost allow you to determine which features were the most important in predicting the target variable’s value. By quickly creating one of this models and conducting the feature importance you will get understanding of which variables are useful than others.
2. Dimentionality Reduction : One of the most common dimentionality reduction technique is the Principle Component Analysis(PCA). It takes larger no of features and uses linear algebra techniques to reduce them into a fewer features.

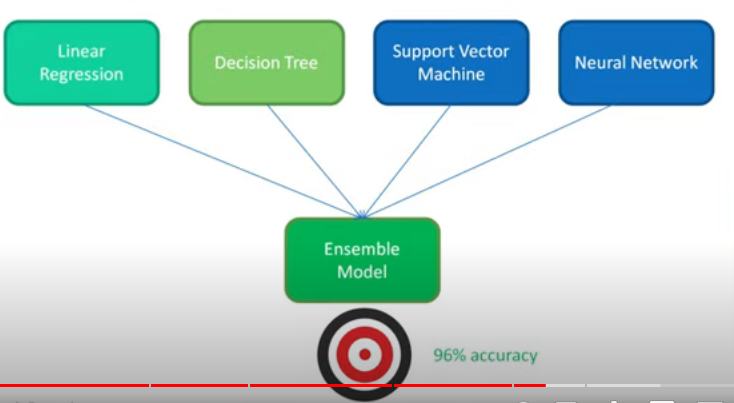
Method 3 : Hyperparameter Tunning:

Training a Machine learning problem is skill that will come if you practice more in problem solving and some times this above methods are not enough to improve the accuracy of your model. So in the field of machine learning , there are lot of techniques to increase the accuracy of the ML model . You can’t able to stick with particular technique to improve the accuracy of ML model . One of the best way to increase the accuracy of the model is by implementing the hyper parameter tuning to your ML model so it is like trial and error method. You have to change the model parameters until you reach some good results and accuracy. So this is strategy that is followed in a hyperparameter tuning. Hyperparametric tuning is more important in unsupervised learning . So take an example , when you are training k-mean algorithms and you don’t know how many clusters to add . In this case you have to change the cluster values until you get some good accuracy.



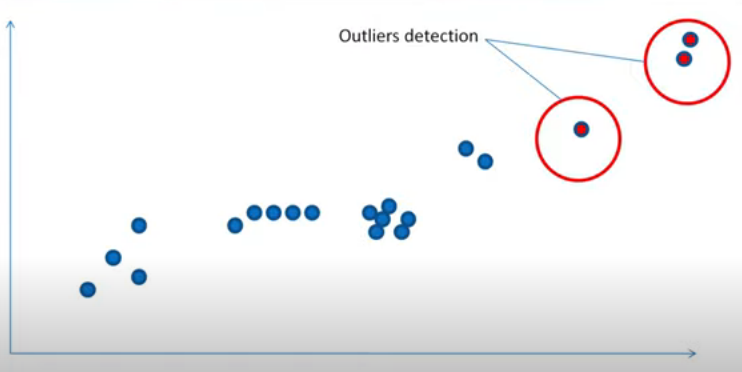
Mothod 4. Ensemble Methods:

This is one of the most common method used by every data scientist. Ensemble methods are nothing but combing the multiple train models and product the good accuracy at the end. So this the process of ensemble methods. Ensemble methods is winning strategy of every data scientist and ML engineers. This can be achieved through two common methods . The first one is bagging and 2nd one is boosting.



Method 5 : Outlier Detection and Anomalies :

Outliers and Anomalies are same so , the outlier term explains that every data point which present in dataset has some relations between each other . If some datapoints are unlike to each other that points are called outliers. So outliers will reduce the accuracy of the model that you trained. So if you find any outlier you have to remove first in your dataset. The process of removing outlier is called outlier detection or anomaly detection. So outlier dataset is very harmful to ML model means that data leads to data corruption and it affects the accuracy also. There are some methods and algorithms are there for identifying the outliers in the dataset. That are the standard deviation and inter quartile and range method .This two alogorithms are very useful to remove outliers from the dataset.



1. **What is stratiﬁed cross-validation and when should we use it?**

Ans:

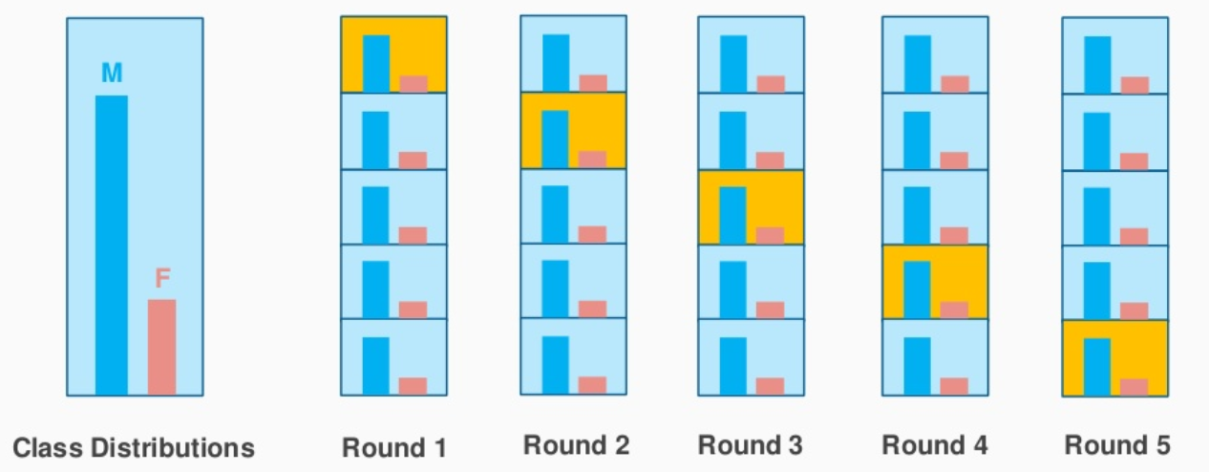
What is stratified sampling?

Before diving deep into stratified cross-validation, it is important to know about stratified sampling. Stratified sampling is a sampling technique where the samples are selected in the same proportion (by dividing the population into groups called ‘strata’ based on a characteristic) as they appear in the population. For example, if the population of interest has 30% male and 70% female subjects, then we divide the population into two (‘male’ and ‘female’) groups and choose 30% of the sample from the ‘male’ group and ‘70%’ of the sample from the ‘female’ group.

* **How is stratified sampling related to cross-validation?**

Implementing the concept of stratified sampling in cross-validation ensures the training and test sets have the same proportion of the feature of interest as in the original dataset. Doing this with the target variable ensures that the cross-validation result is a close approximation of generalization error

Stratified Cross Validation — **When we split our data into folds, we want to make sure that each fold is a good representative of the whole data**. The most basic example is that we want the same proportion of different classes in each fold.



**When should we use it ? :**

it is common, **in the case of class imbalances** in particular, to use stratified 10-fold cross-validation, which ensures that the proportion of positive to negative examples found in the original distribution is respected in all the folds.

1. **What is the difference between the validation set and the hold out test set?**

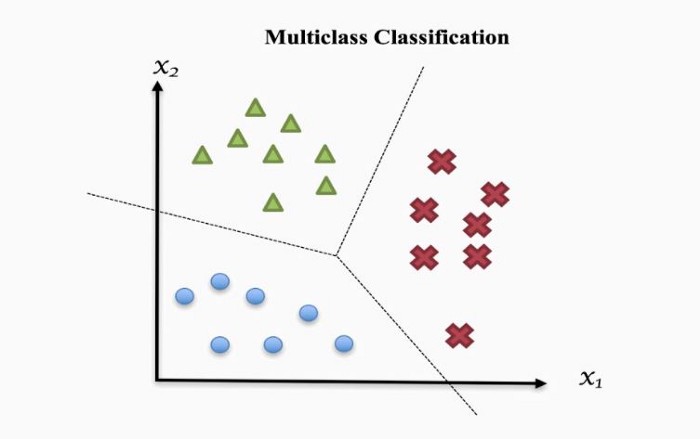
**Definitions of Train, Validation, and Test Datasets**

* + **Training Dataset**: The sample of data used to fit the model.
  + **Validation Dataset**: The sample of data used to provide an unbiased evaluation of a model fit on the training dataset while tuning model hyperparameters. The evaluation becomes more biased as skill on the validation dataset is incorporated into the model configuration.
  + **Test Dataset**: The sample of data used to provide an unbiased evaluation of a final model fit on the training dataset.

1. **How to solve a multi-class classiﬁcation problem vs multi-label classiﬁcation problem?**

Ans:

**1. What is Multi-class Classification?**

****

When we solve a classification problem having only two class labels, then it becomes easy for us to filter the data, apply any classification algorithm, train the model with filtered data, and predict the outcomes. But when we have more than two class instances in input train data, then it might get complex to analyze the data, train the model, and predict relatively accurate results. To handle these multiple class instances, we use multi-class classification.

Multi-class classification is the classification technique that allows us to categorize the test data into multiple class labels present in trained data as a model prediction.

There are mainly two types of multi-class classification techniques:-

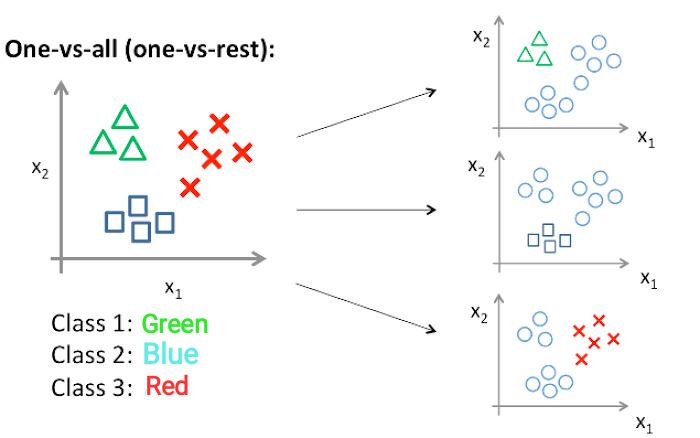
* **One vs. All (one-vs-rest)**
* **One vs. One**

**Multi-class Classification**

* Multiple class labels are present in the dataset.
* The number of classifier models depends on the classification technique we are applying to.
* One vs. All:- **N-class instances**then **N binary classifier models**
* One vs. One:- **N-class instances**then**N\* (N-1)/2 binary classifier models**
* The Confusion matrix is easy to derive but complex to understand.
* Example:- Check whether the fruit is apple, banana, or orange.

**One vs. All (One-vs-Rest)**

In one-vs-All classification, for the N-class instances dataset, we have to generate the N-binary classifier models. The number of class labels present in the dataset and the number of generated binary classifiers must be the same.

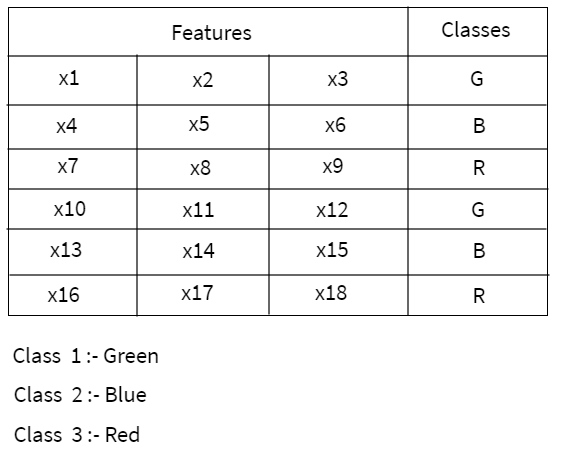


As shown in the above image, consider we have three classes, for example, type 1 for Green, type 2 for Blue, and type 3 for Red.

Now, as I told you earlier that we have to generate the same number of classifiers as the class labels are present in the dataset, So we have to create three classifiers here for three respective classes.

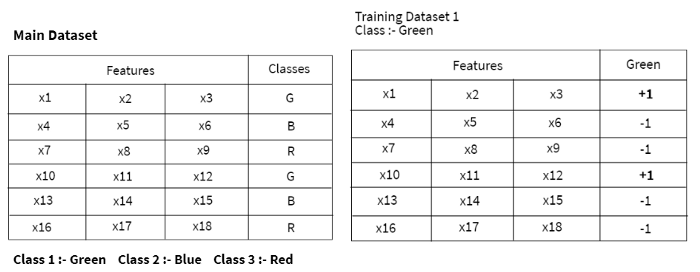
* **Classifier 1:- [Green] vs [Red, Blue]**
* **Classifier 2:- [Blue] vs [Green, Red]**
* **Classifier 3:- [Red] vs [Blue, Green]**

Now to train these three classifiers, we need to create three training datasets. So let’s consider our primary dataset is as follows,



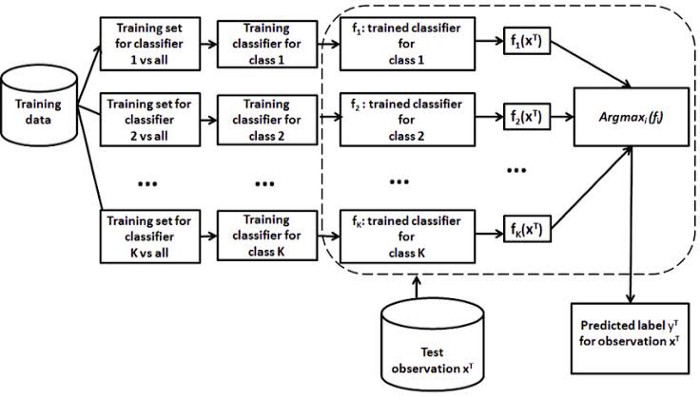
You can see that there are three class labels **Green**, **Blue,** and **Red** present in the dataset. Now we have to create a training dataset for each class.

Here, we created the training datasets by putting +1 in the class column for that feature value, which is aligned to that particular class only. For the costs of the remaining features, we put -1 in the class column.

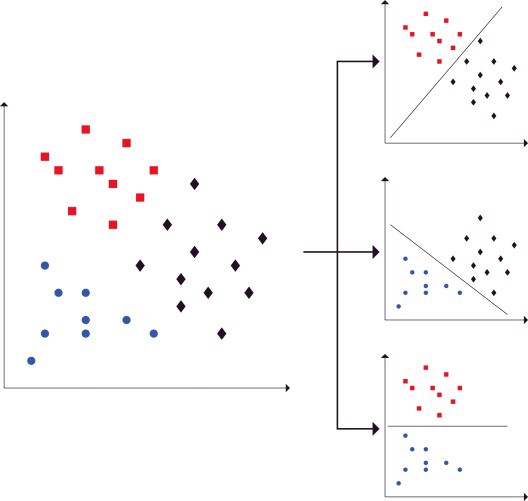


Traing dataset of green class

# blue and red class training dataset.png Traing dataset of blue and red class



# One vs. One (OvO)



In One-vs-One classification, for the **N-class** instances dataset, we have to generate the **N\* (N-1)/2**binary classifier models. Using this classification approach, we split the primary dataset into one dataset for each class opposite to every other class.

Taking the above example, we have a classification problem having three types: **Green**, **Blue**, and **Red (N=3).**

We divide this problem into **N\* (N-1)/2 = 3**binary classifier problems:

* Classifier 1: Green vs. Blue
* Classifier 2: Green vs. Red
* Classifier 3: Blue vs. Red

Each binary classifier predicts one class label. When we input the test data to the classifier, then the model with the majority counts is concluded as a result.

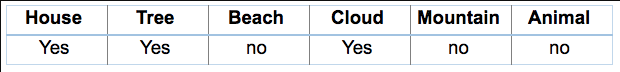
What is Multi-Label Classification?

Let us take a look at the image below.

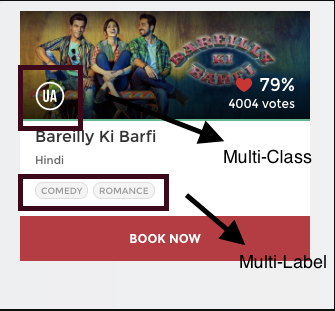


What if I ask you that does this image contains a house? The option will be **YES** or **NO**.

Consider another case, like what all things (or labels) are relevant to this picture?



These types of problems, where we have a set of target variables, are known as **multi-label classification** problems. So, is there any difference between these two cases? Clearly, yes because in the second case any image may contain a different set of these multiple labels for different images.



**Basically, there are three methods to solve a multi-label classification problem, namely:**

1. Problem Transformation
2. Adapted Algorithm
3. Ensemble approaches

### 1) Problem Transformation

In this method, we will try to transform our multi-label problem into single-label problem(s).

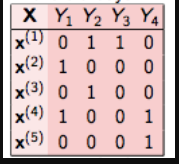
This method can be carried out in three different ways as:

1. Binary Relevance
2. Classifier Chains
3. Label Powerset

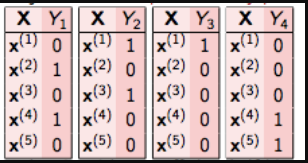
#### .1 Binary Relevance

This is the simplest technique, which basically treats each label as a separate single class classification problem.

For example, let us consider a case as shown below. We have the data set like this, where X is the independent feature and Y’s are the target variable.



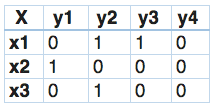
In binary relevance, this problem is broken into 4 different single class classification problems as shown in the figure below.



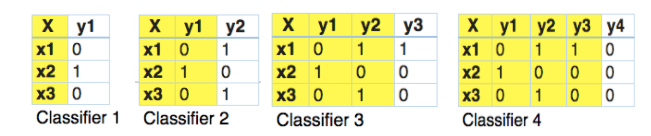
#### 2 Classifier Chains

In this, the first classifier is trained just on the input data and then each next classifier is trained on the input space and all the previous classifiers in the chain.

Let’s try to this understand this by an example. In the dataset given below, we have X as the input space and Y’s as the labels



In classifier chains, this problem would be transformed into 4 different single label problems, just like shown below. Here yellow colored is the input space and the white part represent the target variable.



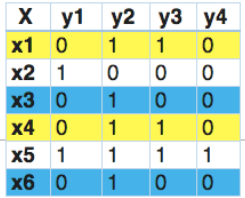
This is quite similar to binary relevance, the only difference being it forms chains in order to preserve label correlation.

We can see that using this we obtained an accuracy of about **21%**, which is very less than binary relevance. This is maybe due to the absence of label correlation since we have randomly generated the data.

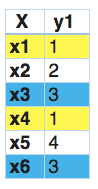
#### 3 Label Powerset

In this, we transform the problem into a multi-class problem with one multi-class classifier is trained on all unique label combinations found in the training data.

Let’s understand it by an example.



In this, we find that x1 and x4 have the same labels, similarly, x3 and x6 have the same set of labels. So, label powerset transforms this problem into a single multi-class problem as shown below.



So, label powerset has given a unique class to every possible label combination that is present in the training set.

### 2)Adapted Algorithm

Adapted algorithm, as the name suggests, adapting the algorithm to directly perform multi-label classification, rather than transforming the problem into different subsets of problems.

For example, multi-label version of kNN is represented by MLkNN. So, let us quickly implement this on our randomly generated data set.

### 3) Ensemble Approaches

Ensemble always produces better results. Scikit-Multilearn library provides different ensembling classification functions, which you can use for obtaining better results.