MODELLING AND EVALUATION

Introduction

- Machine learning tries to emulate human learning by applying mathematical and statistical formulations.
- Both human and machine learning strives to build formulations or mapping based on a limited number of observations.
- The basic learning process, irrespective of the fact that the learner is a human or a machine, can be divided into three parts:
  1. Data Input
  2. Abstraction
  3. Generalization
- In the learning process, abstraction is a significant step as it represents raw input data in a summarized and structured format, such that a meaningful insight is obtained from the data.
- This structured representation of raw input data to the meaningful pattern is called a model.
- The model might have different forms. It might be a mathematical equation, it might be a graph or tree structure, it might be a computational block, etc.
- The decision regarding which model is to be selected for a specific data set is taken by the learning task, based on the problem to be solved and the type of data.
- For example, when the problem is related to prediction and the target field is numeric and continuous, the regression model is assigned.
- The process of assigning a model, and fitting a specific model to a data set is called model training.
- Once the model is trained, the raw input data is summarized into an abstracted form. However, with abstraction, the learner is able to only summarize the knowledge. This knowledge might be still very broad-based consisting of a huge number of feature-based data and inter-relations.
- To generate actionable insight from such broad-based knowledge is very difficult. This is where generalization comes into
  play.
- Generalization searches through the huge set of abstracted knowledge to come up with a small and manageable set of key findings.
- It is not possible to do an exhaustive search by reviewing each of the abstracted findings one-by-one.
- A heuristic search is employed, an approach which is also used for human learning (often termed as ‘gut-feel’).
- It is quite obvious that the heuristics sometimes result in erroneous result.
- If the outcome is systematically incorrect, the learning is said to have a bias.

SELECTING A MODEL

- For example city Police want to find the pattern of criminal activities in the recent past, i.e. they want to see whether the number of criminal incidents per month has any relation with an average income of the local population, weapon sales, the inflow of immigrants, and other such factors. Therefore, an association between potential causes of disturbance and criminal incidents has to be determined.

- In other words, the goal or target is to develop a model to infer how the criminal incidents change based on the potential influencing factors mentioned above.
- In machine learning paradigm, the potential causes of disturbance, e.g. average income of the local population, weapon sales, the inflow of immigrants, etc. are input variables.
- They are also called predictors, attributes, features, independent variables, or simply variables. The number of criminal incidents is an output variable (also called response or dependent variable).
- Input variables can be denoted by X, while individual input variables are represented as X1 , X2 , X3 , ..., Xn and output variable by symbol Y.
- The relationship between X and Y is represented in the general form: Y = f (X) + e, where 'f ' is the target function and 'e' is a random error term.
- Just like a target function with respect to a machine learning model, some other functions which are frequently tracked are
- **A cost function** (also called error function) helps to measure the extent to which the model is going wrong in estimating the relationship between X and Y. In that sense, cost function can tell how bad the model is performing.
- **Loss function** is almost synonymous to cost function – only difference being loss function is usually a function defined on a data point, while cost function is for the entire training data set.
- Machine learning is an optimization problem. We define a model and tune the parameters to find the most suitable solution to a problem.
- However, we need to have a way to evaluate the quality or optimality of a solution. This is done using objective function. Objective means goal.
- **Objective function** takes in data and model (along with parameters) as input and returns a value. Target is to find values of model parameter to maximize or minimize the return value. When the objective is to minimize the value, it becomes synonymous to cost function.

- Multiple factors play a role in selecting the model for solving a machine learning problem. The most important factors are (i) the kind of problem we want to solve using machine learning and (ii) the nature of the underlying data.
- The problem may be related to the prediction of a class value like whether a tumour is malignant or benign, whether the next day will be snowy or rainy, etc.
- It may be related to prediction – but of some numerical value like what the price of a house should be in the next quarter, what is the expected growth of a certain IT stock in the next 7 days, etc.
- Certain problems are related to grouping of data like finding customer segments that are using a certain product, movie genres which have got more box office success in the last one year, etc.
- So, it is very difficult to give a generic guidance related to which machine learning has to be selected.
- In other words, there is no one model that works best for every machine learning problem. This is what 'No Free Lunch' theorem also states.

- Any learning model tries to simulate some real-world aspect. However, it is simplified to a large extent removing all intricate details.
- These simplifications are based on certain assumptions – which are quite dependent on situations.
- Based on the exact situation, i.e. the problem in hand and the data characteristics, assumptions may or may not hold. So the same model may yield remarkable results in a certain situation while it may completely fail in a different situation.
- we need to understand the data characteristics, combine this understanding with the problem we are trying to solve and then decide which model to be selected for solving the problem.
- Machine learning algorithms are broadly of two types: models for supervised learning, which primarily focus on solving predictive problems and models for unsupervised learning, which solve descriptive problems.

Predictive models

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

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

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

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

The models which are used for prediction of the numerical value of the target feature of a data instance are known as regression models. Linear Regression andLogistic Regression models are popular regression models.

Descriptive models

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

Examples of clustering include
1. Customer grouping or segmentation based on social, demographic, ethnic, etc. factors
2. Grouping of music based on different aspects like genre, language, time-period, etc.
3. Grouping of commodities in an inventory

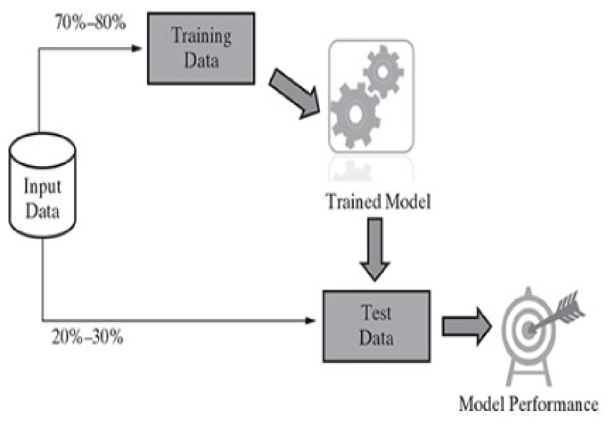
- The most popular model for clustering is k-Means.
- Descriptive models related to pattern discovery is used for market basket analysis of transactional data.
- In market basket analysis, based on the purchase pattern available in the transactional data, the possibility of purchasing one product based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

.

TRAINING A MODEL (FOR SUPERVISED LEARNING)

Holdout method

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

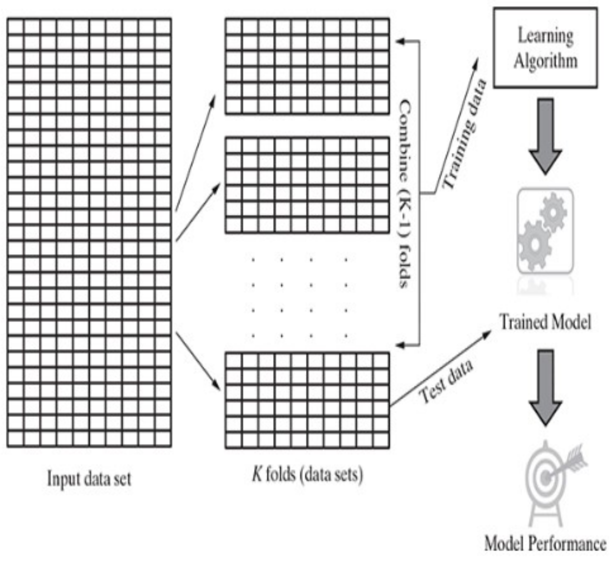


- Once the model is trained using the training data, the labels of the test data are predicted using the model's target function.
- Then the predicted value is compared with the actual value of the label.
- This is possible because the test data is a part of the input data with known labels.
- The performance of the model is in general measured by the accuracy of prediction of the label value.
- In certain cases, the input data is partitioned into three portions – a training and a test data, and a third validation data.
- The validation data is used in place of test data, for measuring the model performance.
- It is used in iterations and to refine the model in each iteration.
- The test data is used only for once, after the model is refined and finalized, to measure and report the final performance of the model as a reference for future learning efforts.
- An obvious problem in this method is that the division of data of different classes into the training and test data may not be proportionate.
- This situation is worse if the overall percentage of data related to certain classes is much less compared to other classes.
- This may happen despite the fact that random sampling is employed for test data selection.
- This problem can be addressed to some extent by applying stratified random sampling in place of sampling.
- In case of stratified random sampling, the whole data is broken into several homogenous groups or strata and a random sample is selected from each such stratum.
- This ensures that the generated random partitions have equal proportions of each class.

K-fold Cross-validation method

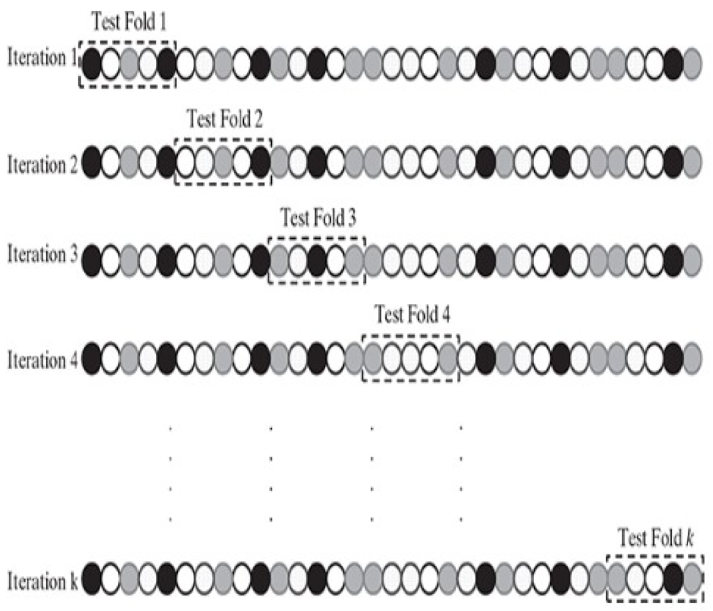
- Holdout method employing stratified random sampling approach still heads into issues in certain specific situations.
- Especially, the smaller data sets may have the challenge to divide the data of some of the classes
proportionally amongst training and test data sets.
- A special variant of holdout method, called repeated holdout, is sometimes employed to ensure the randomness of the composed data sets.

- In repeated holdout, several random holdouts are used to measure the model performance.
- In the end, the average of all performances is taken.
- As multiple holdouts have been drawn, the training and test data (and also validation data, in case it is drawn) are more likely to contain representative data from all classes and resemble the
  original input data closely.
- This process of repeated holdout is the basis of k-fold cross-validation technique.
- In k-fold cross-validation, the data set is divided into k- completely distinct or non-overlapping random partitions called folds.



The value of 'k' in k-fold cross-validation can be set to any number. However, there are two approaches which are extremely popular:
1. 10-fold cross-validation (10-fold CV)
2. Leave-one-out cross-validation (LOOCV)

10-fold cross-validation is by far the most popular approach.
In this approach, for each of the 10-folds, each comprising of approximately 10% of the data, one of the folds is used as the test data for validating model performance trained based on the remaining 9 folds (or 90% of the data). This is repeated 10 times, once for each of the 10 folds being used as the test data and the remaining folds as the training data.
The average performance across all folds is being reported.

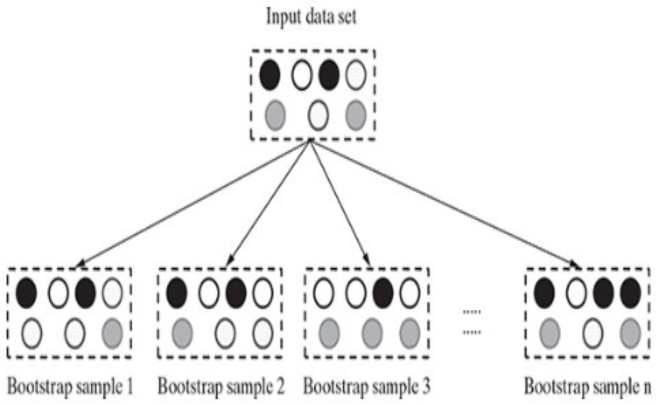


- Each of the circles resembles a record in the input data set whereas the different colors indicate the different classes that the records belong to.
- The entire data set is broken into 'k' folds – out of which one fold is selected in each iteration as the test data set.
- The fold selected as test data set in each of the 'k' iterations is different.
- Also, the circles resemble the records in the input data set, the contiguous circles represented as folds do not mean that they are subsequent records in the data set.
- This is more a virtual representation and not a physical representation. As, the records in a fold are drawn by using random sampling technique

Leave-one-out cross-validation (LOOCV) is an extreme case of k-fold cross-validation using one record or data instance at a time as a test data.
This is done to maximize the count of data used to train the model. It is obvious that the number of iterations for which it has to be run is equal to the total number of data in the input data set. Hence, obviously, it is computationally very expensive and not used much in practice.

Bootstrap sampling
Bootstrap sampling or simply bootstrapping is a popular way to identify training and test data sets from the input data set.
It uses the technique of Simple Random Sampling with Replacement (SRSWR), which is a well-known technique in sampling theory for drawing random samples.
k-fold cross-validation divides the data into separate partitions say 10 partitions in case of 10-fold cross-validation.
Then it uses data instances from partition as test data and the remaining partitions as training data.
Unlike this approach adopted in case of k-fold cross- validation, bootstrapping randomly picks data instances from the input data set, with the possibility of the same data instance to be picked multiple times.

This essentially means that from the input data set having ‘n’ data instances, bootstrapping can create one or more training data sets having ‘n’ data instances, some of the data instances being repeated multiple times.



The above figure briefly presents the approach followed in bootstrap sampling.
This technique is particularly useful in case of input data sets of small size, i.e. having very less number of data instances.

Lazy vs. Eager learner

- Eager learning follows the general principles of machine learning – it tries to construct a generalized, input-independent target function during the model training phase.
- It follows the typical steps of machine learning,i.e. abstraction and generalization and comes up with a trained model at the end of the learning phase.
- Hence,when the test data comes in for classification, the eager learner is ready with the model and doesn’t need to refer back to the training data.
- Eager learners take more time in the learning phase than the lazy learners.
- Some of the algorithms which adopt eager learning approach include Decision Tree, Support vector Machine, Neural Network,etc.
  Lazy learning, on the other hand, completely skips the abstraction and generalization processes, as in context of a typical machine learning process.
- In that respect, strictly speaking, lazy learner doesn’t ‘learn’ anything.
- It uses the training data in exact, and uses the knowledge to classify the unlabelled test data. Since lazy learning uses training data as-is, it is also known as rote learning (i.e. memorization technique based on repetition).
- Due to its heavy dependency on the given training data instance, it is also known as instance
  learning.
- They are also called non-parametric learning Lazy learners take very little time in training because not much of training actually happens.
- However, it takes quite some time in classification as for each tuple of test data, a comparison-based assignment of label happens.
- One of the most popular algorithm for lazy learning is k-nearest neighbor.

Parametric learning models have finite number of parameters. In case of non-parametric models, the number of parameters is potentially infinite.

Models such as Linear Regression and Support Vector Machine, since the coefficients form the learning parameters, they are fixed in size.
Hence, these models are clubbed as parametric. On the other hand, in case of models such as k-Nearest Neighbor (kNN) and decision tree, number of parameters grows with the size of the training data. Hence, they are considered as non-parametric learning models.

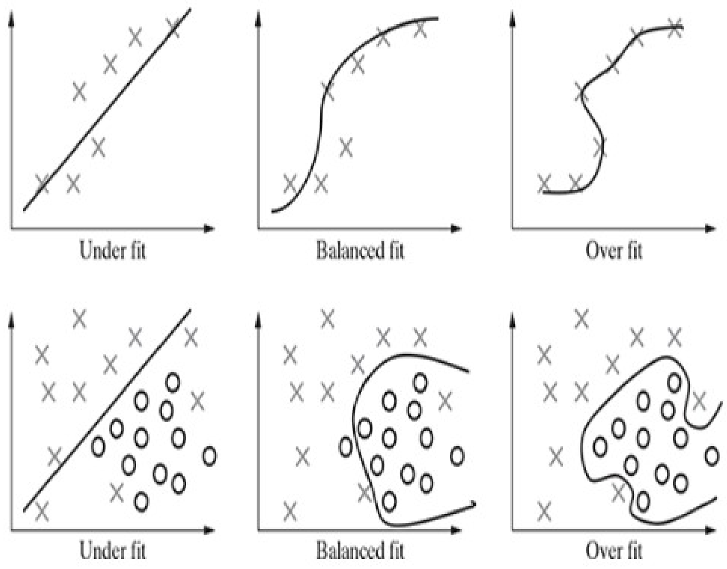
MODEL REPRESENTATION AND INTERPRETABILITY

- The goal of supervised machine learning is to learn or derive a target function which can best determine the target variable from the set of input variables.
- A key consideration in learning the target function from the training data is the extent of generalization.
- This is because the input data is just a limited, specific view and the new, unknown data in the test data set may be differing quite a bit from the training data.
- Fitness of a target function approximated by a learning algorithm determines how correctly it is able to classify a set of data it has never seen.

Underfitting

- If the target function is kept too simple, it may not be able to capture the essential nuances and represent the underlying data well.
- A typical case of underfitting may occur when trying to represent a non-linear data with a linear model
- Many times underfitting happens due to unavailability of sufficient training data.
- Underfitting results in both poor performance with training data as well as poor generalization to test data.

Underfitting can be avoided by
1. using more training data
2. reducing features by effective feature selection



Overfitting

- Overfitting refers to a situation where the model has been designed in such a way that it emulates the training data too closely.
- In such a case, any specific deviation in the training data, like noise or outliers, gets embedded in
  the model.
- It adversely impacts the performance of the model on the test data.
- Overfitting, in many cases, occur as a result of trying to fit an excessively complex model to closely match the training data.
- The target function, in these cases, tries to make sure all training data points are correctly partitioned by the decision boundary.
  However, more often than not, this exact nature is not replicated in the unknown test data set. Hence, the target function results in wrong classification in the test dataset.
- Overfitting results in good performance with training data set, but poor generalization and hence poor performance with test data set.

Overfitting can be avoided by
1. using re-sampling techniques like k-fold cross validation
2. hold back of a validation data set
3. remove the nodes which have little or no predictive power for the given machine learning problem.

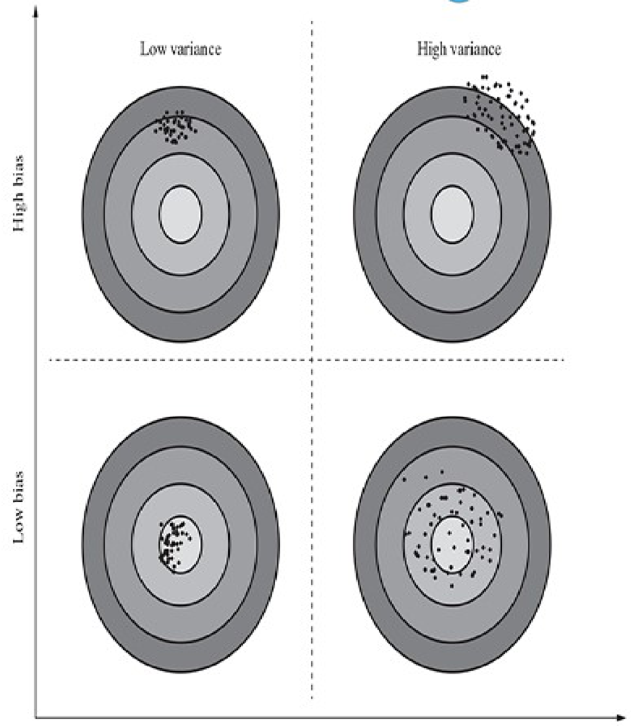
Both underfitting and overfitting result in poor classification quality which is reflected by low classification accuracy.

Bias – variance trade-off
In supervised learning, the class value assigned by the learning model built based on the training data may differ from the actual class value.
This error in learning can be of two types – errors due to 'bias' and error due to 'variance'.

Errors due to 'Bias'
- Errors due to bias arise from simplifying assumptions made by the model to make the target function less complex or easier to learn.
- In short, it is due to underfitting of the model.
- Parametric models generally have high bias making them easier to understand/interpret and faster to learn.
- These algorithms have a poor performance on data sets, which are complex in nature and do not align with the simplifying assumptions made by the algorithm.
- Underfitting results in high bias.

Errors due to 'Variance'
- Errors due to variance occur from difference in training data sets used to train the model. Different training data sets (randomly sampled from the input data set) are used to train the model.
- Ideally the difference in the data sets should not be significant and the model trained using different training data sets should not be too different.

- However, in case of overfitting, since the model closely matches the training data, even a small difference in training data gets magnified in the model.



- So, the problems in training a model can either happen because either (a) the model is too simple and hence fails to interpret the data grossly or (b) the model is extremely complex and magnifies even small differences in the training data.
- Increasing the bias will decrease the variance, and Increasing the variance will decrease the bias
- On one hand, parametric algorithms are generally seen to demonstrate high bias but low variance.
- On the other hand, non-parametric algorithms demonstrate low bias and high variance.
- The best solution is to have a model with low bias as well as low variance.
- However, that may not be possible in reality. Hence, the goal of supervised machine learning is to achieve a balance between bias and variance.
- The learning algorithm chosen and the user parameters which can be configured helps in striking a trade-off between bias and variance.
- For example, in a popular supervised algorithm k-Nearest Neighbors or kNN, the user configurable parameter 'k' can be used to do a trade-off between bias and variance.
- In one hand, when the value of 'k' is decreased, the model becomes simpler to fit and bias increases.
- On the other hand, when the value of 'k' is increased, the variance increases

EVALUATING PERFORMANCE OF A MODEL

Supervised learning – classification

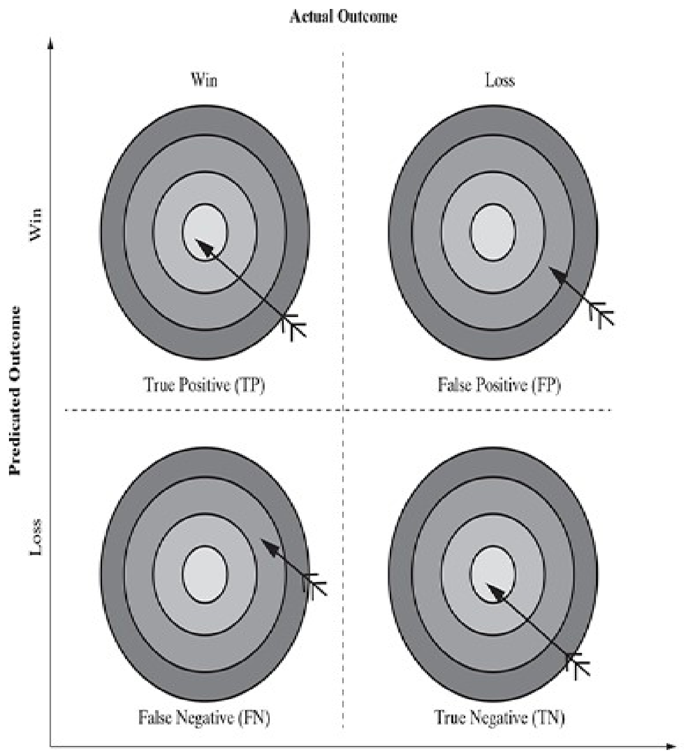
- In supervised learning, one major task is classification. The responsibility of the classification model is to assign class label to the target feature based on the value of the predictor features.

- For example, in the problem of predicting the win/loss in a cricket match, the classifier will assign a class value win/loss to target feature based on the values of other features like whether the team won the toss, number of spinners in the team, number of wins the team had in the tournament, etc.
- To evaluate the performance of the model, the number of correct classifications or predictions made by the model has to be recorded.
- A classification is said to be correct if, say for example in the given problem, it has been predicted by the model that the team will win and it has actually won.
- Based on the number of correct and incorrect classifications or predictions made by a model, the accuracy of the model is calculated.
- If 99 out of 100 times the model has classified correctly, e.g. if in 99 out of 100 games what the model has predicted is same as what the outcome has been, then the model accuracy is said to be 99%.
- However, it is quite relative to say whether a model has performed well just by looking at the accuracy value.
- For example, 99% accuracy in case of a sports win predictor model may be reasonably good but the same number may not be acceptable as a good threshold when the learning problem deals with predicting a critical illness.
- In this case, even the 1% incorrect prediction may lead to loss of many lives.
- So the model performance needs to be evaluated in light of the learning problem in question. Also, in certain cases, erring on the side of caution may be preferred at the cost of overall accuracy.
- For that reason, we need to look more closely at the model accuracy and also at the same time look at other measures of performance of a model like sensitivity, specificity, precision, etc. T

There are four possibilities with regards to the cricket match win/loss prediction:
1. the model predicted win and the team won
2. the model predicted win and the team lost
3. the model predicted loss and the team won
4. the model predicted loss and the team lost

In this problem, the obvious class of interest is 'win'.

- ✓ The first case, i.e. the model predicted win and the team won is a case where the model has correctly classified data instances as the class of interest. These cases are referred as True Positive (TP) cases.
- ✓ The second case, i.e. the model predicted win and the team lost is a case where the model incorrectly classified data instances as the class of interest. These cases are referred as False Positive (FP) cases.
- ✓ The third case, i.e. the model predicted loss and the team won is a case where the model has incorrectly classified as not the class of interest. These cases are referred as False Negative (FN) cases.
- ✓ The fourth case, i.e. the model predicted loss and the team lost is a case where the model has correctly classified as not the class of interest. These cases are referred as True Negative (TN) cases



- For any classification model, model accuracy is given by total number of correct classifications (either as the class of interest, i.e. True Positive or as not the class of interest, i.e. True Negative) divided by total number of classifications done.

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

A matrix containing correct and incorrect predictions in the form of TPs, FPs, FNs and TNs is known as **confusion matrix.**
The win/loss prediction of cricket match has two classes of interest – win and loss.
For that reason it will generate a 2 × 2 confusion matrix. For a classification problem involving three classes, the confusion matrix would be 3 × 3, etc
Let's assume the confusion matrix of the win/loss prediction of cricket match problem to be as below:

	ACTUAL WIN	ACTUAL LOSS
Predicted Win	85	4
Predicted Loss	2	9

In context of the above confusion matrix, total count of TPs = 85, count of FPs = 4, count of FNs = 2 and count of TNs = 9.

$$\therefore \text{Model accuracy} = \frac{TP + TN}{TP + FP + FN + TN} = \frac{85 + 9}{85 + 4 + 2 + 9} = \frac{94}{100} = 94\%$$

The percentage of misclassifications is indicated using error rate which is measured as

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

$$\text{Error rate} = \frac{FP + FN}{TP + FP + FN + TN} = \frac{4 + 2}{85 + 4 + 2 + 9} = \frac{6}{100} = 6\%$$
$$= 1\text{–Model accuracy}$$

Sometimes, correct prediction, both TPs as well as TNs, may happen by mere coincidence.
Since these occurrences boost model accuracy, ideally it should not happen.
Kappa value of a model indicates the adjusted the model accuracy.
It is calculated using the formula below

$$\text{Kappa value (k)} = \frac{P(a) - P(p_r)}{1 - P(p_r)}$$

P(a) = Proportion of observed agreement between actual and predicted in overall data set

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

$P(p_r)$ = Proportion of expected agreement between actual and predicted data both in case of class of interest as well as the other classes

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

In context of the above confusion matrix, total count of TPs = 85, count of FPs = 4, count of FNs = 2 and count of TNs = 9.

$$\therefore P(a) = \frac{TP + TN}{TP + FP + FN + TN} = \frac{85 + 9}{85 + 4 + 2 + 9} = \frac{94}{100} = 0.94$$

$$P(p_r) = \frac{85 + 4}{85 + 4 + 2 + 9} \times \frac{85 + 2}{85 + 4 + 2 + 9} + \frac{2 + 9}{85 + 4 + 2 + 9} \times \frac{4 + 9}{85 + 4 + 2 + 9}$$

$$= \frac{89}{100} \times \frac{87}{100} + \frac{11}{100} \times \frac{13}{100} = 0.89 \times 0.87 + 0.11 \times 0.13 = 0.7886$$

$$\therefore k = \frac{0.94 - 0.7886}{1 - 0.7886} = 0.7162$$

- Kappa value can be 1 at the maximum, which represents perfect agreement between model's prediction and actual values.
- There are some measures of model performance which are more important than accuracy. Two such critical measurements are sensitivity and specificity of the model.

- The **sensitivity** of a model measures the proportion of TP examples or positive cases which were correctly classified. It is measured as

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

- **Specificity** is also another good measure to indicate a good balance of a model being excessively conservative or excessively aggressive.
- Specificity of a model measures the proportion of negative examples which have been correctly classified

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

- There are two other performance measures of a supervised learning model which are similar to
  sensitivity and specificity.
- These are **Precision** and **Recall**.
- **Precision** gives the proportion of positive predictions which are truly positive,
- **Recall** gives the proportion of TP cases over all actually positive cases.

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

- Precision indicates the reliability of a model in predicting a class of interest.
- When the model is related to win / loss prediction of cricket, precision indicates how often it predicts the win correctly.
- **Recall** indicates the proportion of correct prediction of positives to the total number of positives.
- In case of win/loss prediction of cricket, recall resembles what proportion of the total wins were predicted correctly.

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

F-measure

- F-measure is another measure of model performance which combines the precision and recall.
- It takes the harmonic mean of precision and recall as calculated as

$$F\text{-measure} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

- As a combination of multiple measures into one, F-score gives the right measure using which performance of different models can be compared.
- However, one assumption the calculation is based on is that precision and recall have equal weight, which may not always be true in reality. In certain problems, the disease prediction problems, e.g., precision may be given far more weightage.
- In that case, different weightages may be assigned to precision and recall.

- However, there may be a serious dilemma regarding what value to be adopted for each and what is the basis for the specific value adopted.

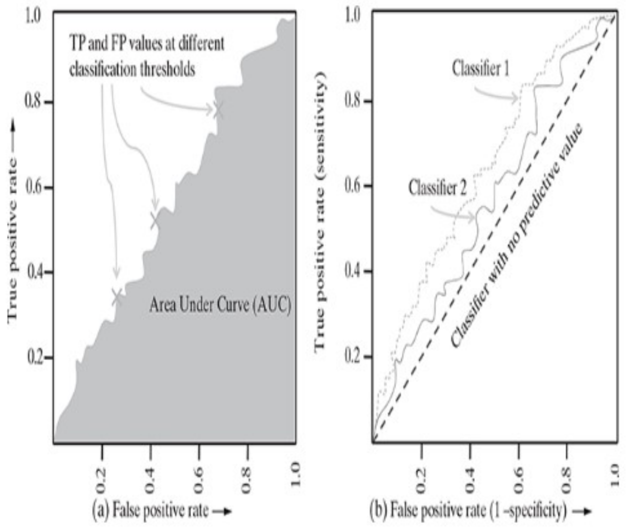
Receiver operating characteristic (ROC) curves

- Though accuracy is the most popular measure, there are quite a number of other measures to evaluate the performance of a supervised learning model.
- Visualization is an easier and more effective way to understand the model performance.
- It also helps in comparing the efficiency of two models.
- Receiver Operating Characteristic (ROC) curve helps in visualizing the performance of a classification model.
- It shows the efficiency of a model in the detection of true positives while avoiding the occurrence of false positives

$$\text{True Positive Rate TPR} = \frac{TP}{TP + FN}$$

$$\text{False Positive Rate FPR} = \frac{FP}{FP + TN}$$

- In the ROC curve, the FP rate is plotted (in the horizontal axis) against true positive rate (in the vertical axis) at different classification thresholds.
- If we assume a lower value of classification threshold, the model classifies more items as positive.
- Hence, the values of both False Positives and True Positives increase.
- The area under curve (AUC) value is the area of the two-dimensional space under the curve extending from (0, 0) to (1, 1), where each point on the curve gives a set of true and false positive values at a specific classification threshold.
- This curve gives an indication of the predictive quality of a model.
- AUC value ranges from 0 to 1, with an AUC of less than 0.5 indicating that the classifier has no predictive ability



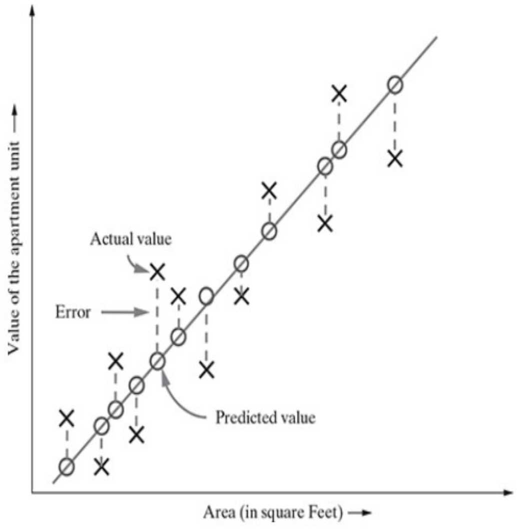
- Figure b shows the curves of two classifiers – classifier 1 and classifier 2 The AUC of classifier 1 is more than the AUC of classifier 2.

- So, we can draw the inference that classifier 1 is better than classifier 2.

✓ Interpretation of the predictive values from 0.5 to 1.0 is given below:
0.5 – 0.6 ➔ Almost no predictive ability
0.6 – 0.7 ➔ Weak predictive ability
0.7 – 0.8 ➔ Fair predictive ability
0.8 – 0.9 ➔ Good predictive ability
0.9 – 1.0 ➔ Excellent predictive ability

Supervised learning – regression

- A well-fitted regression model churns out predicted values close to actual values.
- Hence, a regression model which ensures that the difference between predicted and actual values is low can be considered as a good model.



- The Figure represents a very simple problem of real estate value prediction solved using linear regression model.
- If ‘area’ is the predictor variable (say x) and ‘value’ is the target variable (say y), the linear regression model can be represented in the form:

$$y = \alpha + \beta x$$

- For a certain value of x, say x̂ , the value of y is predicted as ŷ whereas the actual value of y is Y (say).
- The distance between the actual value and the fitted or predicted value, i.e. ŷ is known as residual.
- The regression model can be considered to be fitted well if the difference between actual and predicted value, i.e. the residual value is less.

R-squared is a good measure to evaluate the model fitness. It is also known as the coefficient of determination, or for multiple regression, the coefficient of multiple determination.
The R-squared value lies between 0 to 1 (0%–100%) with a larger value representing a better fit. It is calculated as:

$$R^2 = \frac{SST - SSE}{SST}$$

Sum of Squares Total (SST) = squared differences of each observation from the overall mean

$$=\sum_{i=1}^{n}(y_i - \bar{y})^2$$

where ȳ is the mean.
Sum of Squared Errors (SSE) (of prediction) = sum of the squared residuals

$= \sum_{i=1}^{n}(Y_i - \hat{y})^2$ where $\hat{y}_i$ is the predicted value of y_i and Y_i is the actual value of y_i.

Unsupervised learning – clustering

- Clustering algorithms try to reveal natural groupings amongst the data sets.
- However, it is quite tricky to evaluate the performance of a clustering algorithm.
- Clustering, is very subjective and whether the cluster is good or bad is open for interpretations.
- It was noted, 'clustering is in the eye of the beholder'.
- This stems from the two inherent challenges which lie in the process of clustering:
  1. It is generally not known how many clusters can be formulated from a particular data set. It is completely open-ended in most cases and provided as a user input to a clustering algorithm.
  2. Even if the number of clusters is given, the same number of clusters can be formed with different groups of data instances.
- In a more objective way, it can be said that a clustering algorithm is successful if the clusters identified using the algorithm is able to achieve the right results in the overall problem domain.
- For example, if clustering is applied for identifying customer segments for a marketing campaign of a new product launch, the clustering can be considered successful only if the marketing campaign ends with a success,i.e. it is able to create the right brand recognition resulting in steady revenue from new product sales.
- However, there are couple of popular approaches which are adopted for cluster quality evaluation.
  1. **Internal evaluation**
  In this approach, the cluster is assessed based on the underlying data that was clustered. The internal evaluation methods generally measure cluster quality based on homogeneity of data belonging to the same cluster and heterogeneity of data belonging to different clusters. The homogeneity/heterogeneity is decided by some similarity measure. For example, silhouette coefficient, which is one of the most popular internal evaluation methods, uses distance (Euclidean or Manhattan distances most commonly used) between data elements as a similarity measure. The value of silhouette width ranges between –1 and +1, with a high value indicating high intra-cluster homogeneity and inter-cluster heterogeneity.For a data set clustered into 'k' clusters, silhouette width is calculated as:

$$\text{Silhouette width} = \frac{b(i) - a(i)}{\max\{a(i),\ b(i)\}}$$

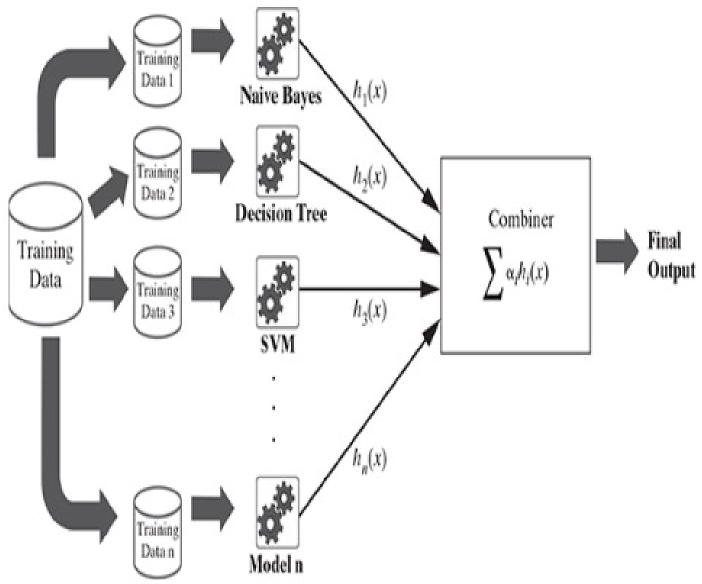
a(i) is the average distance between the i th data instance and all other data instances belonging to the same cluster and b(i) is the lowest average distance between the i-the data instance and data instances of all other clusters

2. External evaluation
In this approach, class label is known for the data set subjected to clustering. However, quite obviously, the known class labels are not a part of the data used in clustering. The cluster algorithm is assessed based on how close the results are compared to those known class labels. For example, purity is one of the most popular measures of cluster algorithms – evaluates the extent to which clusters contain a single class. For a data set having 'n' data instances and 'c' known class labels which generates 'k' clusters, purity is measured as:

$$\text{Purity} = \frac{1}{n}\sum_{k} \max(k \cap c)$$

IMPROVING PERFORMANCE OF A MODEL

- One effective way to improve model performance is by tuning model parameter.
- Model parameter tuning is the process of adjusting the model fitting options.
- For example, in the popular classification model k-Nearest Neighbour (kNN), using different values of 'k' or the number of nearest neighbours to be considered, the model can be tuned.
- In the same way, a number of hidden layers can be adjusted to tune the performance in neural networks model.
- Most machine learning models have at least one parameter which can be tuned.
  As an alternate approach of increasing the performance of one model, several models may be
  combined together.
- The models in such combination are complimentary to each other, i.e. one model may learn
  one type data sets well while struggle with another type of data set.
- Another model may perform well with the data set which the first one struggled with.
- This approach of combining different models with diverse strengths is known as ensemble.



- Ensemble helps in averaging out biases of the different underlying models and also reducing the variance.

- Ensemble methods combine weaker learners to create stronger ones.
- A performance boost can be expected even if models are built as usual and then ensembled.
  Following are the typical steps in ensemble process:
  1.Build a number of models based on the training data
  2.For diversifying the models generated, the training data subset can be varied using the allocation function. Sampling techniques like bootstrapping may be used to generate unique training data sets.
  3.Alternatively, the same training data may be used but the models combined are quite varying, e.g, SVM, neural network, kNN, etc.
  4.The outputs from the different models are combined using a combination function. A very simple strategy of combining, say in case of a prediction task using ensemble, can be majority voting of the different models combined. For example, 3 out of 5 classes predict 'win' and 2 predict 'loss' – then the final outcome of the ensemble using majority vote would be a 'win'.

- One of the earliest and most popular ensemble models is **bootstrap aggregating or bagging.** Bagging uses bootstrap sampling method to generate multiple training data sets.
- These training data sets are used to generate (or train) a set of models using the same learning algorithm.
- Then the outcomes of the models are combined by majority voting (classification) or by average (regression).
- Bagging is a very simple ensemble technique which can perform really well for unstable learners like a decision tree, in which a slight change in data can impact the outcome of a model significantly.
- Just like bagging, boosting is another key ensemble-based technique.
- In this type of ensemble, weaker learning models are trained on resampled data and the outcomes are combined using a weighted voting approach based on the performance of different models
- **Adaptive boosting or AdaBoost** is a special variant of boosting algorithm.
- It is based on the idea of generating weak learners and slowly learning
- **Random forest** is another ensemble-based technique.
- It is an ensemble of decision trees – hence the name random forest to indicate a forest of decision trees.

Feature Engineering

- The key aspect which plays a critical role in solving any machine learning problem – feature
  engineering.
- Feature engineering is a critical preparatory process in machine learning.
- It is responsible for taking raw input data and converting that to well-aligned features which are ready to be used by the machine learning models.
- **Unstructured data** is raw, unorganized data which doesn't follow a specific format or hierarchy.

- Typical examples of unstructured data include text data from social networks, e.g. Twitter, Facebook,etc. or data from server logs, etc

Feature

- A feature is an attribute of a data set that is used in a machine learning process.
- Selection of the subset of features which are meaningful for machine learning is a sub-area of feature engineering which draws a lot of research interest.
- The features in a data set are also called its dimensions.
- So a data set having 'n' features is called an n-dimensional data set.

Feature Engineering

- Feature engineering refers to the process of translating a data set into features such that these features are able to represent the data set more effectively and result in a better learning performance.
- Feature engineering is an important pre-processing step for machine learning. It has two major elements:
    1. Feature Transformation
    2. Feature subset selection
- **Feature transformation** transforms the data – structured or unstructured, into a new set of features which can represent the underlying problem which machine learning is trying to solve.
- There are two variants of feature transformation:
    1. Feature construction
    2. Feature extraction

  Both are sometimes known as Feature discovery.
- **Feature construction** process discovers missing information about the relationships between features and augments the feature space by creating additional features.
- Hence, if there are 'n' features or dimensions in a data set, after feature construction 'm' more features or dimensions may get added. So at the end, the data set will become 'n + m' dimensional.
- **Feature extraction** is the process of extracting or creating a new set of features from the original set of features using some functional mapping.
- In **Feature subset selection (or simply feature selection)** no new feature is generated. The objective of feature selection is to derive a subset of features from the full feature set which is most meaningful in the context of a specific machine learning problem.

FEATURE TRANSFORMATION

- All available attributes of the data set are used as features and the problem of identifying the important features is left to the learning model.
- This is definitely not a feasible approach, particularly for certain domains e.g. medical image classification, text categorization, etc.
- Feature transformation is used as an effective tool for dimensionality reduction and hence for boosting learning model performance.
- Broadly, there are two distinct goals of feature transformation:
    1. Achieving best reconstruction of the original features in the data set
    2. Achieving highest efficiency in the learning task

Feature construction

- Feature construction involves transforming a given set of input features to generate a new set of more powerful features.
- For example a real estate data set having details of all apartments sold in a specific region. The data set has three features – apartment length, apartment breadth, and price of the apartment.
- If it is used as an input to a regression problem, such data can be training data for the regression model.
- So given the training data, the model should be able to predict the price of an apartment whose price is not known or which has just come up for sale.
- However, instead of using length and breadth of the apartment as a predictor, it is much convenient and makes more sense to use the area of the apartment, which is not an existing feature of the data set.
- So such a feature, namely apartment area, can be added to the data set.
- Here the three-dimensional data set is tranformed to a four-dimensional data set, with the newly 'discovered' feature apartment area being added to the original data set.
- There are certain situations where feature construction is an essential activity. These situations are
  - when features have categorical value and machine learning needs numeric value inputs
  - when features having numeric (continuous) values and need to be converted to ordinal values
  - when text-specific feature construction needs to be done

Encoding categorical (nominal) variables

- Suppose there are three features – City of origin, Parents athlete, and Chance of win, which are categorical in nature and cannot be used by any machine learning task.
- In this case, feature construction can be used to create new dummy features which are usable by machine
- learning algorithms. Since the feature 'City of origin' has three unique values namely City A, City B, and City C, three dummy features namely origin_ city_A, origin_city_B, and origin_city_C is created.
- In the same way, dummy features parents_athlete_Y and parents_athlete_N are created for feature 'Parents athlete' and win_chance_Y and win_chance_N are created for feature 'Chance of win'.
- The dummy features have value 0 or 1 based on the categorical value for the original feature in that row.
- Features such as 'Parents athlete' and 'Chance of win' in the original data set can have two values only.
- So creating two features from them is a kind of duplication, since the value of one feature can be decided from the value of the other.
- To avoid this duplication, leave one feature and eliminate the other,

Age (Years)	City of origin	Parents athlete	Chance of win
18	City A	Yes	Y
20	City B	No	Y
23	City B	Yes	Y
19	City A	No	N
18	City C	Yes	N
22	City B	Yes	Y

(a)

Age (Years)	origin_city_A	origin_city_B	origin_city_C	parents_athlete_Y	parents_athlete_N	win_chance_Y	win_chance_N
18	1	0	0	1	0	1	0
20	0	1	0	0	1	1	0
23	0	1	0	1	0	1	0
19	1	0	0	0	1	0	1
18	0	0	1	1	0	0	1
22	0	1	0	1	0	1	0

(b)

Age (Years)	origin_city_A	origin_city_B	origin_city_C	parents_athlete_Y	win_chance_Y
18	1	0	0	1	1
20	0	1	0	0	1
23	0	1	0	1	1
19	1	0	0	0	0
18	0	0	1	1	0
22	0	1	0	1	1

(c)

Encoding categorical (ordinal) variables

- For a example In a student data set. Assume that there are three variable – science marks, maths marks and grade .
- The grade is an ordinal variable with values A, B, C, and D. To transform this variable to a numeric variable, a feature num_grade can be created ,used as mapping a numeric value against each ordinal value.
- In the context of the current example, grades A, B, C, and D is mapped to values 1, 2, 3, and 4 in the transformed variable .

marks_science	marks_maths	Grade
78	75	B
56	62	C
87	90	A
91	95	A
45	42	D
62	57	B

(a)

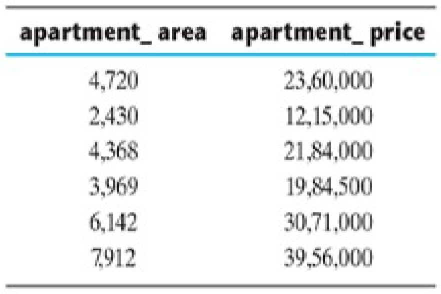
marks_science	marks_maths	num_grade
78	75	2
56	62	3
87	90	1
91	95	1
45	42	4
62	57	2

(b)

Feature construction (encoding ordinal variables)

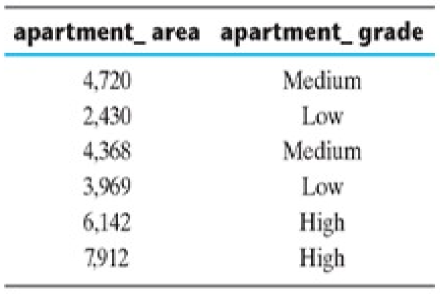
Transforming numeric (continuous) features to categorical features

- Sometimes there is a need of transforming a continuous numerical variable into a categorical variable.
- For example, we may want to treat the real estate price prediction problem, which is a regression problem, as a real estate price category prediction, which is a classification problem.
- In that case, we can 'bin' the numerical data into multiple categories based on the data range.
- In the context of the real estate price prediction example, the original data set has a numerical feature apartment_price
- It can be transformed to a categorical variable price-grade .



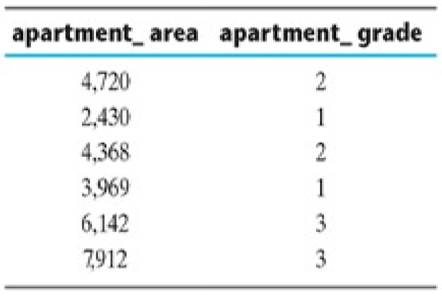
apartment_ area	apartment_ price
4,720	23,60,000
2,430	12,15,000
4,368	21,84,000
3,969	19,84,500
6,142	30,71,000
7,912	39,56,000

(a)



apartment_ area	apartment_ grade
4,720	Medium
2,430	Low
4,368	Medium
3,969	Low
6,142	High
7,912	High

(b)



apartment_ area	apartment_ grade
4,720	2
2,430	1
4,368	2
3,969	1
6,142	3
7,912	3

(c)

Text-specific feature construction

- In the current world, text is the most predominant medium of communication. Whether it is a about social networks like Facebook or micro-blogging channels like Twitter or emails or short messaging services such as Whatsapp, text plays a major role in the flow of information.
- All machine learning models need numerical data as input. So the text data in the data sets need to be transformed into numerical features.
- Text data, or corpus which is the more popular keyword, is converted to a numerical representation following a process is known as **vectorization**.
- In this process, word occurrences in all documents belonging to the corpus are consolidated in the form of bag-of-words.
- There are three major steps that are followed:
  1. tokenize
  2. count
  3. Normalize

- In order to tokenize a corpus, the blank spaces and punctuations are used as delimiters to separate out the words, or tokens.
- Then the number of occurrences of each token is counted, for each document.
- Lastly, tokens are weighted with reducing importance when they occur in the majority of the documents.
- A matrix is then formed with each token representing a column and a specific document of the corpus representing each row.
- Each cell contains the count of occurrence of the token in a specific document. This matrix is known as a **document-term matrix (also known as a term-document matrix)**.

This	House	Build	Feeling	Well	Theatre	Movie	Good	Lonely	...
2	1	1	0	0	1	1	1	0	
0	0	0	1	1	0	0	0	0	
1	0	0	2	1	1	0	0	1	
0	0	0	0	1	0	1	1	0	
.	.	.	.	.	.	.	.	.	
.	.	.	.	.	.	.	.	.	
.	.	.	.	.	.	.	.	.	

Feature construction(Text-Specific)

Feature extraction

In feature extraction, new features are created from a combination of original features. Some of the commonly used operators for combining the original features include

1. For Boolean features: Conjunctions, Disjunctions, Negation, etc.
2. For nominal features: Cartesian product, M of N, etc.
3. For numerical features: Min, Max, Addition, Subtraction,Multiplication, Division, average, Equivalence, Inequality, etc.

$Feat_A$	$Feat_B$	$Feat_C$	$Feat_D$
34	34.5	23	233
44	45.56	11	3.44
78	22.59	21	4.5
22	65.22	11	322.3
22	33.8	355	45.2
11	122.32	63	23.2

$\rightarrow$

$Feat_1$	$Feat_2$
41.25	185.80
54.20	53.12
43.73	35.79
65.30	264.10
37.02	238.42
113.39	167.74

$$Feat_1 = 0.3 \times Feat_A + 0.9 \times Feat_A$$
$$Feat_2 = Feat_A + 0.5\ Feat_B + 0.6 \times Feat_C$$

Feature Extraction

The most popular feature extraction algorithms used in machine learning:

1. Principal Component Analysis
2. Singular value decomposition
3. Linear Discriminant Analysis

Principal Component Analysis

- Every data set, has multiple attributes or dimensions – many of which might have similarity with each other.
- For example, the height and weight of a person, in general, are quite related. If the height is more, generally weight is more and vice versa.
- So if a data set has height and weight as two of the attributes, obviously they are expected to be having quite a bit of similarity.
- In general, any machine learning algorithm performs better as the number of related attributes or features reduced.
- In other words, a key to the success of machine learning lies in the fact that the features are less in number as well as the similarity between each other is very less.
- This is the main guiding philosophy of principal component analysis (PCA) technique of feature extraction.

The objective of PCA is to make the transformation in such a way that

1. The new features are distinct, i.e. the covariance between the new features, i.e. the principal components is 0.
2. The principal components are generated in order of the variability in the data that it captures. Hence, the first principal component should capture the maximum variability, the second principal component should capture the next highest variability etc.
3. The sum of variance of the new features or the principal components should be equal to the sum of variance of the original features.

PCA works based on a process called eigenvalue decomposition of a covariance matrix of a data set. Below are the steps to be followed:

1. First, calculate the covariance matrix of a data set.
2. Then, calculate the eigenvalues of the covariance matrix.
3. The eigenvector having highest eigenvalue represents the direction in which there is the highest variance. So this will help in identifying the first principal component.
4. The eigenvector having the next highest eigenvalue represents the direction in which data has the highest remaining variance and also orthogonal to the first direction. So this helps in identifying the second principal component.
5. Like this, identify the top 'k' eigenvectors having top 'k' eigenvalues so as to get the 'k' principal components.

-

  Singular value decomposition

  Singular value decomposition (SVD) is a matrix factorization technique commonly used in linear algebra.

  SVD of a matrix A (m × n) is a factorization of the form:

$$A = U\sum V$$

- where, U and V are orthonormal matrices, U is an m × m unitary matrix, V is an n × n unitary matrix and ∑ is an m × n rectangular diagonal matrix.
- The diagonal entries of ∑ are known as singular values of matrix A.
  The columns of U and V are called the left-singular and right-singular vectors of matrix A, respectively.
  SVD is generally used in PCA, once the mean of each variable has been removed.
- Since it is not always advisable to remove the mean of a data attribute, especially when the data set is sparse (as in case of text data),
- SVD is a good choice for dimensionality reduction in those situations.
- SVD of a data matrix is expected to have the properties highlighted below:

1. Patterns in the attributes are captured by the right-singular vectors, i.e. the columns of V.
2. Patterns among the instances are captured by the left-singular, i.e. the columns of U.
3. Larger a singular value, larger is the part of the matrix A that it accounts for and its associated vectors.
4. New data matrix with ‘k’ attributes is obtained using the equation

$$D' = D \times [v_1, v_2, \dots , v_k]$$

- Thus, the dimensionality gets reduced to k SVD is often used in the context of text data.

Linear Discriminant Analysis

- Linear discriminant analysis (LDA) is another commonly used feature extraction technique like PCA or SVD.
- The objective of LDA is similar to the sense that it intends to transform a data set into a lower dimensional feature space.
- However, unlike PCA, the focus of LDA is not to capture the data set variability.
- Instead, LDA focuses on class separability, i.e. separating the features based on class separability so as to avoid over-fitting of the machine learning model.
- Unlike PCA that calculates eigenvalues of the covariance matrix of the data set,
- LDA calculates eigenvalues and eigenvectors within a class and inter-class scatter matrices.
- Below are the steps to be followed:
  1. Calculate the mean vectors for the individual classes.
  2. Calculate intra-class and inter-class scatter matrices.
  3. Calculate eigenvalues and eigenvectors for S_W^{-1} and S_B , where S_W is the intra-class scatter matrix and S_B is the inter-class scatter matrix

$$S_W = \sum_{i=1}^{c} S_i;$$

$$S_i = \sum_{x \in D_i}^{n} (x - m_i)(x - m_i)^T$$

where, m_i is the mean vector of the i-th class

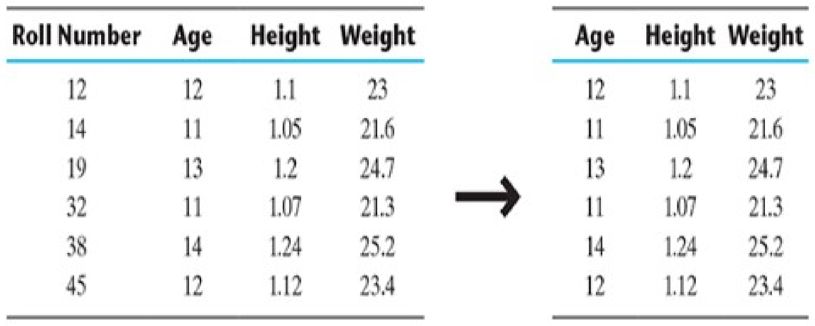
$$S_B = \sum_{i=1}^{c} N_i \,(m_i - m)\,(m_i - m)^T$$

where, mi is the sample mean for each class, m is the overall mean of the data set, Ni is the sample size of each class

4. Identify the top ‘k’ eigenvectors having top ‘k’ eigenvalues

FEATURE SUBSET SELECTION

- Feature selection is the most critical pre-processing activity in any machine learning project.
- It intends to select a subset of system attributes or features which makes a most meaningful contribution in a machine learning activity.
- To predict the weight of students based on past information about similar students, which is captured in a ‘student weight’ data set.
- The student weight data set has features such as Roll Number, Age, Height, and Weight.
- Roll number can have no bearing,whatsoever, in predicting student weight.
- So feature Roll number can be eliminated and build a feature subset to be considered in this machine learning problem.
- The subset of features is expected to give better results than the full set.



Roll Number	Age	Height	Weight
12	12	1.1	23
14	11	1.05	21.6
19	13	1.2	24.7
32	11	1.07	21.3
38	14	1.24	25.2
45	12	1.12	23.4

→

Age	Height	Weight
12	1.1	23
11	1.05	21.6
13	1.2	24.7
11	1.07	21.3
14	1.24	25.2
12	1.12	23.4

Issues in high-dimensional data

- High-dimensional’ refers to the high number of variables or attributes or features present in certain data sets, more so in the domains like DNA analysis, geographic information systems (GIS), social networking, etc.
- The high-dimensional spaces often have hundreds or thousands of dimensions or attributes, e.g. DNA microarray data can have up to 450,000 variables (gene probes)..
- To get insight from such high-dimensional data may be a big challenge for any machine learning algorithm.
- On one hand, very high quantity of computational resources and high amount of time will be required.
- On the other hand the performance of the model – both for supervised and unsupervised machine learning task, also degrades sharply due to unnecessary noise in the data.
- Also, a model built on an extremely high number of features may be very difficult to understand. For this reason, it is necessary to take a subset of the features instead of the full set.

The objective of feature selection is three-fold:

1. Having faster and more cost-effective (i.e. less need for computational resources) learning model
2. Improving the efficiency of the learning model
3. Having a better understanding of the underlying model that generated the data

Key drivers of feature selection – feature relevance and redundancy

Feature relevance

- In supervised learning, the input data set which is the training data set, has a class label attached. A model is inducted based on the training data set – so that the inducted model can assign class labels to new, unlabelled data.
- Each of the predictor variables, is expected to contribute information to decide the value of the class label.
- In case a variable is not contributing any information, it is said to be **irrelevant**.
- In case the information contribution for prediction is very little, the variable is said to be **weakly relevant.**
- Remaining variables, which make a significant contribution to the prediction task are said to be **strongly relevant variables**.
- In unsupervised learning, there is no training data set or labelled data.
- Grouping of similar data instances are done and similarity of data instances are evaluated based on the value of different variables.
- Certain variables donot contribute any useful information for deciding the similarity and dissimilarity of data instances.
- Hence, those variables make no significant information contribution in the grouping process. These variables are marked as **irrelevant variables**.
- For example , in the student data set Roll number of a student doesn't contribute any significant information in predicting what the Weight of a student would be.

- So, in context of the supervised task of predicting student Weight or the unsupervised task of grouping students with similar academic merit, the variable Roll number is quite irrelevant.
- Any feature which is irrelevant in the context of a machine learning task is a candidate for rejection when a subset of features are to be selected.
- weakly relevant features are to be rejected or not must be considered on a case-to-case basis.

Feature redundancy

- A feature may contribute information which is similar to the information contributed by one or more other features.
- For example, in the weight prediction problem , both the features Age and Height contribute similar information.
- This is because with an increase in Age, Weight is expected to increase. Similarly, with the increase of Height also Weight is expected to increase. Also, Age and Height increase with each other.
- So, in context of the Weight prediction problem, Age and Height contribute similar information.
- In other words, irrespective of whether the feature height is present as a part of the feature subset, the learning model will give almost same results.
- In the same way, without age being part of the predictor variables, the outcome of the learning model will be more or less same.

- In this kind of a situation when one feature is similar to another feature, the feature is said to be potentially redundant in the context of the learning problem.
- All features having potential redundancy are candidates for rejection in the final feature subset. Only a small number of representative features out of a set of potentially redundant features are considered for being a part of the final feature subset.
- So, the main objective of feature selection is to remove all features which are irrelevant and take a representative subset of the features which are potentially redundant.
- This leads to a meaningful feature subset in context of a specific learning task.

Measures of feature relevance and redundancy
Measures of feature relevance

- Feature relevance is to be gauged by the amount of information contributed by a feature.
- For supervised learning, mutual information is considered as a good measure of information contribution of a feature to decide the value of the class label.
- Higher the value of mutual information of a feature, more relevant is that feature.
- Mutual information can be calculated as follows:

$$MI(C, f) = H(C) + H(f) - H(C, f)$$

where, marginal entropy of the class, H(C) =

$$-\sum_{i=1}^{k} p(C_i)\log_2 p(C_i)$$

marginal entropy of the feature ‘x’, H(f) =

$$-\sum_{c} p(f = x)\log_2 p(f = x)$$

and K = number of classes, C = class variable, f = feature set that take discrete values.

- In case of unsupervised learning, there is no class variable. Hence, feature-to-class mutual information cannot be used to measure the information contribution of the features.
- In case of unsupervised learning, the entropy of the set of features without one feature at a time is calculated for all the features.
- Then, the features are ranked in a descending order of information gain from a feature and top ‘β’ percentage (value of ‘β’ is a design parameter of the algorithm) of features are selected as relevant features.
- The entropy of a feature f is calculated using Shannon’s formula below:

$$H(f) = -\sum_{x} p(f = x)\log_2 p(f = x)$$

$\sum_{x}$ is used only for features that take discrete values.

For continuous features, it should be replaced by discretization performed first to estimate probabilities p(f = x).

Measures of Feature redundancy
Feature redundancy, is based on similar information contribution by multiple features.
There are multiple measures of similarity of information contribution, salient ones being

1. Correlation-based measures

2. Distance-based measures, and
3. Other coefficient-based measure

1. Correlation-based similarity measure

- Correlation is a measure of linear dependency between two random variables.
- Pearson's product moment correlation coefficient is one of the most popular and accepted measures of correlation between two random variables.
- For two random feature variables F_1 and F_2 ,
- Pearson correlation coefficient is defined as:

$$\alpha = \frac{cov(F_1, F_2)}{\sqrt{var(F_1).var(F_2)}}$$

$$cov(F_1, F_2) = \sum (F_{1_i} - \overline{F_1}).(F_{2_i} - \overline{F_2})$$

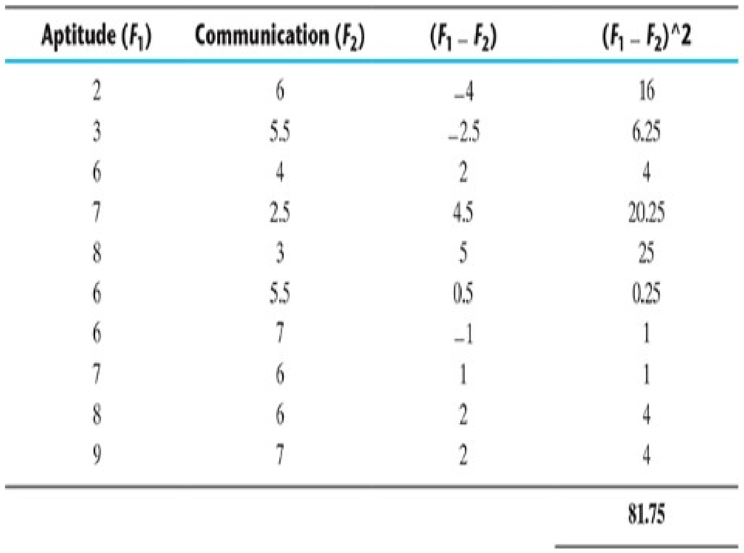
$$var(F_1) = \sum (F_{1_i} - \overline{F_1})^2, \text{where } \overline{F_1} = \frac{1}{n}.\sum F_{1_i}$$

$$var(F_2) = \sum (F_{2_i} - \overline{F_2})^2, \text{where } \overline{F_2} = \frac{1}{n}.\sum F_{2_i}$$

- Correlation values range between +1 and –1. A correlation of 1 (+ / –) indicates perfect correlation, i.e. the two features having a perfect linear relationship.
- In case the correlation is 0, then the features seem to have no linear relationship.
- Generally, for all feature selection problems, a threshold value is adopted to decide whether two features have adequate similarity or not

Distance-based similarity measure

- The most common distance measure is the Euclidean distance, which, between two features F_1 and F_2 are calculated as:
- $$d(F_1,\ F_2) = \sqrt{\sum_{i=1}^{n} (F_{1_i} - F_{2_i})^2}$$
  where F_1 and F_2 are features of an n-dimensional data set.
- For example



Aptitude (F_1)	Communication (F_2)	($F_1 - F_2$)	($F_1 - F_2$)^2
2	6	–4	16
3	5.5	–2.5	6.25
6	4	2	4
7	2.5	4.5	20.25
8	3	5	25
6	5.5	0.5	0.25
6	7	–1	1
7	6	1	1
8	6	2	4
9	7	2	4
			81.75

- A more generalized form of the Euclidean distance is the Minkowski distance, measured as

$$d(F_1, F_2) = \sqrt{\sum_{i=1}^{n}(F_{1_i} - F_{2_i})^r}$$

- At r = 1, it takes the form of Manhattan distance (also called L norm), as shown below:

$$d(F_1, F_2) = \sum_{i=1}^{n}|F_{1_i} - F_{2_i}|$$

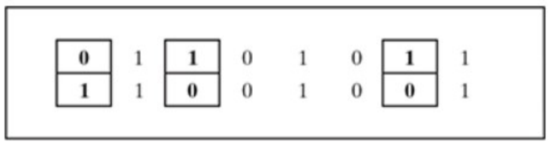
- A specific example of Manhattan distance, used more frequently to calculate the distance between binary vectors is the **Hamming distance**. For example, the Hamming distance between two vectors 01101011 and 11001001 is 3

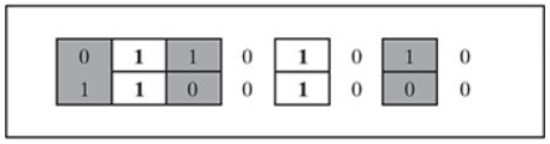
Other similarity measures

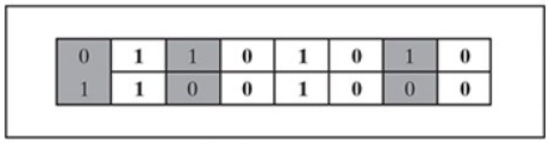
- **Jaccard index/coefficient** is used as a measure of similarity between two features.
- The Jaccard distance, a measure of dissimilarity between two features, is complementary of Jaccard index.
- For two features having binary values, Jaccard index is measured as

$$J = \frac{n_{11}}{n_{01} + n_{10} + n_{11}}$$

- where, n_{11} = number of cases where both the features have value 1
- n_{01} = number of cases where the feature 1 has value 0 and feature 2 has value 1
- n_{10} = number of cases where the feature 1 has value 1 and feature 2 has value 0
- Jaccard distance, d_J = 1 - J


(a) Hamming distance measurement


(b) Jaccard coefficient measurement


(c) SMC measurement

consider two features F_1 and F_2 having values (0,1, 1, 0, 1, 0, 1, 0) and (1, 1, 0, 0, 1, 0, 0, 0).

Jaccard coefficient of F_1 and F_2, $J =$

$$\frac{n_{11}}{n_{01} + n_{10} + n_{11}} = \frac{2}{1 + 2 + 2} = \frac{2}{5} \text{ or } 0.4.$$

$\therefore$ Jaccard distance between F_1 and F_2, $d_J = 1 - J = \frac{1}{2}$ or

0.6.

- **Simple matching coefficient (SMC)** is almost same as Jaccard coeficient except the fact that it includes a number of cases where both the features have a value of 0.

$$SMC = \frac{n_{11} + n_{00}}{n_{00} + n_{01} + n_{10} + n_{11}}$$

-
- where, n_{11} = number of cases where both the features have value 1
- n_{01} = number of cases where the feature 1 has value 0 and feature 2 has value 1
- n_{10} = number of cases where the feature 1 has value 1 and feature 2 has value 0
- n_{00} = number of cases where both the features have value 0
- Quite understandably, the total count of rows, $n = n_{00} + n_{01} + n_{10} + n_{11}$.

One more measure of similarity using similarity coefficient calculation is **Cosine Similarity**.

- For example in a typical text classification problem. The text corpus needs to be first transformed into features with a word token being a feature and the number of times the word occurs in a document comes as a value in each row. There are thousands of features in such a text data set.
- However, the data set is sparse in nature as only a few words do appear in a document, and hence in a row of the data set.

- So each row has very few non-zero values. However, the non-zero values can be anything integer value as the same word may occur any number of times.
- Also, considering the sparsity of the data set, the 0-0 matches (which obviously is going to be pretty high) need to be ignored.
- Cosine similarity which is one of the most popular measures in text classification is calculated as:

$$cos\,(x,\ y) = \frac{x.y}{\|x\|.\|y\|}$$

where, $x.y$ = vector dot product of x and $y = \sum_{i=1}^{n} x_i y_i$

$$\|x\| = \sqrt{\sum_{i=1}^{n} x_i^2} \text{ and } \|y\| = \sqrt{\sum_{i=1}^{n} y_i^2}$$

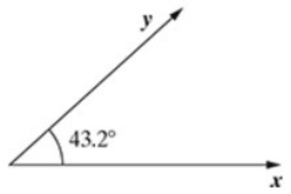
Let's calculate the cosine similarity of x and y, where $x = (2, 4, 0, 0, 2, 1, 3, 0, 0)$ and $y = (2, 1, 0, 0, 3, 2, 1, 0, 1)$.

In this case, $x.y = 2*2 + 4*1 + 0*0 + 0*0 + 2*3 + 1*2 + 3*1 + 0*0 + 0*1 = 19$

$$\|x\| = \sqrt{2^2 + 4^2 + 0^2 + 0^2 + 2^2 + 1^2 + 3^2 + 0^2 + 0^2} = \sqrt{34} = 5.83$$

$$\|y\| = \sqrt{2^2 + 1^2 + 0^2 + 0^2 + 3^2 + 2^2 + 1^2 + 0^2 + 1^2} = \sqrt{20} = 4.47$$

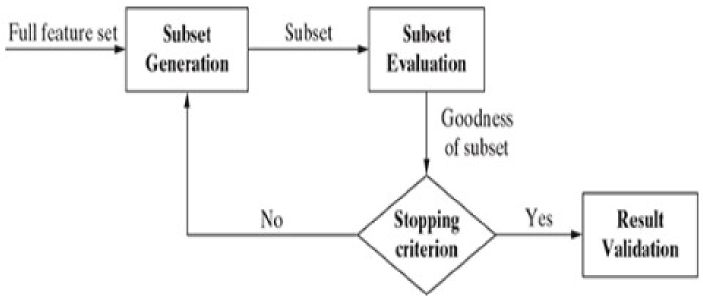
$$\therefore cos\,(x, y) = \frac{19}{5.83*4.47} = 0.729$$

- Cosine similarity actually measures the angle between x and y vectors.
- Hence, if cosine similarity has a value 1, the angle between x and y is 0° which means x and y are same except for the magnitude.
- If cosine similarity is 0, the angle between x and y is 90°.Hence, they do not share any similarity
- 

Overall feature selection process

- ➢ Feature selection is the process of selecting a subset of features in a data set.
- ➢ A typical feature selection process consists of four steps:
    1. Generation of possible subsets

2. Subset evaluation
3. Stop searching based on some stopping criterion
4. Validation of the result



Feature selection process

- **Subset generation**, which is the first step of any feature selection algorithm, is a search procedure which ideally should produce all possible candidate subsets.
- For an n-dimensional data set, 2 subsets can be generated. So, as the value of 'n' becomes high,
  finding an optimal subset from all the 2 candidate subsets becomes intractable.
- For that reason, different approximate search strategies are employed to find candidate subsets for evaluation.
- On one hand, the search may start with an empty set and keep adding features.This search strategy is termed as a sequential forward selection.
- On the other hand, a search may start with a full set and successively remove features. This strategy is termed as sequential backward elimination.
- In certain cases, search start with both ends and add and remove features simultaneously. This strategy is termed as a bi-directional selection.
- Each candidate subset is then evaluated and compared with the previous best performing subset based on certain evaluation criterion.
- If the new subset performs better, it replaces the previous one. This cycle of subset generation and evaluation continues till a pre-defined stopping criterion is fulfilled.

Some commonly used stopping criteria are

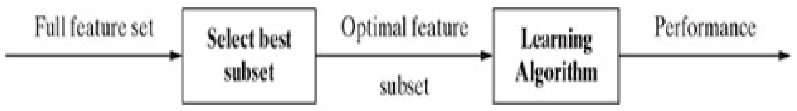
1. The search completes
2. some given bound (e.g. a specified number of iterations) is reached
3. subsequent addition (or deletion) of the feature is not producing a better subset
4. a sufficiently good subset (e.g. a subset having better classification accuracy than the existing benchmark) is selected

- Then the selected best subset is validated either against prior benchmarks or by experiments using real-life or synthetic but authentic data sets.
- In case of supervised learning, the accuracy of the learning model may be the performance parameter considered for validation.
- The accuracy of the model using the subset derived is compared against the model accuracy of the subset derived using some other benchmark algorithm.
  In case of unsupervised, the cluster quality may be the parameter for validation.

Feature selection approaches
There are four types of approach for feature selection:
1. Filter approach
2. Wrapper approach
3. Hybrid approach
4. Embedded approach

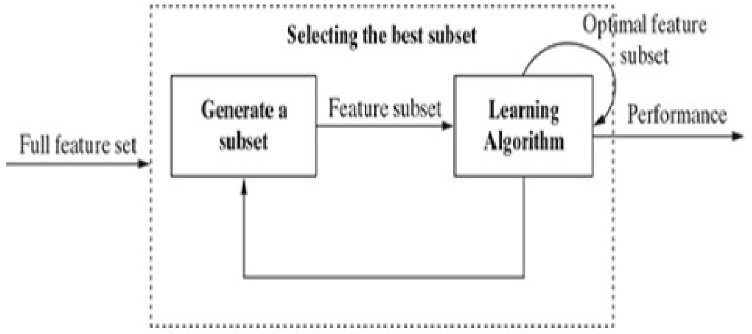
In the **filter approach** , the feature subset is selected based on statistical measures done to assess the merits of the features from the data perspective. No learning algorithm is employed to evaluate the goodness of the feature selected. Some of the common statistical tests conducted on features as a part of filter approach are –

Pearson's correlation,
Information gain,
Fisher score,
Analysis of variance(ANOVA),
Chi-Square, etc.



Filter approach

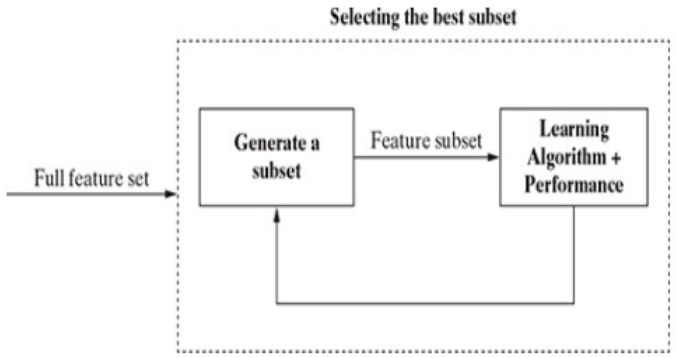
In the **wrapper approach**,identification of best feature subset is done using the induction algorithm as a black box. The feature selection algorithm searches for a good feature subset using the induction algorithm itself as a part of the evaluation function. Since for every candidate subset, the learning model is trained and the result is evaluated by running the learning lgorithm, wrapper approach is computationally very expensive. However, the performance is generally superior compared to filter approach.



Wrapper approach

Hybrid approach takes the advantage of both filter and wrapper approaches. A typical hybrid algorithm makes use of both the statistical tests as used in filter approach to decide the best subsets for a given cardinality and a learning algorithm to select the final best subset among the best subsets across different cardinalities.

Embedded approach is quite similar to wrapper approach as it also uses and inductive algorithm to evaluate the generated feature subsets. However, the difference is it performs feature selection and classification simultaneously.



Embedded approach