Module 2

Understanding Data

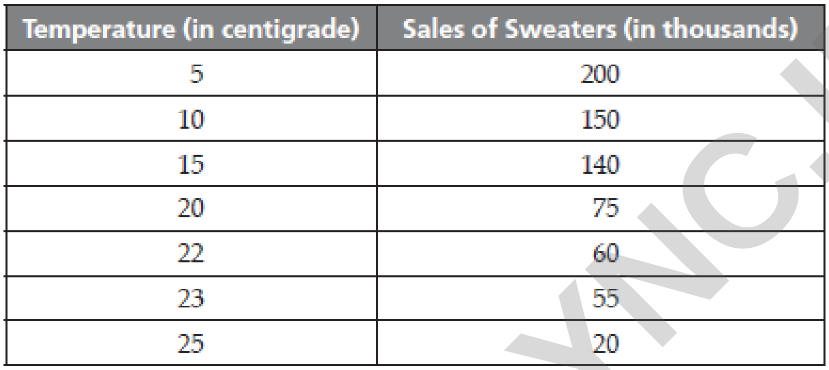
Bivariate and Multivariate data, Multivariate statistics**,** Essential mathematics for Multivariate data, Overview hypothesis, Feature engineering and dimensionality reduction techniques, **Basics of Learning Theory:** Introduction to learning and its types**,** Introduction computation learning theory, Design of learning system, Introduction concept learning. **Similarity-based learning**: Introduction to Similarity or instance based learning, Nearest-neighbour learning**,** weighted **k**- Nearest **-** Neighbour **algorithm.**

CHAPTER -2

2.6 BIVARIATE DATA AND MULTIVARIATE DATA

Bivariate Data involves two variables. Bivariate data deals with causes of relationships. The aim is to find relationships among data. Consider the following Table 2.3, with data of the temperature in a shop and sales of sweaters.

Table 2.3: Temperature in a Shop and Sales Data



Temperature (in centigrade)	Sales of Sweaters (in thousands)
5	200
10	150
15	140
20	75
22	60
23	55
25	20

VTUSYNC.IN

Here, the aim of bivariate analysis is to find relationships among variables. The relationships can then be used in comparisons, finding causes, and in further explorations. To do that, graphical display of the data is necessary. One such graph method is called scatter plot.

Scatter plot is used to visualize bivariate data. It is useful to plot two variables with or without nominal variables, to illustrate the trends, and also to show differences. It is a plot between explanatory and response variables. It is a 2D graph showing the relationship between two variables. Line graphs are similar to scatter plots. The Line Chart for sales data is shown in Figure 2.12.

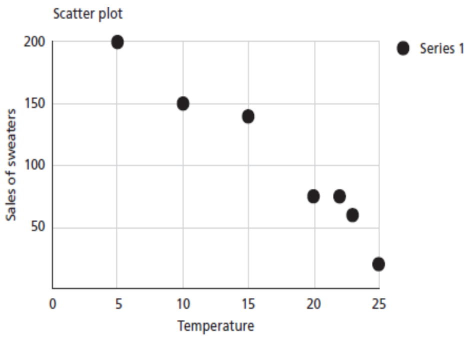


Figure 2.11: Scatter Plot

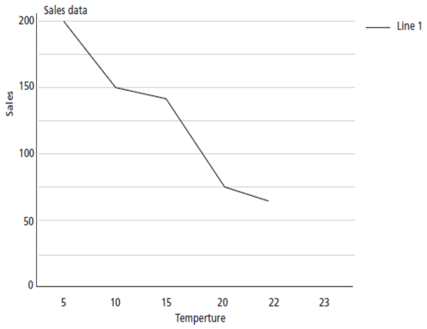


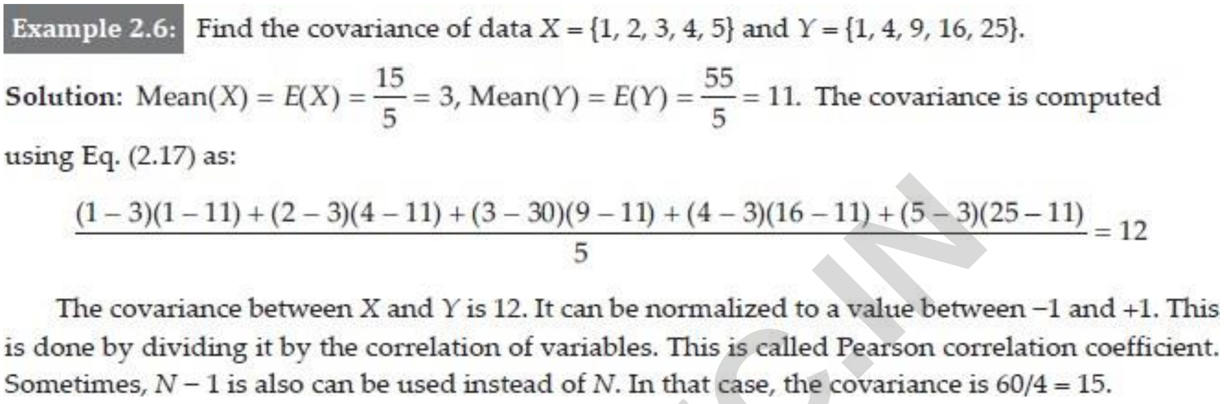
Figure 2.12: Line Chart

2.6.1 Bivariate Statistics

Covariance and Correlation are examples of bivariate statistics. Covariance is a measure of joint probability of random variables, say X and Y. Generally, random variables are represented in capital letters. It is defined as covariance (X, Y) or COV (X, Y) and is used to measure the variance between two dimensions. The formula for finding co-variance for specific x, and y are:

$$\text{cov}(X,Y) = \frac{1}{N}\sum_{i=1}^{N}(x_i - E(X))(y_i - E(Y))$$

Here, xi and yi are data values from X and Y. E(X) and E(Y) are the mean values of xi and yi. N is the number of given data. Also, the COV(X, Y) is same as COV(Y, X).



Example 2.6: Find the covariance of data $X = \{1, 2, 3, 4, 5\}$ and $Y = \{1, 4, 9, 16, 25\}$.

Solution: $\text{Mean}(X) = E(X) = \frac{15}{5} = 3$, $\text{Mean}(Y) = E(Y) = \frac{55}{5} = 11$. The covariance is computed using Eq. (2.17) as:

$$\frac{(1-3)(1-11)+(2-3)(4-11)+(3-30)(9-11)+(4-3)(16-11)+(5-3)(25-11)}{5} = 12$$

The covariance between X and Y is 12. It can be normalized to a value between −1 and +1. This is done by dividing it by the correlation of variables. This is called Pearson correlation coefficient. Sometimes, $N - 1$ is also can be used instead of N. In that case, the covariance is $60/4 = 15$.

VTUSYNC.IN

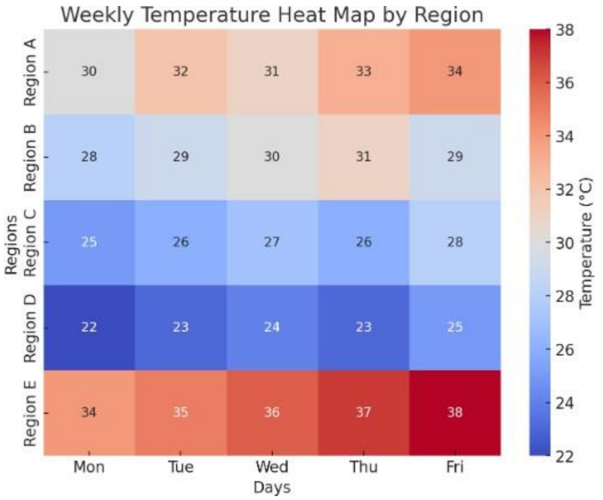
If the given attributes are X = (x1, x2, … , xN) and Y = (y1, y2, … , yN), then the Pearson correlation coefficient, that is denoted as r, is given as: (σX, σY are the standard deviations of X and Y.)

$$r = \frac{COV(X,Y)}{\sigma_x \sigma_y} \quad (2.18)$$

2.7 MULTIVARIATE STATISTICS

In machine learning, almost all datasets are multivariable. Multivariate data is the analysis of more than two observable variables, and often, thousands of multiple measurements need to be conducted for one or more subjects. Multivariate data has three or more variables. The aim of the multivariate analysis is much more. They are regression analysis, factor analysis and multivariate analysis of variance.

Heatmap A heat map is a graphical representation of data where individual values are represented by colors. Heat maps are often used in data analysis and visualization to show patterns, density, or intensity of data points in a two-dimensional grid.
Example: Let's consider a heat map to display the average temperatures (in °C) across different regions in a country over a week. Each cell in the heat map will represent a temperature for a specific region on a specific day. This is useful to quickly identify trends, such as higher temperatures in certain regions or specific days with unusual weather patterns. The **color gradient** (from blue to red) indicates the temperature range: cooler colors represent lower temperatures, while warmer colors represent higher temperatures.



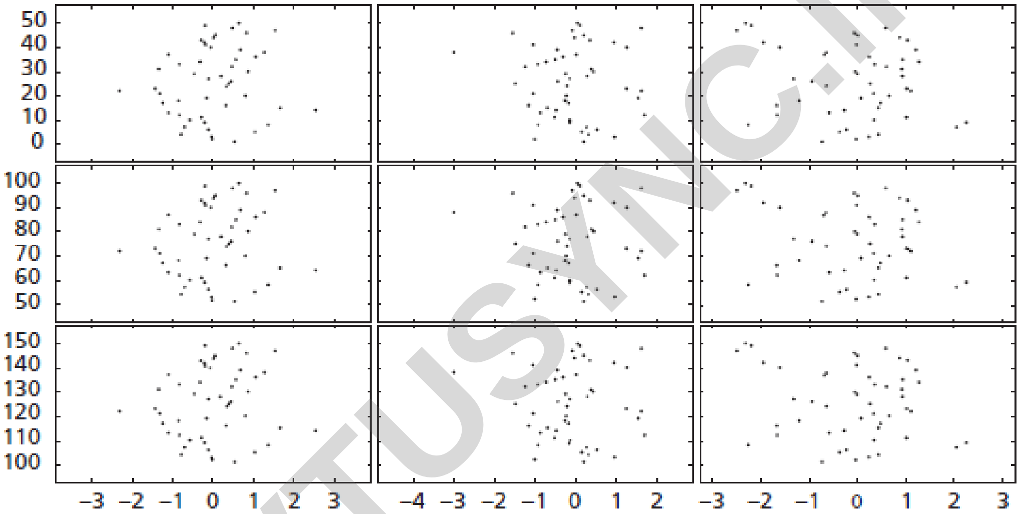


Figure 2.14: Pairplot for Random Data

VTUSYNC.IN

Pairplot

Pairplot or scatter matrix is a data visual technique for multivariate data. A scatter matrix consists of several pair-wise scatter plots of variables of the multivariate data. A random matrix of three columns is chosen and the relationships of the columns is plotted as a pairplot (or scatter matrix) as shown in Figure 2.14.

2.8 ESSENTIAL MATHEMATICS FOR MULTIVARIATE DATA

Machine learning involves many mathematical concepts from the domain of Linear algebra, Statistics, Probability and Information theory. The subsequent sections discuss important aspects of linear algebra and probability.

2.8.1 Linear Systems and Gaussian Elimination for Multivariate Data

A linear system of equations is a group of equations with unknown variables. Let Ax = y, then the solution x is given as: **$x = y/A = A^{-1}y$.** This is true if y is not zero and A is not zero. The logic can be extended for N-set of equations with ‘n’ unknown variables. **It means if A= and y=(y1 y2…yn), then the unknown variable x can be** computed as: **x= y/A=** $\mathbf{A^{-1}y}$

If there is a unique solution, then the system is called **consistent independent**. If there are various solutions, then the system is called **consistent dependant**. If there are no solutions and if the equations are contradictory, then the system is called **inconsistent**.

For solving large number of system of equations, **Gaussian elimination** can be used. The procedure for applying Gaussian elimination is given as follows:

1. **Write the given matrix.**
2. **Append vector y to the matrix A. This matrix is called augmentation matrix.**
3. **Keep the element a11 as pivot and eliminate all a11 in second row using the matrix operation,**

R_2 - (a_{21}/a_{11}), here R_2 is the 2nd row and (a_{21}/a_{11}) is called the multiplier.

The same logic can be used to remove a11 in all other equations.

4. **Repeat the same logic and reduce it to reduced echelon form. Then, the unknown variable as:**

$$x_n = \frac{y_n}{a_{nn}}$$

5. **Then, the remaining unknown variables can be found by back-substitution as:**

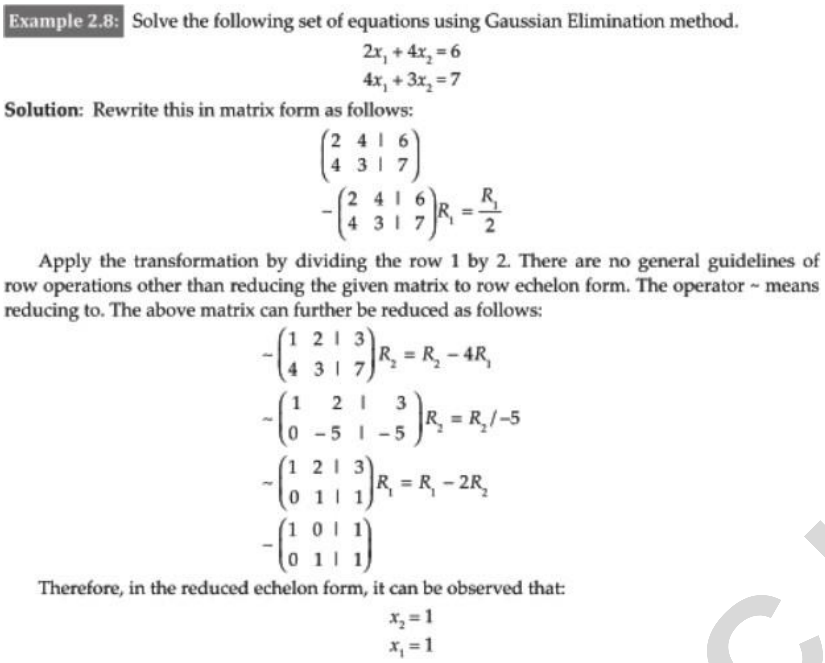
$$x_{n-1} = \frac{y_{n-1} - a_{n-1,n} \cdot x_n}{a_{n-1,n-1}}$$

To facilitate the application of Gaussian elimination method, the following row operations are applied:

1. Swapping the rows
2. Multiplying or dividing a row by a constant
3. Replacing a row by adding or subtracting a multiple of another row to it

VTUSYNC.IN

These concepts are illustrated in Example 2.8.



Example 2.8: Solve the following set of equations using Gaussian Elimination method.

$$2x_1 + 4x_2 = 6$$
$$4x_1 + 3x_2 = 7$$

Solution: Rewrite this in matrix form as follows:

$$\begin{pmatrix} 2 & 4 & | & 6 \\ 4 & 3 & | & 7 \end{pmatrix}$$
$$\sim \begin{pmatrix} 2 & 4 & | & 6 \\ 4 & 3 & | & 7 \end{pmatrix} R_1 = \frac{R_1}{2}$$

Apply the transformation by dividing the row 1 by 2. There are no general guidelines of row operations other than reducing the given matrix to row echelon form. The operator ~ means reducing to. The above matrix can further be reduced as follows:

$$\sim \begin{pmatrix} 1 & 2 & | & 3 \\ 4 & 3 & | & 7 \end{pmatrix} R_2 = R_2 - 4R_1$$
$$\sim \begin{pmatrix} 1 & 2 & | & 3 \\ 0 & -5 & | & -5 \end{pmatrix} R_2 = R_2/-5$$
$$\sim \begin{pmatrix} 1 & 2 & | & 3 \\ 0 & 1 & | & 1 \end{pmatrix} R_1 = R_1 - 2R_2$$
$$\sim \begin{pmatrix} 1 & 0 & | & 1 \\ 0 & 1 & | & 1 \end{pmatrix}$$

Therefore, in the reduced echelon form, it can be observed that:

$$x_2 = 1$$
$$x_1 = 1$$

2.8.2 Matrix Decomposition

It is often necessary to reduce a matrix to its constituent parts so that complex matrix operations can be performed.

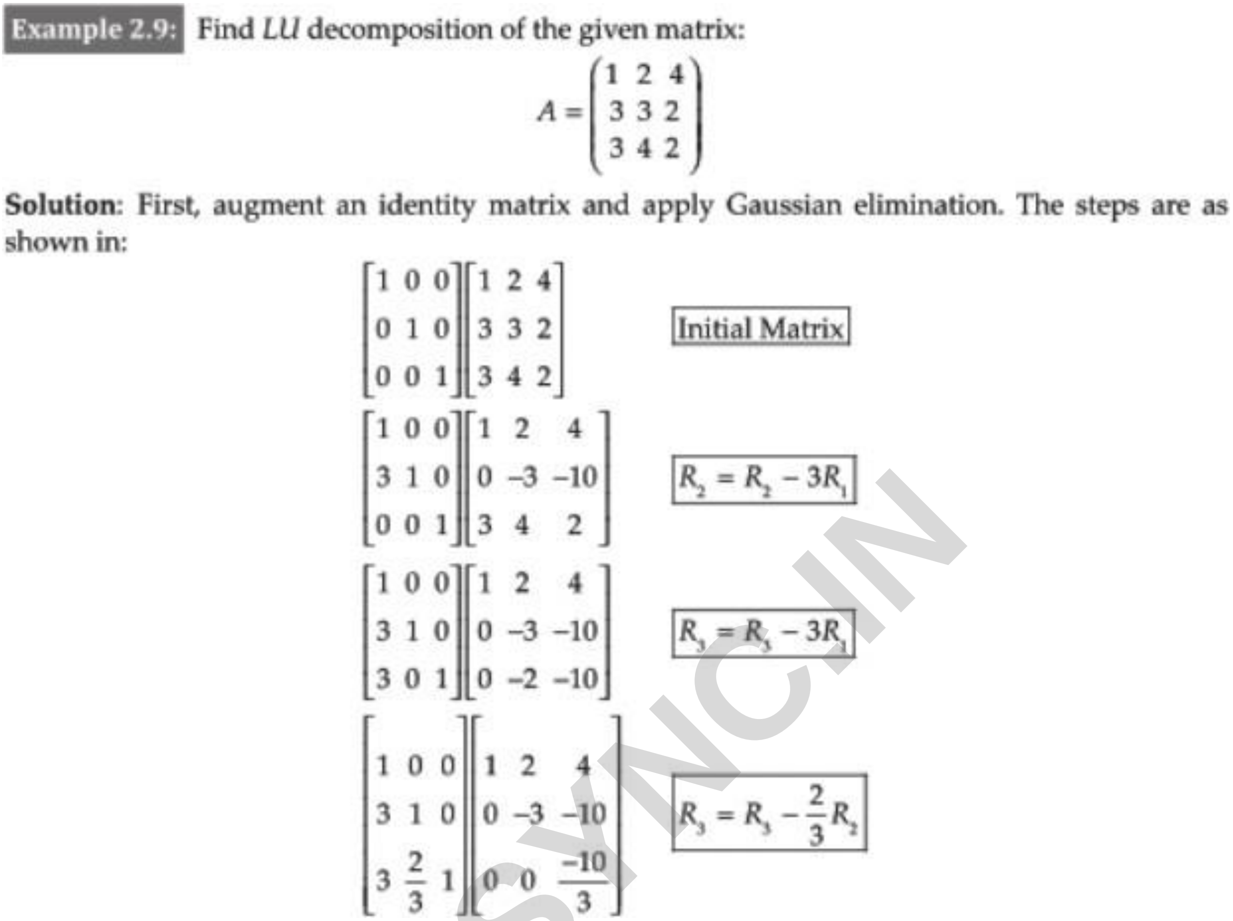
Then, the matrix A can be decomposed as: A=Q ^ Q^T

where, Q is the matrix of eigen vectors, Λ is the diagonal matrix and Q^T is the transpose of matrix Q.

LU Decomposition

One of the simplest matrix decomposition is LU decomposition where the matrix A can be decomposed matrices: A = LU. Here, L is the lower triangular matrix and U is the upper triangular matrix. The decomposition can be done using Gaussian elimination method as discussed in the previous section. First, an identity matrix is augmented to the given matrix. Then, row operations and Gaussian elimination is applied to reduce the given matrix to get matrices L and U. Example 2.9 illustrates the application of Gaussian elimination to get LU.

VTUSYNC.IN



Example 2.9: Find *LU* decomposition of the given matrix:

$$A = \begin{pmatrix} 1 & 2 & 4 \\ 3 & 3 & 2 \\ 3 & 4 & 2 \end{pmatrix}$$

Solution: First, augment an identity matrix and apply Gaussian elimination. The steps are as shown in:

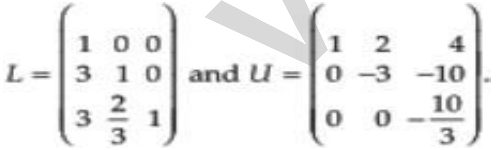
$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 2 & 4 \\ 3 & 3 & 2 \\ 3 & 4 & 2 \end{bmatrix} \quad \boxed{\text{Initial Matrix}}$$

$$\begin{bmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 2 & 4 \\ 0 & -3 & -10 \\ 3 & 4 & 2 \end{bmatrix} \quad \boxed{R_2 = R_2 - 3R_1}$$

$$\begin{bmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 3 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 2 & 4 \\ 0 & -3 & -10 \\ 0 & -2 & -10 \end{bmatrix} \quad \boxed{R_3 = R_3 - 3R_1}$$

$$\begin{bmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 3 & \frac{2}{3} & 1 \end{bmatrix}\begin{bmatrix} 1 & 2 & 4 \\ 0 & -3 & -10 \\ 0 & 0 & \frac{-10}{3} \end{bmatrix} \quad \boxed{R_3 = R_3 - \frac{2}{3}R_2}$$

Now, it can be observed that the first matrix is L as it is the lower triangular matrix whose values are the determiners used in the reduction of equations above such as 3, 3 and 2/3.
The second matrix is U, the upper triangular matrix whose values are the values of the reduced matrix because of Gaussian elimination.



$$L = \begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 3 & \frac{2}{3} & 1 \end{pmatrix} \text{ and } U = \begin{pmatrix} 1 & 2 & 4 \\ 0 & -3 & -10 \\ 0 & 0 & -\frac{10}{3} \end{pmatrix}.$$

VTUSYNC.IN

Introduction to Machine Learning and Probability/Statistics

- **Importance**: Machine learning relies heavily on statistics and probability to make predictions and analyze data.
- **Statistics in ML**: Key for understanding data patterns, measuring relationships, and quantifying uncertainties.

Probability Distributions

- **Definition**: A probability distribution describes the likelihood of various outcomes for a variable XXX.
- **Types**:

- **Discrete Probability Distributions**: For countable events (e.g., binomial, Poisson).
- **Continuous Probability Distributions**: For measurable events on a continuum (e.g., normal, exponential).

Continuous Probability Distributions

1. Normal Distribution (Gaussian Distribution)

- **Shape**: Bell curve, symmetric around the mean.
- **Characteristics**: Defined by mean μ and standard deviation σ.
- Probability Density Function (PDF)

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

- **Applications**: Common in natural data (e.g., heights, exam scores).
- **Z-score**: Standardizes data points. $Z = X - \mu/\sigma$

2. *Uniform Distribution (Rectangular Distribution)*

- **Definition**: Equal probability for all outcomes within range [a,b].
- **PDF** :

$$f(x) = \frac{1}{b-a} \quad \text{for } a \leq x \leq b$$

3. *Exponential Distribution*

Definition: Models time between events in a Poisson process

$$f(x;\lambda) = \lambda e^{-\lambda x} \quad \text{for } x \geq 0$$

Discrete Probability Distributions

1 Binomial Distribution

- **Definition**: For trials with two outcomes (success/failure).
- **Formula for Probability of k Successes in n Trials**:

$$P(X=k) = \binom{n}{k} p^k (1-p)^{n-k}$$

VTUSYNC.IN

2 Poisson Distribution

- **Definition**: Models the number of events in a fixed interval of time.
- **PDF**

$$P(X=k)=\frac{\lambda^k e^{-\lambda}}{k!}$$

3 Bernoulli Distribution

- **Definition**: Models a single trial with two outcomes (success/failure).
- **Probability Mass Function (PMF)**

$$P(X=x)=p^x(1-p)^{1-x} \quad \text{for } x=0 \text{ or } 1$$

Density Estimation

- **Goal**: Estimate the probability density function (PDF) of data.
- **Types**:
  - **Parametric Density Estimation**: Assumes a known distribution (e.g., Gaussian) and estimates parameters.
  - **Non-Parametric Density Estimation**: Does not assume a fixed distribution (e.g., Parzen window, k-Nearest Neighbors)

Parametric Density Estimation

1 Maximum Likelihood Estimation (MLE)

- **Definition**: A method for estimating the parameters of a distribution by maximizing the likelihood function.
- **Likelihood Function**: Maximize L(Θ) for parameter Θ

$$L(\theta)=P(X|\theta)$$

Gaussian Mixture Model (GMM) and Expectation-Maximization (EM) Algorithm

- **GMM**: A probabilistic model assuming data is generated from a mixture of Gaussian distributions.
- **EM Algorithm**:
  - **E-Step**: Estimate the distribution parameters for each latent variable.
  - **M-Step**: Optimize parameters using MLE.
- **Iteration**: Repeat until convergence.

VTUSYNC.IN

Non-Parametric Density Estimation Methods

1 Parzen Window

- **Definition**: A non-parametric technique that estimates the PDF based on local samples.
- **Example**: Uses a kernel function like Gaussian around each data point.

2 k-Nearest Neighbors (KNN)

- **Definition**: Estimates density by considering the kkk closest neighbors.
- **Application**: Frequently used in classification tasks.

2.10 FEATURE ENGINEERING AND DIMENSIONALITY REDUCTION TECHNIQUES

Features are attributes. Feature engineering is about determining the subset of features that form
an important part of the input that improves the performance of the model, be it classification or any other model in machine learning.

Feature engineering deals with two problems – Feature Transformation and Feature Selection.
Feature transformation is extraction of features and creating new features that may be helpful in increasing performance. For example, the height and weight may give a new attribute called Body Mass Index (BMI).

Feature subset selection is another important aspect of feature engineering that focuses on selection of features to reduce the time but not at the cost of reliability.

The features can be removed based on two aspects:
1. **Feature relevancy – Some features contribute more for classification than other features. For example, a mole on the face can help in face detection than common features like nose. In simple words, the features should be relevant.**
Feature redundancy – Some features are redundant. For example, when a database table has a field called Date of birth, then age field is not relevant as age can be computed easily from date of birth.
So, the procedure is:
1. **Generate all possible subsets**
2. **Evaluate the subsets and model performance**
3. **Evaluate the results for optimal feature selection**

Filter-based selection uses statistical measures for assessing features. In this approach, no learning algorithm is used. Correlation and information gain measures like mutual information and entropy are all examples of this approach.

Wrapper-based methods use classifiers to identify the best features. These are selected and evaluated by the learning algorithms. This procedure is computationally intensive but has superior performance.

2.10.1 Stepwise Forward Selection

This procedure starts with an empty set of attributes. Every time, an attribute is tested for statistical significance for best quality and is added to the reduced set. This process is continued till a good reduced set of attributes is obtained.

2.10.2 Stepwise Backward Elimination

This procedure starts with a complete set of attributes. At every stage, the procedure removes the worst attribute from the set, leading to the reduced set.

VTUSYNC.IN

2.10.3 Principal Component Analysis

The idea of the principal component analysis (PCA) or KL transform is to transform a given set of measurements to a new set of features so that the features exhibit high information packing properties. This leads to a reduced and compact set of features. Consider a group of random vectors of the form:

The mean vector of the set of random vectors is defined as:

The operator E refers to the expected value of the population. This is calculated theoretically using the probability density functions (PDF) of the elements xi and the joint probability density functions between the elements xi and xj. From this, the covariance matrix can be calculated as:

The mapping of the vectors x to y using the transformation can now be described as:

This transform is also called as Karhunen-Loeve or Hoteling transform. The original vector x
can now be reconstructed as follows:

If K largest eigen values are used, the recovered information would be:

The PCA algorithm is as follows:

1. **The target dataset x is obtained**
2. **The mean is subtracted from the dataset. Let the mean be m. Thus, the adjusted dataset is X – m. The objective of this process is to transform the dataset with zero mean.**
3. **The covariance of dataset x is obtained. Let it be C.**
4. **Eigen values and eigen vectors of the covariance matrix are calculated.**
5. **The eigen vector of the highest eigen value is the principal component of the dataset. The eigen values are arranged in a descending order. The feature vector is formed with these eigen vectors in its columns.**
Feature vector = {eigen vector1, eigen vector2, ... , eigen vectorn}
6. **Obtain the transpose of feature vector. Let it be A.**
7. **PCA transform is y = A × (x – m), where x is the input dataset, m is the mean, and A is the transpose of the feature vector.**

The original data can be retrieved using the formula given below:

The new data is a dimensionaly reduced matrix that represents the original data.

Figure 2.15. The scree plot indicates that only 6 out of 246 attributes are important.

From Figure 2.15, one can infer the relevance of the attributes. The scree plot indicates that the first attribute is more important than all other attributes.

2.10.4 Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) is also a feature reduction technique like PCA. The focus of LDA is to project higher dimension data to a line (lower dimension data). LDA is also used to classify the data. Let there be two classes, c1 and c2. Let m1 and m2 be the mean of the patterns of two classes. The mean of the class c1 and c2 can be computed as:

VTUSYNC.IN

The aim of LDA is to optimize the function:

2.10.5 Singular Value Decomposition

Singular Value Decomposition (SVD) is another useful decomposition technique. Let A be the matrix, then the matrix A can be decomposed as:

Here, A is the given matrix of dimension m × n, U is the orthogonal matrix whose dimension is m × n, S is the diagonal matrix of dimension n × n, and V is the orthogonal matrix. The procedure for finding decomposition matrix is given as follows:

1. For a given matrix, find AA^T
2. Find eigen values of AA^T
3. Sort the eigen values in a descending order. Pack the eigen vectors as a matrix U.
4. Arrange the square root of the eigen values in diagonal. This matrix is diagonal matrix, S.
5. Find eigen values and eigen vectors for A^TA. Find the eigen value and pack the eigen vector as a matrix called V.

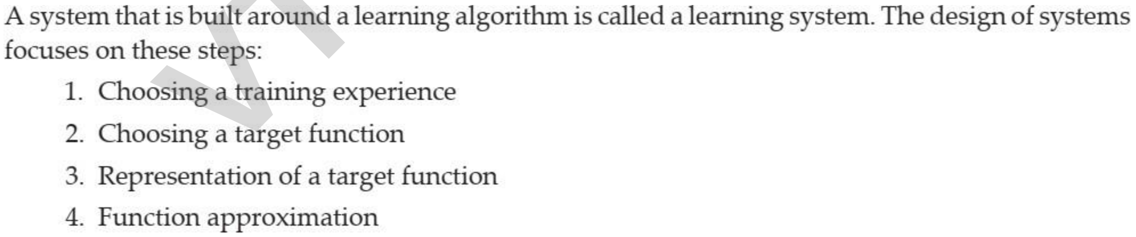
Thus, A = USV^T. Here, U and V are orthogonal matrices. The columns of U and V are left and right singular values, respectively. SVD is useful in compression, as one can decide to retain only a certain component instead of the original matrix A as:

Based on the choice of retention, the compression can be controlled.

VTUSYNC.IN

CHAPTER 3 - BASICS OF LEARNING THEORY

3.3 DESIGN OF A LEARNING SYSTEM

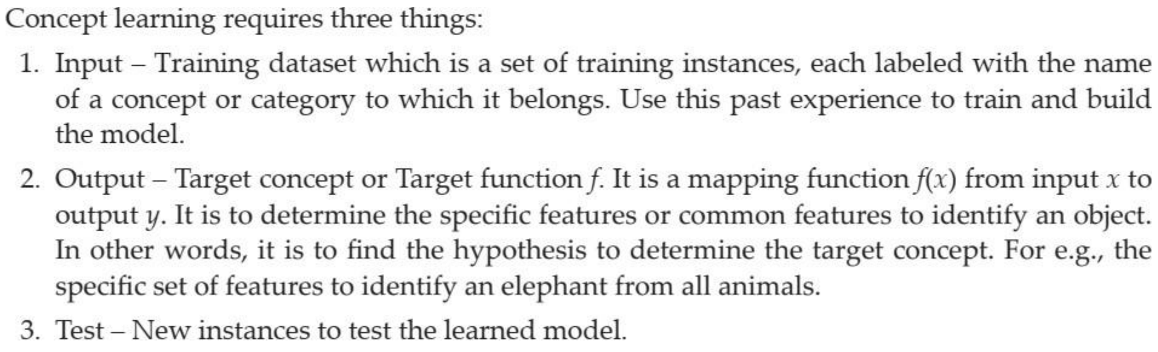


A system that is built around a learning algorithm is called a learning system. The design of systems focuses on these steps:

1. Choosing a training experience
2. Choosing a target function
3. Representation of a target function
4. Function approximation

3.4 INTRODUCTION TO CONCEPT LEARNING

Concept learning is a learning strategy that involves acquiring abstract knowledge or inferring a general concept based on the given training samples. It aims to derive a category or classification from the data, facilitating abstraction and generalization. In machine learning, concept learning is about finding a function that categorizes or labels instances correctly based on the observed features.



Concept learning requires three things:

1. Input – Training dataset which is a set of training instances, each labeled with the name of a concept or category to which it belongs. Use this past experience to train and build the model.
2. Output – Target concept or Target function *f*. It is a mapping function *f*(*x*) from input *x* to output *y*. It is to determine the specific features or common features to identify an object. In other words, it is to find the hypothesis to determine the target concept. For e.g., the specific set of features to identify an elephant from all animals.
3. Test – New instances to test the learned model.

3.4.1 Representation of a Hypothesis

A hypothesis, denoted by h, is an approximation of the target function f. It represents the relationship between independent attributes (input features) and the dependent attribute (output or label) of the training instances. The hypothesis acts as the predicted model that maps inputs to outputs effectively. In concept learning, each hypothesis is represented as a conjunction (AND combination) of attribute conditions in the antecedent part, defining specific constraints on attributes to classify instances accurately.

VTUSYNC.IN

3.4.2 Hypothesis Space

Hypothesis space is the set of all possible hypotheses that approximates the target function *f*.

The subset of hypothesis space that is consistent with all-observed training instances is called as **Version Space.**

3.4.3 Heuristic Space Search

Heuristic search is a search strategy that finds an optimized hypothesis/solution to a problem by iteratively improving the hypothesis/solution based on a given heuristic function or a cost measure.

3.4.4 Generalization and Specialization

Searching the Hypothesis Space

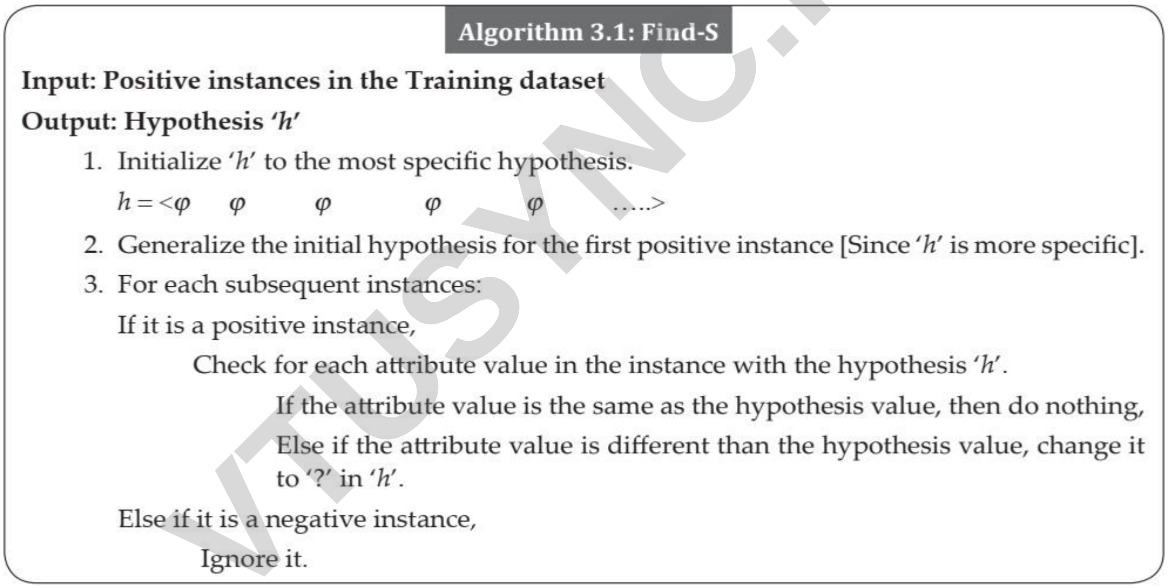
There are two ways of learning the hypothesis, consistent with all training instances from the large hypothesis space.

1. Specialization – General to Specific learning
2. Generalization – Specific to General learning

Generalization – Specific to General Learning This learning methodology will search through the hypothesis space for an approximate hypothesis by generalizing the most specific hypothesis.

Specialization – General to Specific Learning This learning methodology will search through the hypothesis space for an approximate hypothesis by specializing the most general hypothesis.

3.4.5 Hypothesis Space Search by Find-S Algorithm

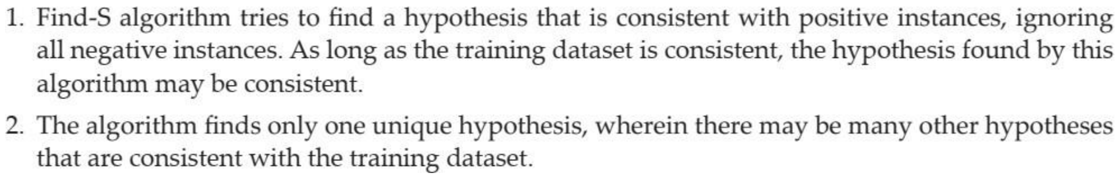


Algorithm 3.1: Find-S

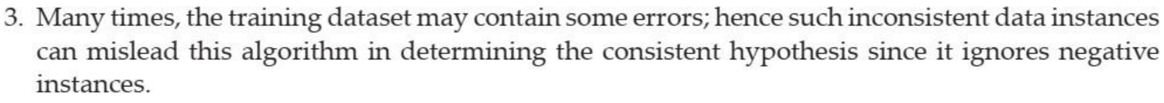
```
Input: Positive instances in the Training dataset
Output: Hypothesis 'h'
    1. Initialize 'h' to the most specific hypothesis.
       $h = <\varphi \quad \varphi \quad \varphi \quad \varphi \quad \varphi \quad .....>$
    2. Generalize the initial hypothesis for the first positive instance [Since 'h' is more specific].
    3. For each subsequent instances:
       If it is a positive instance,
             Check for each attribute value in the instance with the hypothesis 'h'.
                   If the attribute value is the same as the hypothesis value, then do nothing,
                   Else if the attribute value is different than the hypothesis value, change it
                   to '?' in 'h'.
       Else if it is a negative instance,
             Ignore it.
```

VTUSYNC.IN

Limitations of Find-S Algorithm

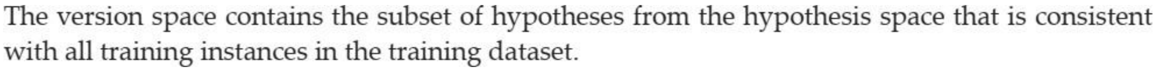


1. Find-S algorithm tries to find a hypothesis that is consistent with positive instances, ignoring all negative instances. As long as the training dataset is consistent, the hypothesis found by this algorithm may be consistent.
2. The algorithm finds only one unique hypothesis, wherein there may be many other hypotheses that are consistent with the training dataset.



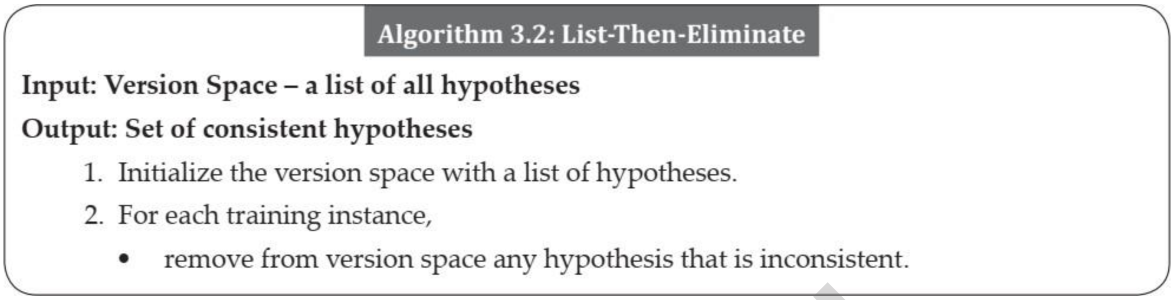
3. Many times, the training dataset may contain some errors; hence such inconsistent data instances can mislead this algorithm in determining the consistent hypothesis since it ignores negative instances.

3.4.6 Version Spaces



The version space contains the subset of hypotheses from the hypothesis space that is consistent with all training instances in the training dataset.

List-Then-Eliminate Algorithm



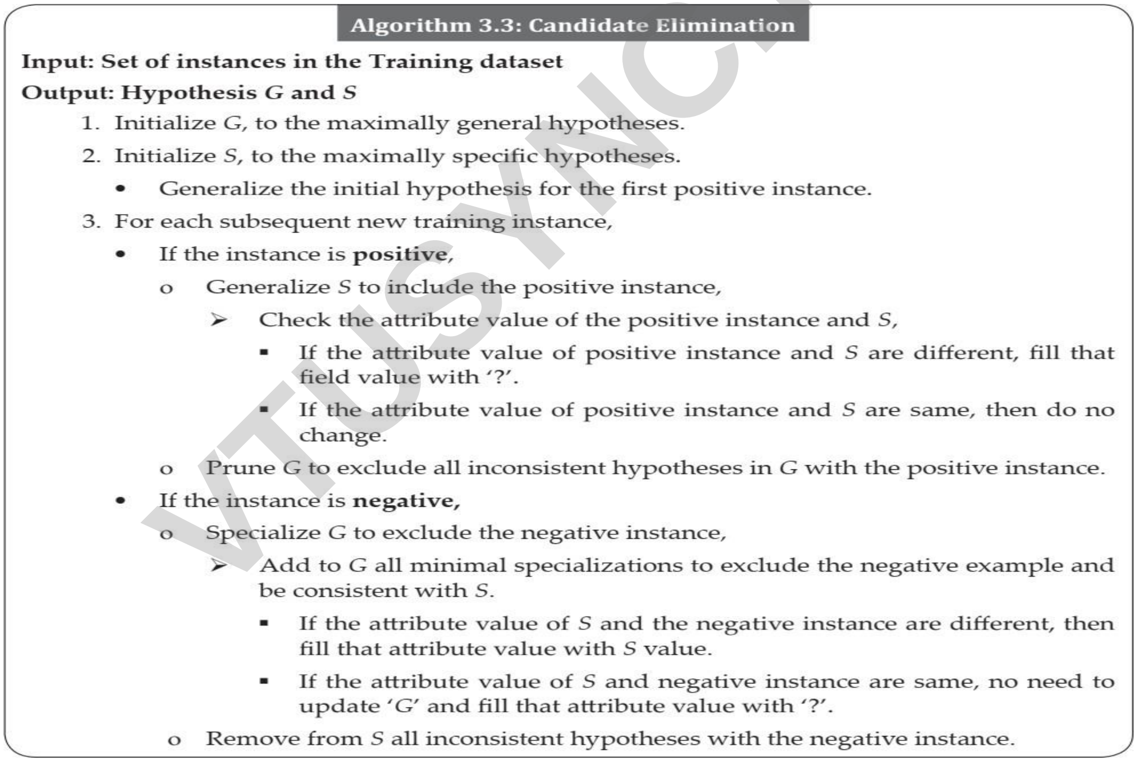
Algorithm 3.2: List-Then-Eliminate

Input: Version Space – a list of all hypotheses

Output: Set of consistent hypotheses

```
1. Initialize the version space with a list of hypotheses.
2. For each training instance,
   • remove from version space any hypothesis that is inconsistent.
```

Candidate Elimination Algorithm



Algorithm 3.3: Candidate Elimination

Input: Set of instances in the Training dataset

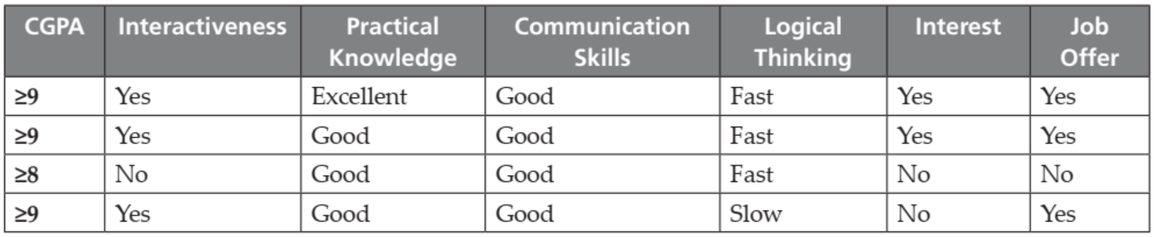
Output: Hypothesis *G* and *S*

```
1. Initialize G, to the maximally general hypotheses.
2. Initialize S, to the maximally specific hypotheses.
   • Generalize the initial hypothesis for the first positive instance.
3. For each subsequent new training instance,
   • If the instance is positive,
      o Generalize S to include the positive instance,
         ➢ Check the attribute value of the positive instance and S,
            ▪ If the attribute value of positive instance and S are different, fill that
              field value with '?'.
            ▪ If the attribute value of positive instance and S are same, then do no
              change.
      o Prune G to exclude all inconsistent hypotheses in G with the positive instance.
   • If the instance is negative,
      o Specialize G to exclude the negative instance,
         ➢ Add to G all minimal specializations to exclude the negative example and
           be consistent with S.
            ▪ If the attribute value of S and the negative instance are different, then
              fill that attribute value with S value.
            ▪ If the attribute value of S and negative instance are same, no need to
              update 'G' and fill that attribute value with '?'.
      o Remove from S all inconsistent hypotheses with the negative instance.
```

VTUSYNC.IN

The diagrammatic representation of deriving the version space is shown below:

Table 3.2: Training Dataset



CGPA	Interactiveness	Practical Knowledge	Communication Skills	Logical Thinking	Interest	Job Offer
≥9	Yes	Excellent	Good	Fast	Yes	Yes
≥9	Yes	Good	Good	Fast	Yes	Yes
≥8	No	Good	Good	Fast	No	No
≥9	Yes	Good	Good	Slow	No	Yes

Deriving the Version Space

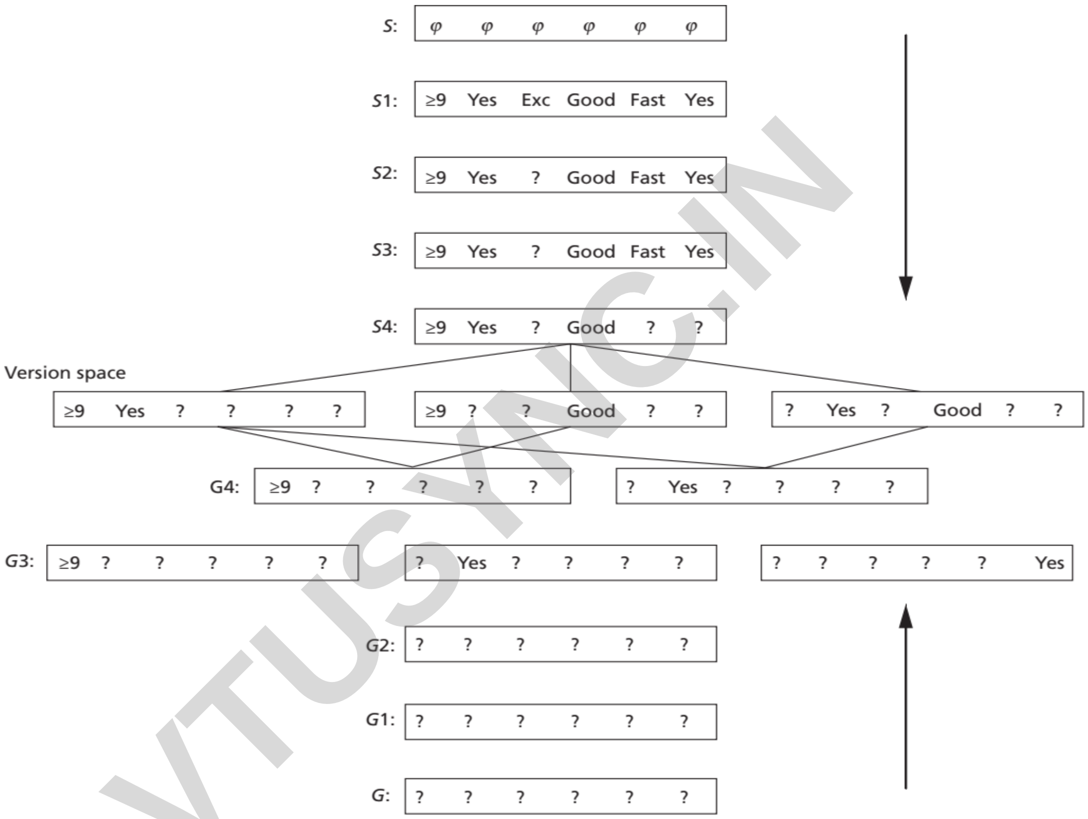


Figure 3.2: Deriving the Version Space

VTUSYNC.IN