**1. What is the definition of a target function? In the sense of a real-life example, express the target function. How is a target function's fitness assessed?**

Ans: A target function, in machine learning, is a method for solving a problem that an AI algorithm parses its training data to find. Once an algorithm finds its target function, that function can be used to predict results (predictive analysis)

In **machine learning**, the *mapping function* takes input data and gives the corresponding output data. This gives the *target function*— the relationship between inputs and outputs:

When designing a Learning system in ML. Steps taken to arrive at best target function which is assessed by metrics.

Example: DeepBlue is an intelligent computer which is ML-based won chess game against the chess expert Garry Kasparov, and it became the first computer which had beaten a human chess expert.

**2. What are predictive models, and how do they work? What are descriptive types, and how do you use them? Examples of both types of models should be provided. Distinguish between these two forms of models.**

Ans: Descriptive Analytics, which use data aggregation and data mining to provide insight into the past and answer: “What has happened?”

Predictive Analytics, which use statistical models and forecasting techniques to understand the future and answer: “What could happen?”

A descriptive model will exploit the past data that are stored in databases and provide you with the accurate report. In a Predictive model, it identifies patterns found in past and transactional data to find risks and future outcomes.

Predictive models: They work by consuming the data. The experience gained from this data is used to predict outcomes from new unseen data.The main goal of predictive models is to predict future results rather than current behaviour. Data modelling is the fundamental of predictive analysis, which works on some variables to anticipate the unknown future data values for other variables.

Descriptive models used to provide correlation, cross tabulation and frequency etc. used to decide the data regularities and to reveal patterns. It focuses on the summarization and conversion of records into significant data for reporting and monitoring. It describes the data. it can modify data into human readable form. An association technique that uses Apriori algorithms to characterize, to find correlation between set of data.

| **Descriptive Model** | **Predictive model** |
| --- | --- |
| Descriptive Model generally used to support correlation, cross-tabulation, frequency, etc. | The term 'Predictive' is defined to predict something, so predictive data mining is the analysis done to predict the future event or multiple data or trends. |
| It defines the features of the data in a target data set. | It executes the induction over the current and past records so that predictions can appear. |
| It requires data aggregation and data mining. | It requires statistics and data forecasting procedures. |
| The descriptive analysis only responds to the situation. | The predictive analysis includes control over the situation along with responding to it. |
| It can support accurate records. | It makes results |

**3. Describe the method of assessing a classification model's efficiency in detail. Describe the various measurement parameters.**

Ans: Methods of assessing a classification model’s :

* Confusion Matrix
* Precision
* Recall/ Sensitivity
* Specificity
* F1-Score
* AUC & ROC CurveConfusion Matrix:

Confusion Matrix usually causes a lot of confusion even in those who are using them regularly. Terms used in defining a confusion matrix are TP, TN, FP, and FN.

Use case: Let’s take an example of a patient who has gone to a doctor with certain symptoms. Since it’s the season of Covid, let’s assume that he went with fever, cough, throat ache, and cold. These are symptoms that can occur during any seasonal changes too. Hence, it is tricky for the doctor to do the right diagnosis.

True Positive (TP):

Let’s say the patient was actually suffering from Covid and on doing the required assessment, the doctor classified him as a Covid patient. This is called TP or True Positive. This is because the case is positive in real and at the same time the case was classified correctly. Now, the patient can be given appropriate treatment which means, the decision made by the doctor will have a positive effect on the patient and society.

False Positive (FP):

Let’s say the patient was not suffering from Covid and he was only showing symptoms of seasonal flu but the doctor diagnosed him with Covid. This is called FP or False Positive. This is because the case was actually negative but was falsely classified as positive. Now, the patient will end up getting admitted to the hospital or home and will be given treatment for Covid. This is an unnecessary inconvenience for him and others as he will get unwanted treatment and quarantine. This is called Type I Error.

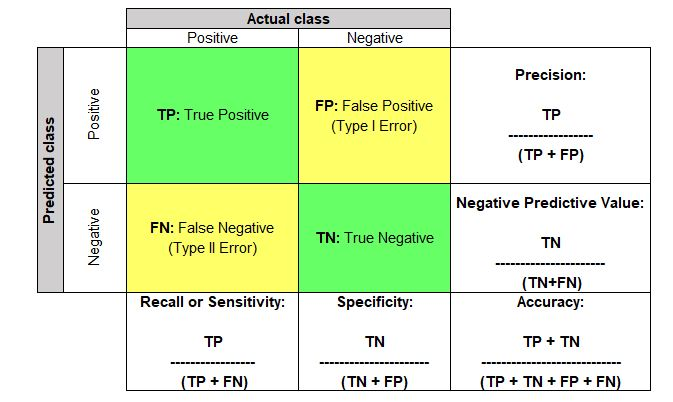
True Negative (TN):

Let’s say the patient was not suffering from Covid and the doctor also gave him a clean chit. This is called TN or True Negative. This is because the case was actually negative and was also classified as negative which is the right thing to do. Now the patient will get treatment for his actual illness instead of taking Covid treatment.

False Negative (FN):

Let’s say the patient was suffering from Covid and the doctor did not diagnose him with Covid. This is called FN or False Negative as the case was actually positive but was falsely classified as negative. Now the patient will not get the right treatment and also he will spread the disease to others. This is a highly dangerous situation in this example. This is also called Type II Error.

Confusion Matrix:



Accuracy:

Accuracy = (TP + TN) / (TP + FP +TN + FN)

This term tells us how many right classifications were made out of all the classifications. In other words, how many TPs and TNs were done out of TP + TN + FP + FNs. It tells the ratio of “True”s to the sum of “True”s and “False”s.

Use case: Out of all the patients who visited the doctor, how many were correctly diagnosed as Covid positive and Covid negative.

Precision:

Precision = TP / (TP + FP)

Out of all that were marked as positive, how many are actually truly positive.

Use case: Let’s take another example of a classification algorithm that marks emails as spam or not. Here, if emails that are of importance get marked as positive, then useful emails will end up going to the “Spam” folder, which is dangerous. Hence, the classification model which has the least FP value needs to be selected. In other words, a model that has the highest precision needs to be selected among all the models.

Recall or Sensitivity:

Recall = TP/ (TN + FN)

Out of all the actual real positive cases, how many were identified as positive.

Use case: Out of all the actual Covid patients who visited the doctor, how many were actually diagnosed as Covid positive. Hence, the classification model which has the least FN value needs to be selected. In other words, a model that has the highest recall value needs to be selected among all the models.

Specificity:

Specificity = TN/ (TN + FP)

Out of all the real negative cases, how many were identified as negative.

Use case: Out of all the non-Covid patients who visited the doctor, how many were diagnosed as non-Covid.

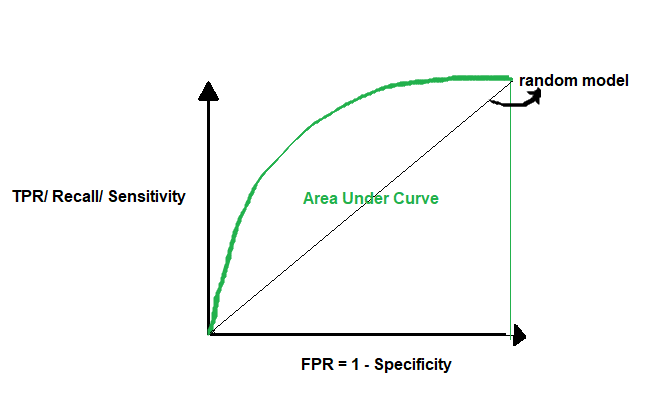
F1-Score:

F1 score = 2\* (Precision \* Recall) / (Precision + Recall)

As we saw above, sometimes we need to give weightage to FP and sometimes to FN. F1 score is a weighted average of Precision and Recall, which means there is equal importance given to FP and FN. This is a very useful metric compared to “Accuracy”. The problem with using accuracy is that if we have a highly imbalanced dataset for training (for example, a training dataset with 95% positive class and 5% negative class), the model will end up learning how to predict the positive class properly and will not learn how to identify the negative class. But the model will still have very high accuracy in the test dataset too as it will know how to identify the positives really well.

Use case: Let’s take an example where we must give equal importance to both the classes – classify an email as Spam and non-Spam. Let’s assume that the model was trained only a highly imbalanced training dataset. Here, Spam is “positive” and non-Spam is “negative” and the training dataset was 90% spam emails and 10% non-spam emails. A model with high accuracy will know to correctly identify all the spam emails but will have trouble identifying non-spam emails. Hence, a lot of important emails will end up going to the spam folder. But if we select a model that has a high F1 score, it would perform better in classifying non-spam from spam.

Area Under Curve (AUC) and ROC Curve:



AUC or Area Under Curve is used in conjecture with ROC Curve which is Receiver Operating Characteristics Curve. AUC is the area under the ROC Curve. So let’s first understand the ROC Curve.

A ROC Curve is drawn by plotting TPR or True Positive Rate or Recall or Sensitivity (which we saw above) in the y-axis against FPR or False Positive Rate in the x-axis. FPR = 1- Specificity (which we saw above).

TPR = TP/ (TP + FN)

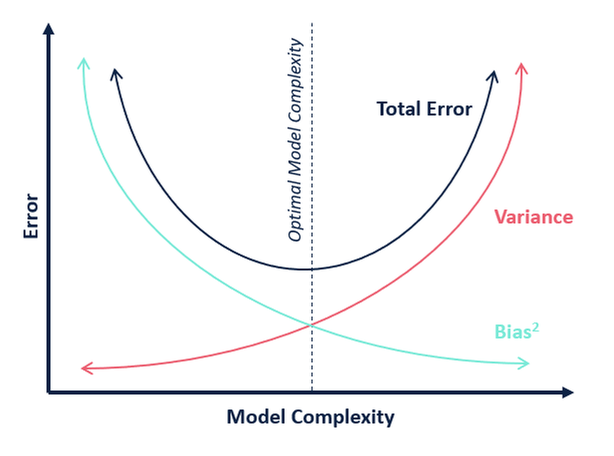
FPR = 1 – TN/ (TN+FP) = FP/ (TN + FP)

If we use a random model to classify, it has a 50% probability of classifying the positive and negative classes correctly. Here, the AUC = 0.5. A perfect model has a 100% probability of classifying the positive and negative classes correctly. Here, the AUC = 1. So when we want to select the best model, we want a model that is closest to the perfect model. In other words, a model with AUC close to 1. When we say a model has a high AUC score, it means the model’s ability to separate the classes is very high (high separability). This is a very important metric that should be checked while selecting a classification model.

**4.**

**i. In the sense of machine learning models, what is underfitting? What is the most common reason for underfitting?**

**Ans:** where the model cannot create a mapping between the input and the target variable. Under-observing the features leads to a higher error in the training and unseen data samples.It is detected when the training error is very high and the model is unable to learn from the training data. High bias and low variance are the most common indicators of underfitting.

****

Underfitting happens when:

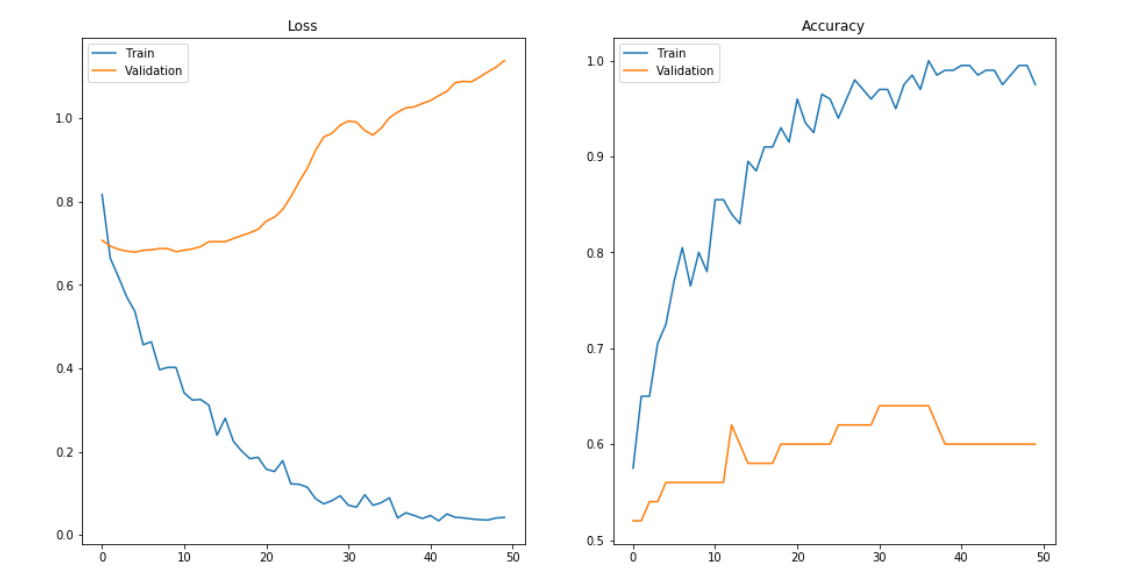
Unclean training data containing noise or outliers can be a reason for the model not being able to derive patterns from the dataset.

The model has a high bias due to the inability to capture the relationship between the input examples and the target values. This usually happens in the case of varied datasets.

The model is assumed to be too simple—for example, we train a linear model in complex scenarios.

Incorrect hyperparameters tuning often leads to underfitting due to under-observing of the features.

**ii. What does it mean to overfit? When is it going to happen?**

**Ans**: Overfitting, in which a model tries to fit the training data entirely and ends up memorizing the data patterns and the noise/random fluctuations. These models fail to generalize and do not perform well in the case of unseen data scenarios, defeating the model's purpose.

Overfitting happens when:

The training data is not cleaned and contains some “garbage” values. The model captures the noise in the training data and fails to generalize the model's learning.

The model has a high variance.

The training data size is insufficient, and the model trains on the limited training data for several epochs.

The architecture of the model has several neural layers bundled together. Deep neural networks are complex and require a significant amount of time to train, and often lead to overfitting the training set.

Incorrect tuning of hyperparameters in the training phase leads to over-observing the training set, resulting in memorizing features.

**iii. In the sense of model fitting, explain the bias-variance trade-off.**

**Ans:** In statistics and machine learning, the bias–variance tradeoff is the property of a model that the variance of the parameter estimated across samples can be reduced by increasing the bias in the estimated parameters.

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 a 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 overfitting and underfitting the data.

This tradeoff in complexity is why there is a tradeoff 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.



An optimal balance of bias and variance would never overfit or underfit the model.

**5. Is it possible to boost the efficiency of a learning model? If so, please clarify how.**

**Ans:**  Boosting efficiency of learning model:

1. Choose Robust Algorithm

Think of machine learning algorithms as the engines of machine learning. Algorithms are responsible for transforming data sets into accurate predictions. Selecting the right algorithm for your model is essential to its performance. So, how do you know which one works best?

There’s no straightforward answer. Some problems call for a highly specific approach, while others are open for interpretation. Here are a few questions to keep in mind when narrowing down your list:

* What kind of problem are you solving?
* What computing resources are available?
* What type of data are you processing? How much data are you processing?
* How scalable do you want your model to be?
* Which business goals would you like your model to fulfill?

1. Use Large, High-Quality Training Datasets:

“garbage in, garbage out”? When it comes to the quality of your training data, no saying could hold more truth. If your objective is to improve the performance of ML models, it’s best to start with the information they’re fed.

For example, let’s say you’re training an algorithm to flag suspicious credit card charges. For your model to accurately predict whether a charge is likely to be fraudulent, you need to train it with accurately labeled credit card transaction data.

When it comes to training data for machine learning algorithms, quantity matters as much as quality. An algorithm for detecting credit card fraud that was trained using 10,000 transactions will perform better than one that was trained using 100.

A machine learning model that learns using low-quality or insufficient training data will perform poorly — regardless of whatever algorithm powers it.

1. Validate and Test Your Machine Learning Model

There’s only one way to check whether your model is accurate: test it. That’s where validation data comes in.

While training data is used to teach your algorithm to identify patterns, testing data evaluate its accuracy. Why test your machine learning model? Testing tells you whether you chose the right algorithm and defined the right parameters by measuring your model’s accuracy, precision, and efficiency.

How do you validate your machine learning model? Let’s walk through an example. Say you’re training your model to detect whether or not a vehicle is present in an image. Training data would include labeled images of vehicles. Testing data, on the other hand, would consist of unlabelled images.

**6. How would you rate an unsupervised learning model's success? What are the most common success indicators for an unsupervised learning model?**

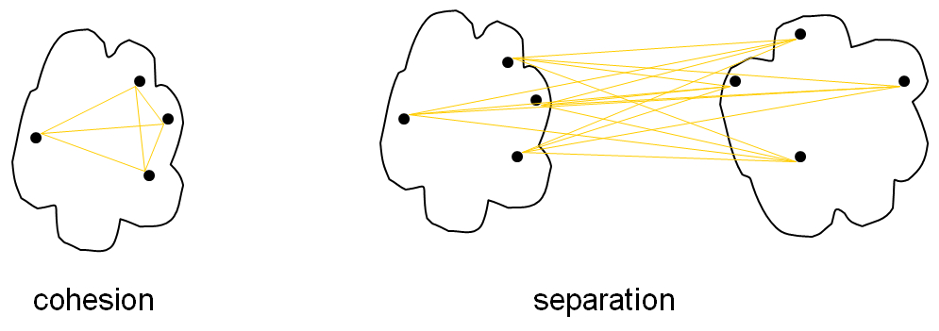
**Ans:** There are two classes of statistical techniques to validate results for cluster learning. These are:

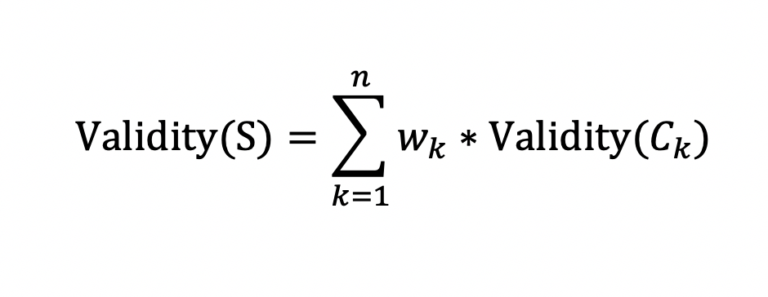
1. Internal validation
   1. Cohesion within each cluster
   2. Separation between different clusters
2. External validation

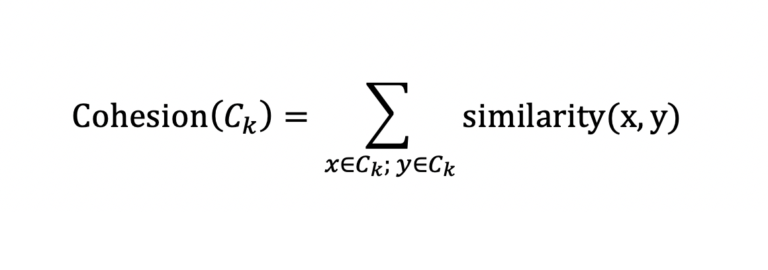
In this article, we propose the twin-sample validation as a methodology to validate results of unsupervised learning in addition to internal validation, which is very similar to external validation, but without the need for human inputs.

Internal Validation

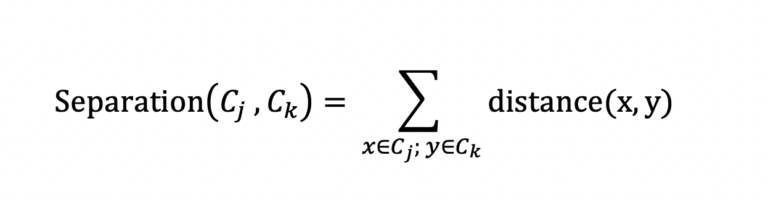
Most of the methods of internal validation combine cohesion and separation to estimate the validation score.





The approach is to compute validation score of each cluster and then combine them in a weighted manner to arrive at the final score for the set of clusters. Let S be a set of clusters {C1 , C2 , C3 ,…………, Cn }, then validity of S will be computed as follows:

Cohesion for a cluster can be computed by summating the similarity between each pair of records contained in that cluster.

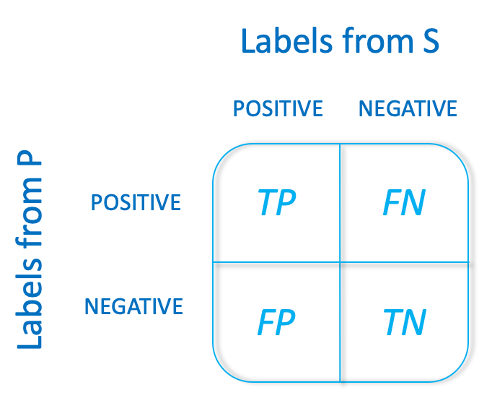


Separation between two clusters can be computed by summating the distance between each pair of records falling within the two clusters and both the records are from different clusters.

* A set of clusters having high cohesion within the clusters and high separation between the clusters is considered to be good.
* In practice, instead of dealing with two metrics, several measures are available which combine both of the above into a single measure. Few examples of such measures are:
* Silhouette coefficient
* Calisnki-Harabasz coefficient
* Dunn index
* Xie-Beni score
* Hartigan index

External Validation

This type of result validation can be carried out if true cluster labels are available. Labels generated by SMEs can also be used to proxy true labels. In this approach we will have a set of clusters S= {C1, C2, C3,…………, Cn } which have been generated as a result of some clustering algorithm. We will have another set of clusters P = {D1, D2, D3, …………, Dm} which represent the true cluster labels on the same data. The idea is to measure the statistical similarity between the two sets. A cluster set is considered as good if it is highly similar to the true cluster set.

In order to measure the similarity between S and P, we label each pair of records from data as Positive if the pairs belong to the same cluster in P else Negative. Similar exercise is carried out for S as well. We then compute a confusion matrix between pair labels of S and P which can be used measure the similarity.

TP: Number of pairs of records which are in the same cluster, for both S and P

FP: Number of pairs of records which are in the same cluster in S but not in P

FN: Number of pairs of records which are in the same cluster in P but not in S

TN: Number of pairs of records which are not in the same cluster S as well as P

On the above 4 indicators, we can calculate different metrics to get an estimate for the similarity between S (cluster labels generated by unsupervised method) and P (true cluster labels). Some example metrics which could be used are as follows:

Precision measures the ratio of true positives to total positives predicted.

Recall measures the ratio of positives captured out of the total true positives.

F1-measure combines precision and recall into a single metric.

Jaccard Similarity

Mutual Information

Fowlkes-Mallows Index

Twin-Sample Validation

In this section, we explain how we can further validate the results of our unsupervised learning model in the absence of true cluster labels. This step takes it as a given that we have already performed clustering on our training data and now want to validate the results. The approach consists of following four steps:

* Creating a twin-sample of training data
* Performing unsupervised learning on twin-sample
* Importing results for twin-sample from training set
* Calculating similarity between two sets of results

1. Creating a twin-sample

This is the most important step in the process of performing the twin-sample validation. The key idea is to create a sample of records which is expected to exhibit similar behavior as the training set. This is similar to a validation set for supervised learning, only with additional constraints. The following constraints should be considered while creating a twin-sample:

It should come from the same distribution as the training set.

It should sufficiently cover most of the patterns observed in the training set.

In case of the time series data:

It should come from a different duration (immediately succeeding is a good choice) than the training set.

It should cover at least 1 complete season of the data i.e. if the data has weekly seasonality, twin-sample should cover at least 1 complete week.

Keeping the above constraints in mind, a twin-sample can be formed and used to validate results of the clustering performed on the training set.

2. Performing unsupervised learning on twin-sample

Now that we have our twin-sample, the next step is to perform cluster learning on it. For this, we will use the same parameter that we used on our training set. This includes the number of clusters, distance metric, etc. We will get a set of cluster labels as output of this step. We will denote this output set as S. The idea here is that we should get similar results on our twin-sample set as we got on our training set, given that both these sets contain similar data and we are using the same parameter set. This similarity will be measured in the subsequent steps.

3. Importing results for twin-sample from training set

In this step, we will compute another set of cluster labels on the twin-sample. This time we will use the results of clustering performed on the training set. For each point in twin-sample, we will perform the following two steps:

1. Identify its nearest neighbor in the training set. Please note that the distance metric should be same as the one used in clustering process.
2. Import the cluster label of its nearest neighbor.

Following the above process, we will have a cluster label for each point in the twin sample. Let’s denote this set cluster labels by P.

4. Calculating similarity between two sets results

Now that we have two sets of cluster labels, S and P, for twin-sample, we can compute their similarity by using any measure such as F1-measure, Jaccard Similarity, etc. defined in the External Validation section.

A set of clusters having high similarity with its twin-sample is considered good.

**7. Is it possible to use a classification model for numerical data or a regression model for categorical data with a classification model? Explain your answer.**

**Ans:**

Numerical data Classification:

A quantitative classification allows numerical division of data into classes. here, each class represents a range of numerical values for the phenomenon under consideration. accordingly we frame each class with a lower and higher value and according to the range of data. So its possible to use a “Classification model for numerical data”.

Regression model for categorical data with a classification model:

Logistic regression model is used for binary classification of categorical data.. It is highly recommended to start from this model setting before more sophisticated categorical modeling is carried out. Dependent variable yi can only take two possible outcomes. We assume yi follows a Bernoulli distribution with probability π*i*. The probability of the 'event' response π*i* depends on a set of individual characteristics *xi*, *i* = 1, . . ., *n*, where *n* is the number of observations.

The logistic regression model specifies that:

Pr(y1=1|xi)=πi=11+exp(−x′iβ)=exp(x′iβ)1+exp(x′iβ)

MULTINOMIAL LOGISTIC REGRESSION MODEL

In the multinomial logistic regression model individual characteristics can be different for different choices. This model is also known as the conditional logit model due to the fact that individual characteristics depend on the chosen alternative.

The multinomial logistic regression model specification is,

Pr(y1=j|xij)=exp(x′ijβ)∑Jk=1exp(x′ikβ),j=1,...,J.

There is only one vector of unknown parameters β, but we have J vectors of known characteristics xi1, xi2, ..., xiJ.

**8. Describe the predictive modeling method for numerical values. What distinguishes it from categorical predictive modeling?**

**Ans:** Regression Techniques

Regression algorithms are machine learning techniques for predicting continuous numerical values. They are supervised learning tasks which means they require labelled training examples.

Categorical predictive modeling has its own challenges like:

* A categorical variable has too many levels. This pulls down the performance level of the model. For example, a cat. the variable “zip code” would have numerous levels.
* A categorical variable has levels which rarely occur. Many of these levels have minimal chance of making a real impact on model fit. For example, a variable ‘disease’ might have some levels which would rarely occur.
* There is one level which always occurs i.e. for most of the observations in the data set there is only one level. Variables with such levels fail to make a positive impact on model performance due to very low variation.
* If the categorical variable is masked, it becomes a laborious task to decipher its meaning. Such situations are commonly found in [data science competitions](http://datahack.analyticsvidhya.com/).
* You can’t fit categorical variables into a regression equation in their raw form. They must be treated.
* Most of the algorithms (or ML libraries) produce better result with numerical variables. In python, library “sklearn” requires features in numerical arrays. Look at the below snapshot. I have applied random forest using sklearn library on titanic data set (only two features sex and pclass are taken as independent variables). It has returned an error because the feature “sex” is categorical and has not been converted to numerical form.

**9. The following data were collected when using a classification model to predict the malignancy of a group of patients' tumors:**

**i. Accurate estimates – 15 cancerous, 75 benign**

**ii. Wrong predictions – 3 cancerous, 7 benign**

**Determine the model's error rate, Kappa value, sensitivity, precision, and F-measure.**

**Ans: TP 35, FN 7,**

**FP 3, TN 75**

**p = 38 , N= 82**

1. Sensitivity/Recall/Hit rate/TPR : TP/TP+FN =35/(35+7)
2. Precision/Positive predictive value: TP/TP+FP =35/(35+3)
3. F-measure : 2\*P\*R/P+R = =2TP/(2TP+FP+FN) = 2 \*35/(2\*35+3+7)
4. error rate = FN+FP/tp+tn+fp+fn = 7+3/(35+75+3+7)
5. Kappa value = 2\*(TP\*TN-FN\*FP)/(TP+FP)\*(FP+TN)+(TP+FN)\*(FN+TN)

**10. Make quick notes on:**

**1. The process of holding out**

Holdout Method is the simplest sort of method to evaluate a classifier. In this method, the data set (a collection of data items or examples) is separated into two sets, called the Training set and Test set.

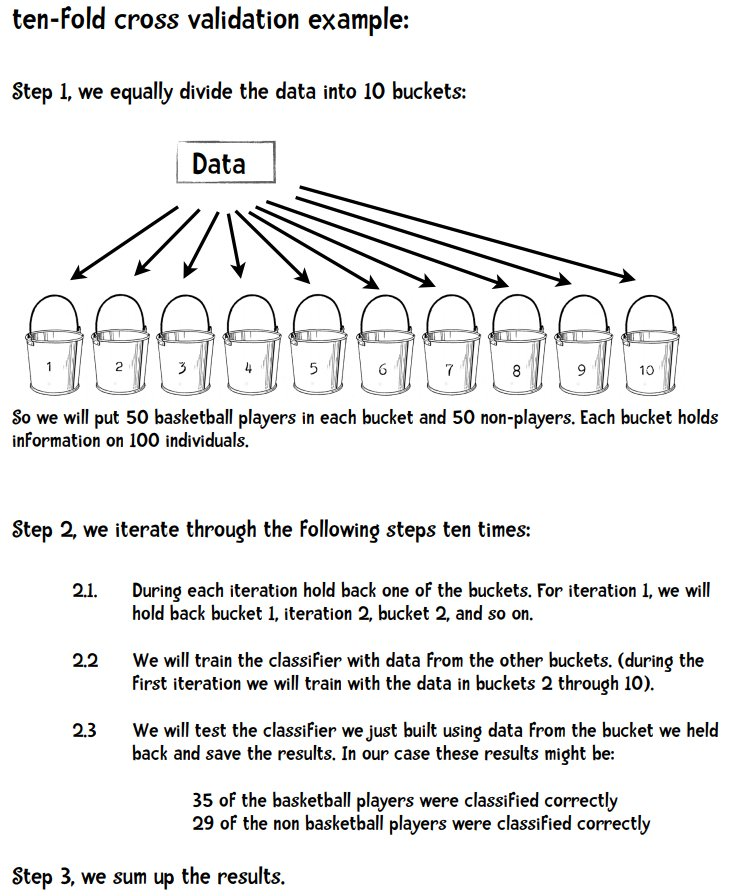
A classifier performs function of assigning data items in a given collection to a target category or class.

Example – E-mails in our inbox being classified into spam and non-spam.

Classifier should be evaluated to find out, it’s accuracy, error rate, and error estimates. It can be done using various methods. One of most primitive methods in evaluation of classifier is ‘Holdout Method’.

**2. Cross-validation by tenfold**

With this method we have one data set which we divide randomly into 10 parts. We use 9 of those parts for training and reserve one tenth for testing. We repeat this procedure 10 times each time reserving a different tenth for testing.



**3. Adjusting the parameters**

In ML/DL, a model is defined or represented by the model parameters. However, the process of training a model involves choosing the optimal hyperparameters that the learning algorithm will use to learn the optimal parameters that correctly map the input features (independent variables) to the labels or targets (dependent variable) such that you achieve some form of intelligence.

Hyperparameters:

Hyperparameters are parameters whose values control the learning process and determine the values of model parameters that a learning algorithm ends up learning. The prefix ‘hyper\_’ suggests that they are ‘top-level’ parameters that control the learning process and the model parameters that result from it.

**11. Define the following terms:**

**1. Purity vs. Silhouette width**

Cluster purity

Another metric to assess cluster separation is the degree to which cells from multiple clusters intermingle in expression space. The “clustering purity” is defined for each cell as the proportion of neighboring cells that are assigned to the same cluster, after some weighting to adjust for differences in the number of cells between clusters. Well-separated clusters should exhibit little intermingling and thus high purity values for all member cells.

Silhouette width

The silhouette width is an established metric for evaluating cluster separation. For each cell, we compute the average distance to all cells in the same cluster. We also compute the average distance to all cells in another cluster, taking the minimum of the averages across all other clusters. The silhouette width for each cell is defined as the difference between these two values divided by their maximum. Cells with large positive silhouette widths are closer to other cells in the same cluster than to cells in different clusters. Thus, clusters with large positive silhouette widths are well-separated from other clusters.

**2. Boosting vs. Bagging**

* Bagging is a method of merging the same type of predictions. Boosting is a method of merging different types of predictions.
* Bagging decreases variance, not bias, and solves over-fitting issues in a model. Boosting decreases bias, not variance.
* In Bagging, each model receives an equal weight. In Boosting, models are weighed based on their performance.
* Models are built independently in Bagging. New models are affected by a previously built model’s performance in Boosting.
* In Bagging, training data subsets are drawn randomly with a replacement for the training dataset. In Boosting, every new subset comprises the elements that were misclassified by previous models.
* Bagging is usually applied where the classifier is unstable and has a high variance. Boosting is usually applied where the classifier is stable and simple and has high bias.

**3. The eager learner vs. the lazy learner**

Eager learning methods construct general, explicit description of the target function based on the provided training examples. Lazy learning methods simply store the data and generalizing beyond these data is postponed until an explicit request is made.

A lazy learner delays abstracting from the data until it is asked to make a prediction while an eager learner abstracts away from the data during training and uses this abstraction to make predictions rather than directly compare queries with instances in the dataset.