UNIT II Modelling and Evaluation &Basics of Feature Engineering
Introduction, selecting a Model, training a Model (for Supervised Learning), Model Representation and Interpretability, Evaluating Performance of a Model, Improving Performance of a Model Basics of Feature Engineering: Introduction, Feature Transformation, Feature Subset Selection

2.1 Introduction:

- 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
- Seeing the current trends in Information technology, a large volume of heterogeneous data is produced widely across the world by means of social media sites such as Facebook, Instagram, Google plus, etc.
- The generated data act as garbage and makes no sense until they are categorized.
- 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 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.
- 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
- **Generalization** is a term used to describe a model's ability to react to new data. That is, after being trained on a training set, a model can digest new data and make accurate predictions. A model's ability to generalize is central to the success of a model.
- If a model has been trained too well on training data, it will be unable to generalize. It will make inaccurate predictions when given new data, making the model useless even though it is able to make accurate predictions for the training data. This is called **overfitting.** The inverse is also true. **Underfitting** happens when a model has not been trained enough on the data. In the case of underfitting, it makes the model just as useless and it is not capable of making accurate predictions, even with the training data.
- If the outcome is systematically incorrect, the learning is said to have a **bias**.

2.2 SELECTING A MODEL(5 Marks)

- There is no one model that works best for every machine learning problem.
- Any learning model tries to simulate some real-world aspect, firstly 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**.

2.2.1 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, in turn, 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 and Logistic Regression models are popular regression models.

2.2.2 Descriptive models:

- Models for unsupervised learning or descriptive models are used to describe a data set. There is no target feature 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 vice versa.

2.3 TRAINING A MODEL (FOR SUPERVISED LEARNING)

2.3.1 Holdout method:(5 Marks)

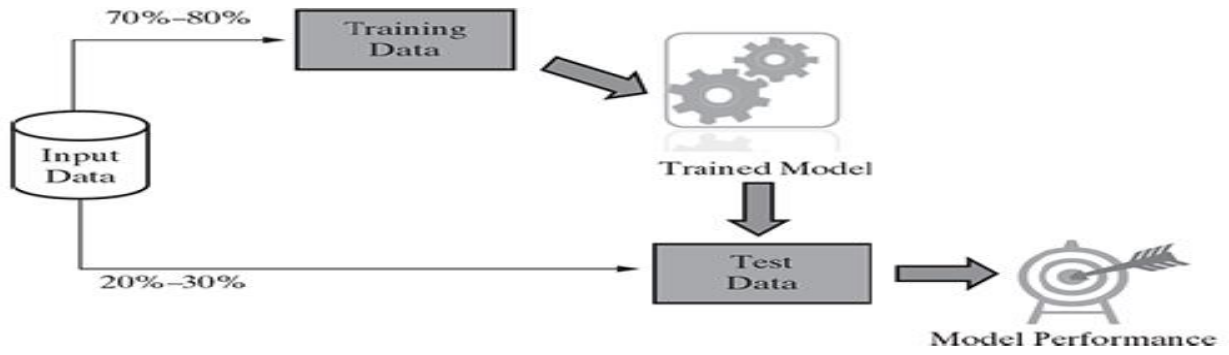


FIG. 3.1 Holdout method

- 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 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 function. Then the predicted value is compared with the actual value of the label. The performance of the model is generally measured by the accuracy of prediction of the label value.
- In certain cases, the input data is partitioned into three portions –
  - **training** data
  - test data
  - 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

- **A 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.
- **A Solution** Stratification is a technique, using which data items belonging to two different classes are placed equally in training and testing data.

2.3.2 K-fold Cross-validation method(5 Marks):

- 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
- A special variant of holdout method, called repeated holdout is used to overcome challenge
- In repeated holdout, several random holdouts are used to measure the model performance. In the end, the average of all performances is taken.
- 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:
    - 10-fold cross-validation (10-fold CV)
    - Leave-one-out cross-validation (LOOCV)

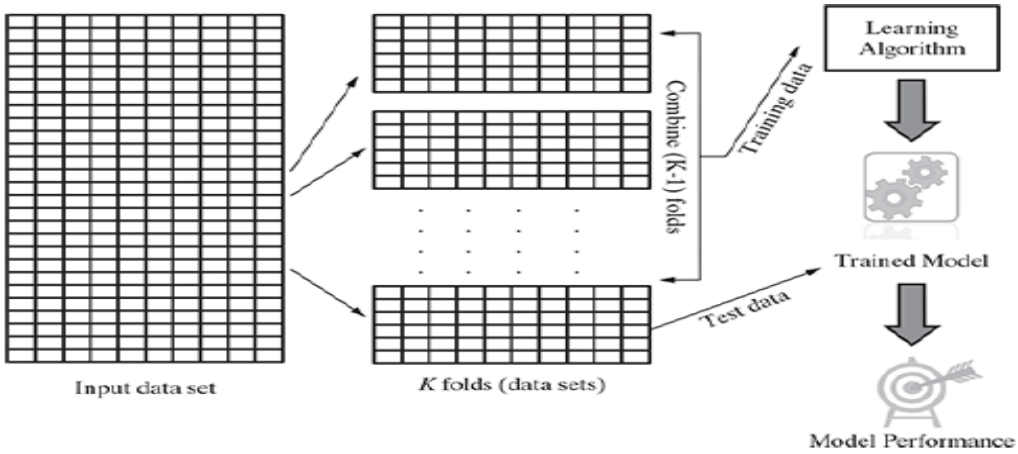


FIG. 3.2 Overall approach for *K*-fold cross-validation

10-fold cross-validation (10-fold CV):

- 10-fold cross-validation is by far the most popular approach. In this approach, for each of the 10-folds, each including of approximately 10% of the data, one of the folds is used as the test data for validating model and remaining 9 folds is used for training.
- We repeat this procedure 10 times each time reserving a different tenth for testing and the remaining folds as the training data.
- The average performance across all folds is being reported

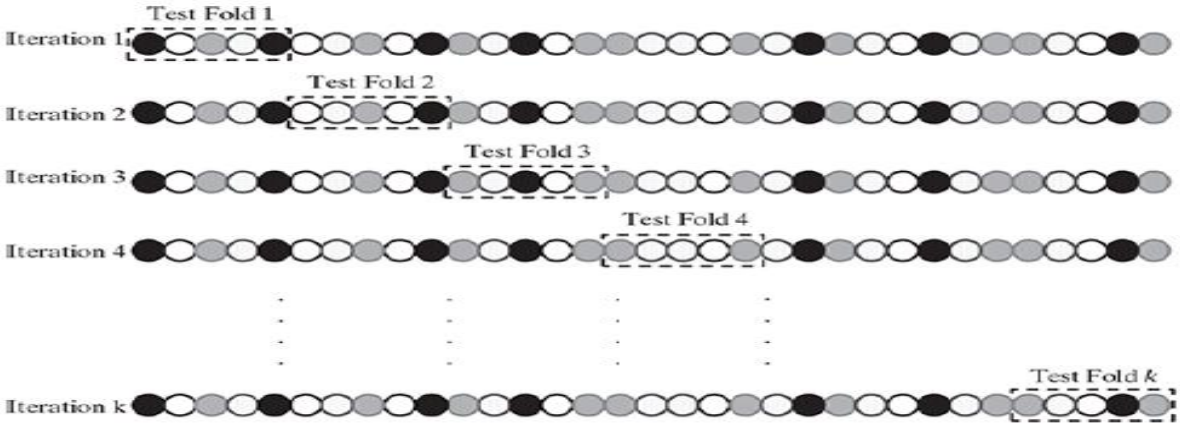


FIG. 3.3 Detailed approach for fold selection

- As can be observed in the figure, 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.

Leave-one-out cross-validation (LOOCV):

- 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.

2.3.3 Bootstrap sampling(5 Marks):

- 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.
- 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**
- **This technique is particularly useful in case of input data sets of small size, i.e. having very less number of data instances.**

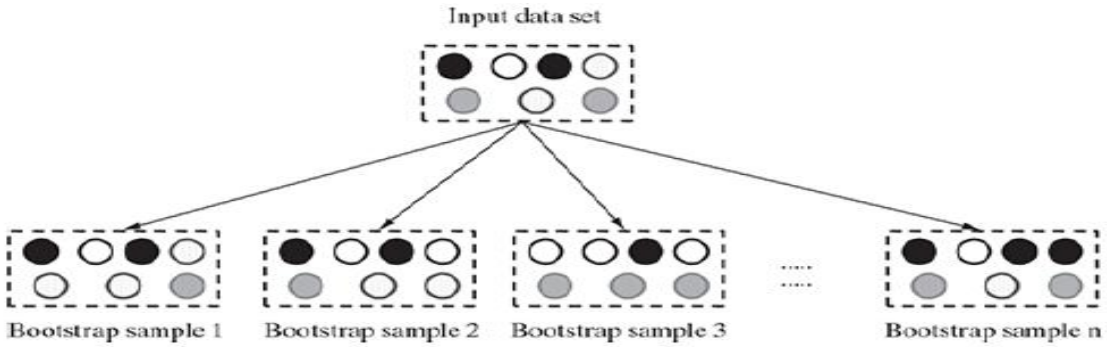


FIG. 3.4 Bootstrap sampling

What is the difference between bootstrapping and cross-validation? (2 or 5 Marks)

Bootstrapping	Cross-Validation
It is a special variant of holdout method, called repeated holdout. Hence use stratified random sampling approach(without replacement). Data set divides into k folds	It uses the technique of **Simple Random Sampling with Replacement (SRSWR).** So same data instance can be picked for multiple times
The number of possible training/test data samples is finite	The number of possible training/test data samples is unlimited
Bootstrapping selects samples with replacements that can be as big as the dataset.	Cross-validation samples are smaller than the dataset.
Bootstrapping contains repeated elements in every subset. Bootstrapping relies on random sampling.	Cross-validation does not rely on random sampling, just splitting the dataset into k unique subsets.

2.3.4 Lazy vs. Eager learner:(2 Marks)

- **Eager learning** follows the general principles of machine learning. 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. In that respect, lazy learner doesn't 'learn' anything. It uses the training data in exact. **Since lazy learning uses training data as-is, it is also known as rote learning**. Due to its heavy dependency

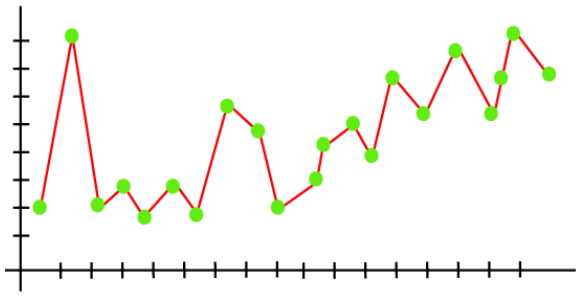
on the given training data instance, it is also known as instance learning. They are also called non-parametric learning. **Lazy learners take very little time in training** because not much of training actually happens. However, it takes quite some time in classification as for each tuple of test data, a comparison-based assignment of label happens. **One of the most popular algorithm for lazy learning is k-nearest neighbor.**

2.4 MODEL REPRESENTATION AND INTERPRETABILITY:

- The main goal of each machine learning model is to generalize well. Here generalization defines the ability of an ML model to provide a suitable output by adapting the given set of unknown input.
- It means after providing training on the dataset, it can produce reliable and accurate output. Hence, the underfitting and overfitting are the two terms that need to be checked for the performance of the model and whether the model is generalizing well or not.
- Before diving further let's understand two important terms:
    - **Bias:** It is actually the error rate of the training data. When the error rate has a high value, we call it High Bias and when the error rate has a low value, we call it low Bias.
    - **Variance:** The difference between the error rate of training data and testing data is called variance. If the difference is high then it's called high variance and when the difference of errors is low then it's called low variance. Usually, we want to make a low variance for generalized our model.

2.4.1 Overfitting:(2 or 5 Marks)

- Overfitting occurs when our machine learning model tries to cover more than the required data present in the given dataset.
- Because of this, the model starts caching noise and inaccurate values present in the dataset, and all these factors reduce the efficiency and accuracy of the model.
- The overfitted model has **low bias** and **high variance.**
- The chances of occurrence of overfitting increase as much we provide training to our model. It means the more we train our model, the more chances of occurring the overfitted model. Overfitting is the main problem that occurs in supervised learning.
- **Example:** The concept of the overfitting can be understood by the below graph of the linear regression output:



How to avoid the Overfitting in Model

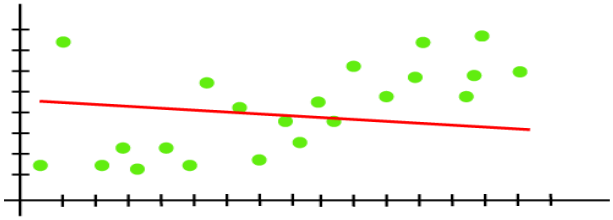
Both overfitting and underfitting cause the degraded performance of the machine learning model. But the main cause is overfitting, so there are some ways by which we can reduce the occurrence of overfitting in our model.

- Cross-Validation
- Training with more data
- Removing features
- Early stopping the training
- Regularization
- Ensembling

2.4.2 Underfitting: (2 or 5 Marks)

- Underfitting occurs when our machine learning model is not able to capture the underlying trend of the data. To avoid the overfitting in the model, the fed of training data can be stopped at an early stage, due to which the model may not learn enough from the training data. As a result, it may fail to find the best fit of the dominant trend in the data.

- In the case of underfitting, the model is not able to learn enough from the training data, and hence it reduces the accuracy and produces unreliable predictions.
- An underfitted model has **high bias and low variance.**
- **Example:** We can understand the underfitting using below output of the linear regression model:



- As we can see from the above diagram, the model is unable to capture the data points present in the plot.

How to avoid underfitting:

- By increasing the training time of the model.
- By increasing the number of features.

2.4.3 Bias – variance trade-off: (2 or 5 Marks)

- 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'
  - error due to 'variance'

2.4.3.1 Errors due to 'Bias'

- Errors due to bias arise 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.

2.4.3.2 Errors due to 'Variance'

- Errors due to variance occur from difference in training data sets used to train the model. 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 increases even small differences in the training data.
- As is quite understandable:
  - Increasing the bias will decrease the variance, and
  - Increasing the variance will decrease the bias

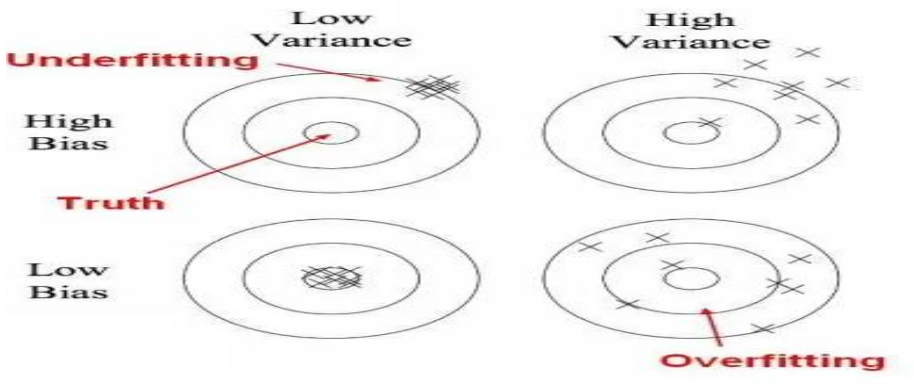


FIG. 3.6 Bias-variance trade-off

- In the above diagram, center of the target is a model that perfectly predicts correct values. As we move away from the bulls-eye our predictions become get worse and worse.

Why is Bias Variance Tradeoff?

- If our model is too simple and has very few parameters then it may have high bias and low variance. On the other hand if our model has large number of parameters then it's going to have high variance and low bias. So we need to find the right/good balance without 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.
- To build a good model, we need to find a good balance between bias and variance such that it minimizes the total error.

Total Error = Bias^2 + Variance + Irreducible Error

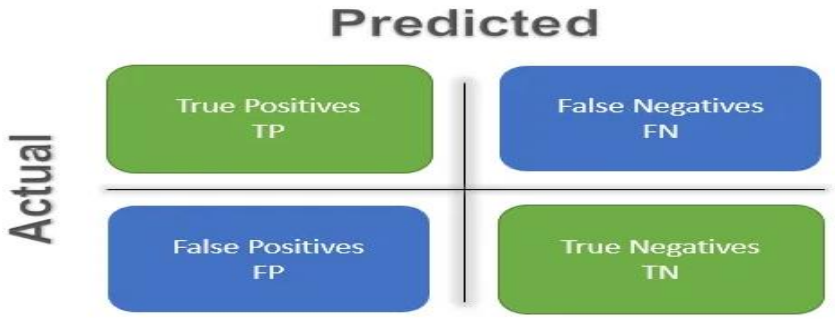
2.5 EVALUATING PERFORMANCE OF A MODEL:

2.5.1 Supervised learning – classification:(10 Marks)

- In machine learning, a performance evaluation metric plays a very important role in determining the performance of our machine learning model on a dataset
- There are many performance evaluation metrics that you can use to measure the performance of your machine learning models for classification
- Some common terms to be clear with are:
  - True positives (TP): Predicted positive and are actually positive.
  - False positives (FP): Predicted positive and are actually negative.
  - True negatives (TN): Predicted negative and are actually negative.
  - False negatives (FN): Predicted negative and are actually positive.

Confusion Matrix:

- 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

Accuracy:(2 Marks)

- For any classification model, **model accuracy** is given by total number of correct classifications (either True Positive or True Negative) divided by total number of classifications done.

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

- 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.

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

Error rate: (2 Marks)

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

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

- In context of the above confusion matrix,

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

$$= 1 - \text{Model accuracy}$$

Kappa Value: (2 Marks)

- **Kappa** value of a model indicates the adjusted the model accuracy.
- kappa statistic is a very good measure that can handle very well both multi-class and imbalanced class problems.
- 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.

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

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

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

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

Note: Kappa value can be 1 at the maximum, which represents perfect agreement between model's prediction and actual values

Sensitivity: (2 Marks)

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

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

- In the context of the above confusion matrix for the cricket match win prediction problem,

$$\text{Sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{85}{85 + 2} = \frac{85}{87} = 97.7\%$$

Specificity: (2 Marks)

- Specificity of a model measures the proportion of negative examples which have been correctly classified.
- A higher value of specificity will indicate a better model performance.
- In the context of the above confusion matrix for the cricket match win prediction problem,

$$\text{Specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}} = \frac{9}{9 + 4} = \frac{9}{13} = 69.2\%$$

Precision: (2 Marks)

- Precision is defined as the ratio of correctly classified positive samples (True Positive) to a total number of classified positive samples (either correctly or incorrectly).

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

- Precision helps us to visualize the reliability of the machine learning model in classifying the model as positive.
- In context of the above confusion matrix for the cricket match win prediction problem,

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{85}{85 + 4} = \frac{85}{89} = 95.5\%$$

Recall: (2 Marks)

- Recall indicates the proportion of correct prediction of positives to the total number of positives.
- The recall measures the model's ability to detect positive samples. The higher the recall, the more positive samples detected.

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

- In the context of the above confusion matrix for the cricket match win prediction problem,

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{85}{85 + 2} = \frac{85}{87} = 97.7\%$$

F-measure: (2 Marks)

- 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

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

- In context of the above confusion matrix for the cricket match win prediction problem,

$$F\text{-measure} = \frac{2 \times 0.955 \times 0.977}{0.955 + 0.977} = \frac{1.866}{1.932} = 96.6\%$$

- As a combination of multiple measures into one, Fscore gives the right measure using which performance of different models can be compared

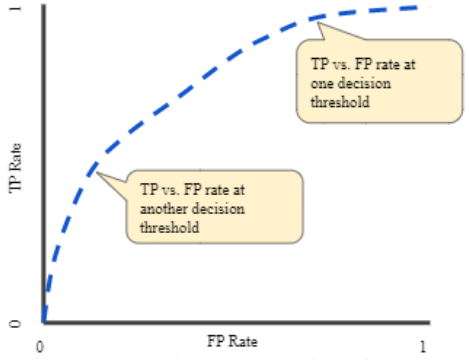
Receiver operating characteristic (ROC) curves: (2 Marks)

- 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.
- It is a probability curve that plots two parameters, the True Positive Rate (TPR) against the False Positive Rate (FPR)
- In the curve, TPR is plotted on Y-axis, whereas FPR is on the X-axis at different classification thresholds

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

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

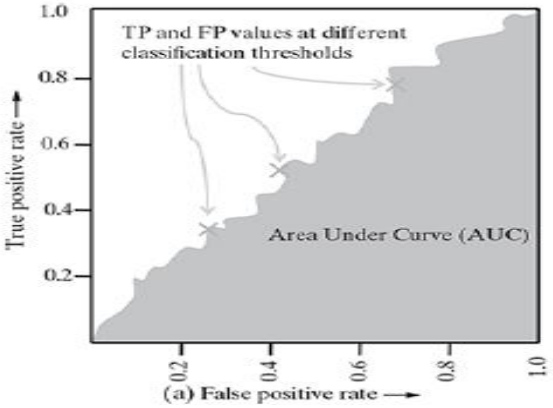
- An ROC curve plots TPR vs. FPR at different classification thresholds. Lowering the classification threshold classifies more items as positive, thus increasing both False Positives and True Positives.
- The following figure shows a typical ROC curve.



TP vs. FP rate at different classification thresholds.

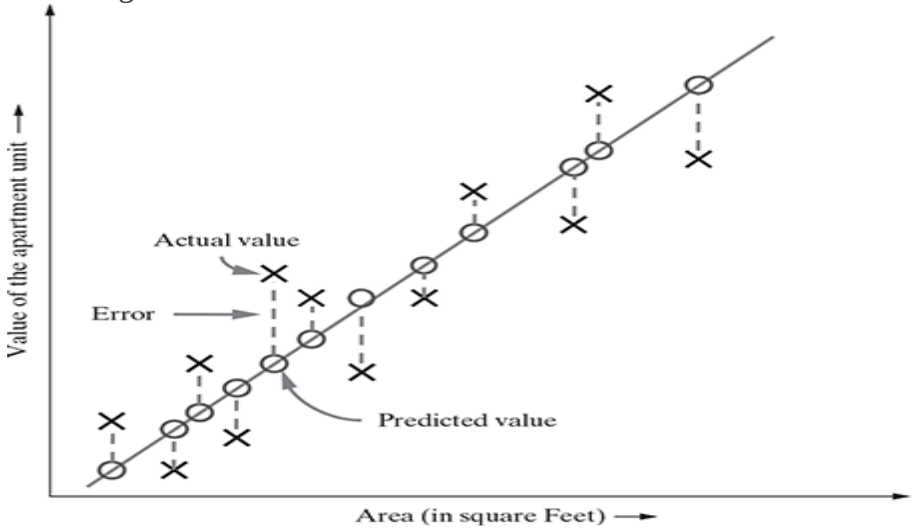
AUC: Area Under the ROC Curve:

- AUC stands for "Area under the ROC Curve." That is, AUC measures the entire two-dimensional area underneath the entire ROC curve (think integral calculus) 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.
  - 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



2.5.2 Supervised learning – regression(5 Marks)

- 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.



Error – Predicted vs. actual value

- The Above graph 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:
- For a certain value of x, say $\hat{x}$, the value of y is predicted as $\hat{y}$ whereas the actual value of y is Y (say). The distance between the actual value and the predicted value, i.e. $\hat{y}$ 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 $\bar{y}$ 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.

3.5.3 Unsupervised learning - clustering(5 Marks)

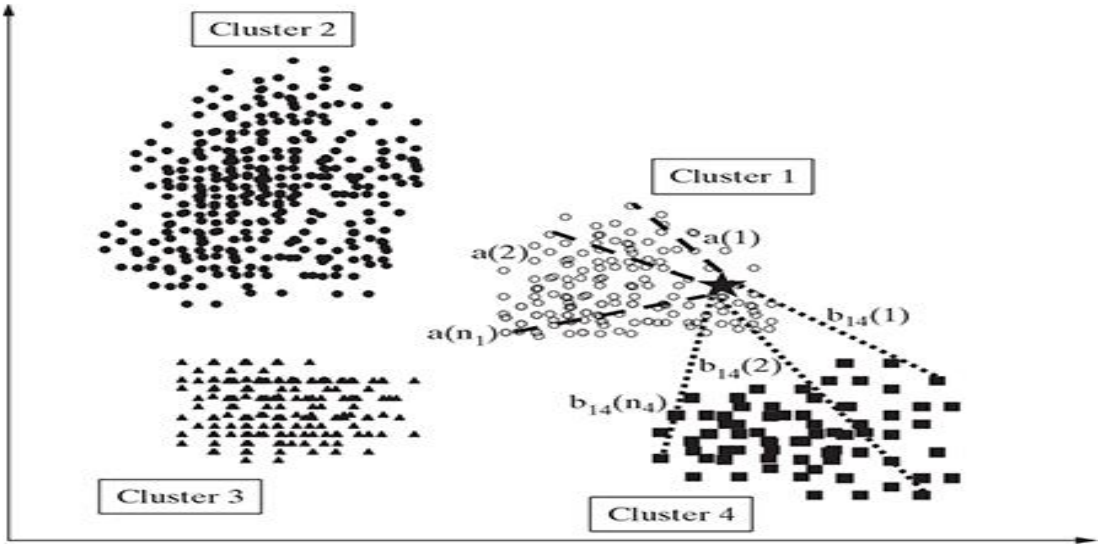
However, there are couple of popular approaches which are adopted for cluster quality evaluation.

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



Silhouette width calculation

- Let's try to understand this in context of the above example. There are four clusters namely cluster 1, 2, 3, and 4. Let'sconsider an arbitrary data element 'i' in cluster 1, resembled by the asterisk. a(i) is the average of the distances a , a , …, a of the different data elements from the i th data element in cluster 1, assuming there are n data elements in cluster 1. Mathematically,

$$a(i) = \frac{a_{i1} + a_{i2} + \ldots + a_{in_1}}{n_1}$$

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:

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

2.6 IMPROVING PERFORMANCE OF A MODEL(5 Marks)

- Machine learning development would be not difficult for ML engineers, but ensuring its performance is important to get accurate and most reliable results. Though, there are various methods you can improve your machine learning model performance.
- **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.

Ensemble:

- 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 methods combine weaker learners to create stronger ones. A performance boost can be expected even if models are built as usual and then ensembled.

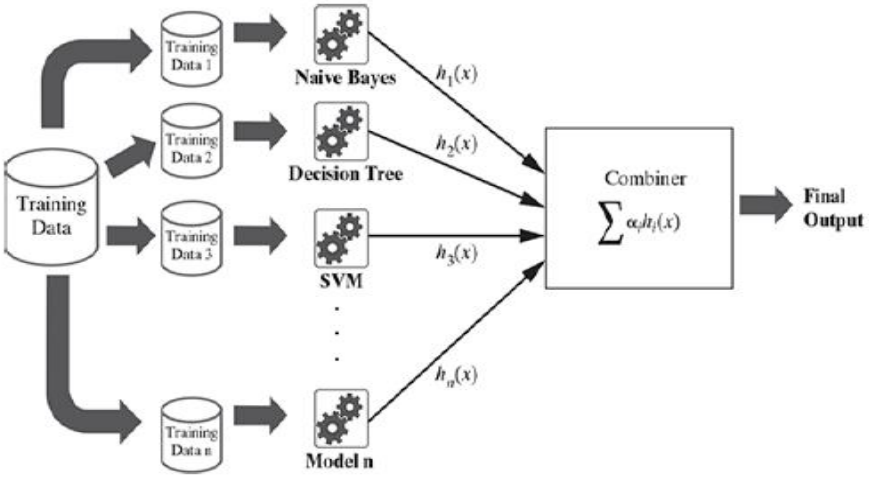


FIG. 3.11 Ensemble

- Following are the typical steps in ensemble process:
  - Build a number of models based on the training data 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.

- Alternatively, the same training data may be used but the models combined are quite varying, e.g, SVM, neural network, *k*NN, etc.
- 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.
- 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.

Basics of Feature Engineering

2.7 What is Feature: (2 Marks)

- A feature is an attribute of a data set that is used in a machine learning process. 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.
- A model for predicting the risk of cardiac disease may have features such as the following: Age, Gender, Weight, Whether the person smokes, etc.
- Let's take the example of a famous machine learning data set, Iris. It has five attributes or features namely Sepal.Length, Sepal.Width, Petal.Length, Petal. Width and Species. Out of these, the feature 'Species' represent the class variable and the remaining features are the predictor variables. It is a five-dimensional data set

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
6.7	3.3	5.7	2.5	Virginica
4.9	3	1.4	0.2	Setosa
5.5	2.6	4.4	1.2	Versicolor
6.8	3.2	5.9	2.3	Virginica
5.5	2.5	4	1.3	Versicolor
5.1	3.5	1.4	0.2	Setosa
6.1	3	4.6	1.4	versicolor

Data set features

2.7.1 Why Feature is Important in Machine Learning? (2 Marks)

- Features in machine learning is very important, being building a block of datasets, the quality of the features in your dataset has major impact on the quality of the insights you will get while using the dataset for machine learning.

2.7.2 What is 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

1) **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.

2) **Feature subset 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. So, essentially the job of feature selection is to derive a subset F_j (F_1 , F_2 , …, F_m) of F_i (F_1 , F_2 , …, F_n), where $m < n$, such that F_j is most meaningful and gets the best result for a machine learning problem.

2.8 FEATURE TRANSFORMATION:

- Feature transformation is a mathematical transformation in which we apply a mathematical formula to a particular column (feature) and transform the values, which are useful for our further analysis. It is a technique by which we can boost our model performance.
- It is also known as Feature Engineering, which creates new features from existing features that may help improve the model performance.
- It refers to the algorithm family that creates new features using the existing features. These new features may not have the same interpretation as the original features, but they may have more explanatory power in a different space rather than in the original space.

2.8.1 Feature construction: (5 Marks)

- Feature construction involves transforming a given set of input features to generate a new set of more powerful features.
- let's take the example of a real estate data set having three features – apartment length, apartment breadth, and price of the apartment
- 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.
- In other words, we transform the three-dimensional data set to a four-dimensional data set, with the newly 'discovered' feature apartment area being added to the original data set

apartment_ length	apartment_ breadth	apartment_ price
80	59	23,60,000
54	45	12,15,000
78	56	21,84,000
63	63	19,84,000
83	74	30,71,000
92	86	39,56,000

→

apartment_ length	apartment_ breadth	apartment_ area	apartment_ price
80	59	4,720	23,60,000
54	45	2,430	12,15,000
78	56	4,368	21,84,000
63	63	3,969	19,84,500
83	74	6,142	30,71,000
92	86	7,912	39,56,000

Feature construction

- There are certain situations where feature construction is an essential activity before we can start with the machine learning task. 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:

- Let's take the example of data set on athletes, data set has features age, city of origin, parents athlete and Chance of Win.
- The feature chance of a win is a class variable while the others are predictor variables
- Any machine learning algorithm, whether it's a classification algorithm (like kNN) or a regression algorithm, requires numerical figures to learn from. So 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.

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

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

Feature construction (encoding nominal variables)

Encoding categorical (ordinal) variables:

- Let's take an example of a student data set and it has three features science marks, maths marks and grade.
- As we can see, the grade is an ordinal variable with values A, B, C, and D. To transform this variable to a numeric variable, we can create a feature num_grade 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

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

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

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 we think about social networks like Facebook, Twitter, emails, Whatsapp, text plays a major role in the flow of information.

- Hence, text mining is an important area of research. However, making sense of text data, due to the inherent unstructured nature of the data, is not so straightforward.
- All machine learning models need numerical data as input. So the text data in the data sets need to be transformed into numerical features.
- The process of converting a text data into a numerical representation is known as vectorization. In this process, word occurrences in all documents belonging to the text data or 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 or Text Data, 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

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)

2.8.2 Feature extraction(5 Marks)

- 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.
- Let's take an example. Say, we have a data set with a feature set F (F_1 , F_2 , …, Fn). After feature extraction using a mapping function f
- we will have a set of features F' (F'_1 , F'_2 , …, F'm) such that $F'_i = f(F_i)$ and m<n
- For example $F'_1 = k_1F_1 + k_2F_2$

$Feat_A$	$Feat_B$	$Feat_C$	$Feat_D$		$Feat_1$	$Feat_2$
34	34.5	23	233		41.25	185.80
44	45.56	11	3.44		54.20	53.12
78	22.59	21	4.5	→	43.73	35.79
22	65.22	11	322.3		65.30	264.10
22	33.8	355	45.2		37.02	238.42
11	122.32	63	23.2		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 are
    - Principal Component Analysis
    - Singular value decomposition
    - Linear Discriminant Analysis

Principal Component Analysis:

- Every data set, has multiple attributes or dimensions – many of which might have similarity with each other.
- 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.
- **In PCA, a new set of features are extracted from the original features which are quite dissimilar in nature. So an n-dimensional feature space gets transformed to an m-dimensional feature space, where the dimensions are orthogonal to each other, i.e. completely independent of each other**.
- 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 Σ is an m × n rectangular diagonal matrix. The diagonal entries of Σ 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 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

$$D = D \times [v_1, v_2, \ldots, v_k]$$

- 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 one of the commonly used feature extraction techniques in machine learning to solve more than two-class classification problems. It is also known as Normal Discriminant Analysis (NDA) or Discriminant Function Analysis (DFA).
- It is a technique used to find a linear combination of features that best separates the classes in a dataset.

- 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
- LDA calculates eigenvalues and eigenvectors within a class and interclass scatter matrices. Below are the steps to be followed:
  - Calculate the mean vectors for the individual classes.
  - Calculate intra-class and inter-class scatter matrices.
  - Calculate eigenvalues and eigenvectors for *S* and *S*, where *S* is the intra-class scatter matrix and *S* 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* 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
  - Identify the top '*k*' eigenvectors having top '*k*' eigenvalues

2.9 FEATURE SUBSET SELECTION: (5 Marks)

- Feature selection is the most critical pre-processing activity in any machine learning. It aims to select a subset of system attributes or features which makes a most meaningful contribution in a machine learning activity
- For example, we are trying 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**. We can well understand that **Roll Number** can have no use in weight prediction. So we can eliminate the feature roll number and build a feature subset to be considered in this machine learning problem.

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

FIG. 4.8 Feature selection

2.9.1 Issues in high-dimensional data:

- High-dimensional' refers to the high number of attributes or features present in certain data sets, in the domains like DNA analysis, geographic information systems (GIS), social networking, etc.
- 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:
  - Having faster and more cost-effective learning model
  - Improving the efficiency of the learning model
  - Having a better understanding of the underlying model that generated the data

2.9.2 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 or feature, is expected to contribute information to decide the value of the class label. In case a feature 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 features. Certain features do not contribute any useful information for deciding the similarity of dissimilarity of data instances. Hence, these features are marked as irrelevant variables in the context of the unsupervised machine learning task.
- Any feature which is irrelevant in the context of a machine learning task is eliminated when we are selecting a subset of features. We can consider whether the weakly relevant features are to be eliminated or not 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. The feature is said to be potentially redundant in the context of the learning problem.
- For example, in the weight prediction problem. The student weight data set has features such as **Roll Number, Age, Height, and Weight**.
- 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
- All features having potential redundancy are eliminated from the final feature subset.
- The main objective of feature selection is to remove all features which are irrelevant and redundant. This leads to a meaningful feature subset in context of a specific learning task.

2.9.3 Measures of feature relevance and redundancy:

<u>Measures of feature relevance:</u>

- Feature relevance is to be measured 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. That's why it is a good indicator of the relevance of a feature with respect to the class variable. **Higher the value of mutual information** of a feature, more relevant is that feature. Mutual information can be calculated as follows:

$$MI(C, f) = H(C) + H(f) - H(C, f) \text{ where,}$$

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. 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: (5 Marks)

- **Feature redundancy:** A feature may contribute information which is similar to the information contributed by one or more other features.
- There are multiple measures of similarity of information contribution, prominent ones are
    1. Correlation-based measures
    2. Distance-based measures, and
    3. Other coefficient-based measure

1. Correlation-based similarity measures:

- 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.

2. 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. Data set has two features, aptitude (F_1) and communication (F_2) under consideration. The Euclidean distance between the features has been calculated using the formula provided above.

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

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

    - Minkowski distance takes the form of Euclidean distance (also called L norm) when r = 2.
- At r = 1, it takes the form of Manhattan distance (also called L norm), as shown below:

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

3. Other similarity measures:

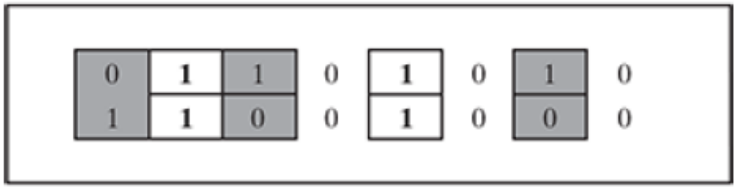
- Some of similarity measures are as follows
  - Jaccard index/coefficient Measurement
  - Simple matching coefficient (SMC)
  - Cosine Similarity

Jaccard index/coefficient Measurement:

- **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, $d_J = 1 - J$



Jaccard coefficient measurement

- Let's consider two features F and F having values (0,1, 1, 0, 1, 0, 1, 0) and (1, 1, 0, 0, 1, 0, 0, 0). The above fig shows the identification of the values of n_{11} , n_{01} and n_{10} . As shown, the cases where both the values are 0 have been left out without border – as an indication of the fact that they will be excluded in the calculation of Jaccard coefficient.

$$\text{Jaccard coefficient of } F_1 \text{ and } F_2, J = \frac{n_{11}}{n_{01} + n_{10} + n_{11}} = \frac{2}{1 + 2 + 2} = \frac{2}{5} \text{ or } 0.4.$$

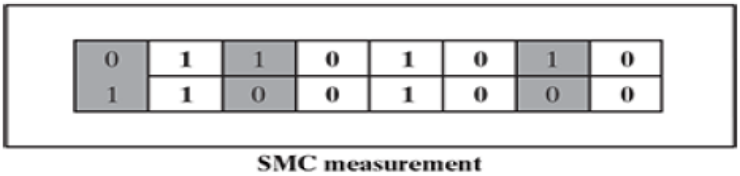
$$\text{Jaccard distance between } F_1 \text{ and } F_2, d_J = 1 - J = \frac{1}{2} \text{ or } 0.6.$$

Simple matching coefficient (SMC):

- **Simple matching coefficient (SMC)** is almost same as Jaccard coefficient 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



SMC measurement

- Quite understandably, the total count of rows, $n = n_{11} + n_{01} + n_{10} + n_{00}$. As shown in above Figure, all values have been included in the calculation of SMC.

$$\therefore \text{SMC of } F_1 \text{ and } F_2 = \frac{n_{11} + n_{00}}{n_{00} + n_{01} + n_{10} + n_{11}} = \frac{2 + 3}{3 + 1 + 2 + 2} = \frac{1}{2} \text{ or } 0.5.$$

Cosine Similarity:

- One more measure of similarity using similarity coefficient calculation is Cosine Similarity.
- Let's take the example of a typical text classification problem. The text data 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:

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

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

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

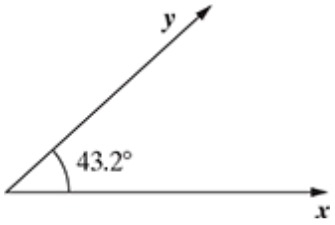
- Let's calculate the cosine similarity of x and y, where x = (2, 4, 0, 0, 2, 1, 3, 0, 0) and y = (2, 1, 0, 0, 3, 2, 1, 0, 1).
- In this case, x.y = 2*2 + 4*1 + 0*0 + 0*0 + 2*3 + 1*2 +3*1 + 0*0 + 0*1 = 19

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

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

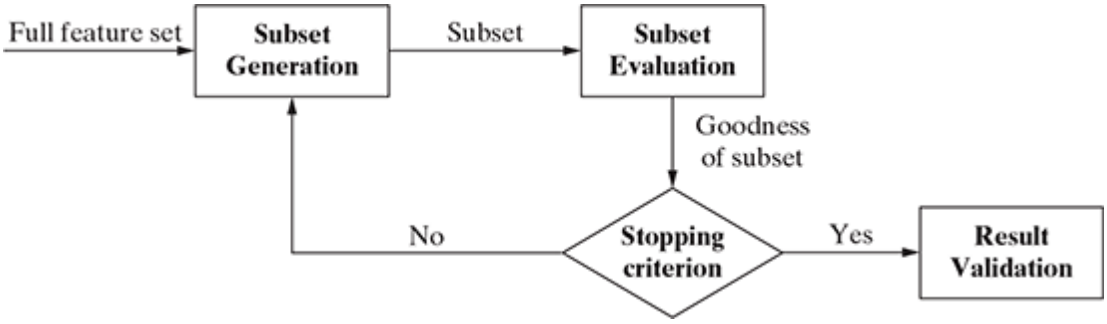
$$\therefore cos\,(x, y) = \frac{19}{5.83*4.47} = 0.729$$

- 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.
- In the above example, the angle comes to be 43.2°.



2.9.4 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



- **Subset generation**, which is the first step of any feature selection algorithm, is a search procedure which ideally should produce all possible candidate subsets.
- However, for an n-dimensional data set, 2^n subsets can be generated. So, as the value of '*n*' becomes high, finding an optimal subset from all the 2^n candidate subsets becomes difficult. 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.

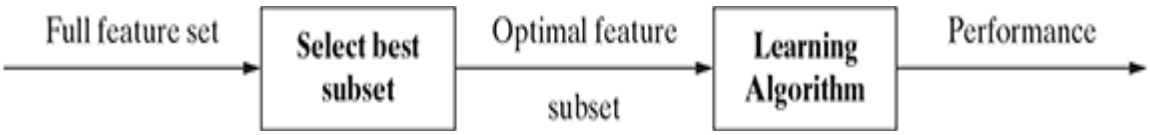
- On the other hand, a search may start with a full set and successively remove features. This strategy is termed as sequential backward elimination. In certain cases, search start with both ends and add and remove features simultaneously. This strategy is termed as a bi-directional selection.
- Each 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:
    - The search completes
    - Some given bound is reached
    - Subsequent addition (or deletion) of the feature is not producing a better subset
    - A sufficiently good subset is selected
- Then the selected best subset is **validated**. 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.

2.9.5 Feature selection approaches: (5 Marks)

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

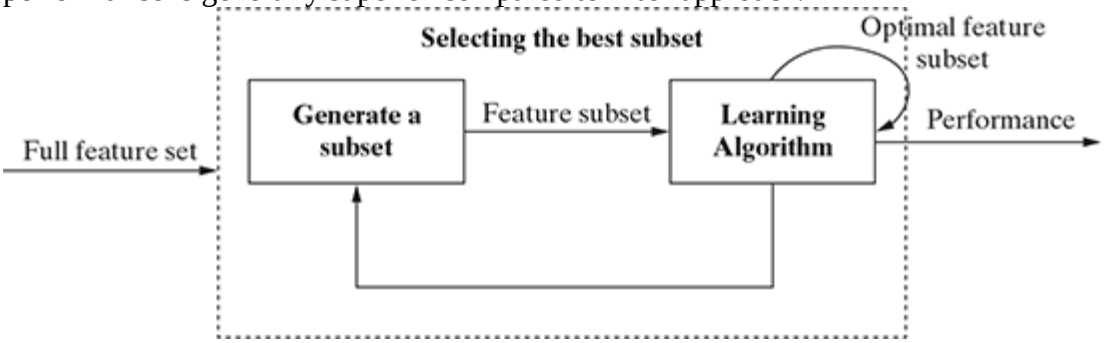
1. Filter approach:

- 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.



2. Wrapper 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 algorithm, wrapper approach is computationally very expensive. However, the performance is generally superior compared to filter approach.



3. Hybrid 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.

4. Embedded approach:

- 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.