Machine Learning

Master Supervised and Unsupervised Learning Algorithms with Real Examples

DR RUCHI DOSHI

DR KAMAL KANT HIRAN

RITESH KUMAR JAIN

DR KAMLESH LAKHWANI

bpb

Machine Learning

Master Supervised and Unsupervised Learning Algorithms with Real Examples

Dr Ruchi Doshi

Dr Kamal Kant Hiran

Ritesh Kumar Jain

Dr Kamlesh Lakhwani

www.bpbonline.com

FIRST EDITION 2022

Copyright © BPB Publications, India

ISBN: 978-93-91392-352

LIMITS OF LIABILITY AND DISCLAIMER OF WARRANTY

The information contained in this book is true to correct and the best of author's and publisher's knowledge. The author has made every effort to ensure the accuracy of these publications, but publisher cannot be held responsible for any loss or damage arising from any information in this book.

All trademarks referred to in the book are acknowledged as properties of their respective owners but BPB Publications cannot guarantee the accuracy of this information.

www.bpbonline.com

Foreword

Recently, machine learning has been utilized by the governments, businesses and general public for different purposes. Machine learning has assisted organizations by generating profit, and its popularity among developers and technologists has skyrocketed. To best assist the readers in better understanding of the facts, this book is well arranged in a way that teaches them what the facts are.

— Dr. Ricardo Saavedra
Director & Chair International Programs
Universidad Azteca, Mexico

In today's digital age, mastering Machine Learning is a must. This book will elegantly guide you through everything you need to know about this topic.

— Dr. Govind Kumawat
Indian Institute of Management, Udaipur, India

The authors provide an easy-to-understand and comprehensive overview of Machine Learning concepts. The explanation is clear and concise, with appropriate diagrams and real-world examples that help to demystify this emerging technology.

— Dr. Deepak Khazanchi
University of Nebraska at Omaha, USA

This book covers a wide range of learning approaches, with machine learning techniques and algorithms with detailed examples to accompany each approach.

— Dr. Samuel Agbesi
Aalborg University, Denmark

The adoption and prevalence of Artificial Intelligence and Machine Learning in our daily lives are the two most significant technological shifts in the 21st century. This book explains the concepts of Machine Learning technologies in a concise, clear and lucid manner.

— Dr. Shiva Raj Pokhrel
Deakin University, Australia

A genuine book for those who want to learn and apply Machine Learning concepts.

— Prof. Dr. Dharm Singh
Namibia University of Science and Technology, Namibia

Machine Learning is a fascinating and important research topic these days. The book also transitions from academic to research topics. As a result, it is extremely beneficial to any researcher or academician, from beginner to advanced level.

— Trilok Nuwal
Microsoft, India

The book is extremely comprehensive and can be used in conjunction with any university's curriculum. The best part of the book is that it discusses machine learning algorithms with real-world examples and practical applications.

— Dr. Tanima Dutta
Indian Institute of Technology (BHU), India

Machine Learning is a game changer in the age of digitization. This book covers almost every aspect of Machine Learning, from the fundamentals to the application level.

— Abhishek Maloo
Twitter, California, USA

Machine Learning, like electricity, will revolutionize our lives in a variety of ways, some of which are not even imaginable today. This book offers a comprehensive conceptual understanding of Machine Learning techniques and algorithms.

— Desmond Okocha, PhD
Bingham University, Nigeria

This book provides an in-depth introduction to Machine Learning even to readers with no pre-requisite knowledge. Many mathematical concepts are explained in an easy-to-understand manner.

— Dr. Patrick Acheampong
Ghana Communication Technology University, Ghana

Provides a comprehensive overview of available techniques and algorithms in conceptual terms, encompassing a variety of machine learning application domains.

— Dr. Sumarga Kumar Sah Tyagi
Zhongyuan University of Technology, China

This book covers everything fundamental to machine learning, to immerse yourself in the theory of the topic and to use practical applications and examples to promote knowledge.

— Dr. Albert Gyamfi

Saskatchewan Polytechnic, Canada

In addition to covering the theoretical aspects of machine learning, the authors teach the various techniques for obtaining data as well as how to use different inputs and outputs to evaluate results. Machine learning is dynamic, so the methods are always evolving.

— Do Manh Thai
Govt. Executive, Vietnam

This book includes the popular learning algorithms, techniques and implementations in the artificial intelligence field. I strongly advise this book.

— Prof. Vinesh Jain
Govt. Engineering College, Ajmer, India

Dedicated to

Our lovely little daughter Miss Bhuvi Jain.
Your endless love and energy charge every day.

– Dr. Ruchi Doshi and
Dr. Kamal Kant Hiran

About the Authors

Dr. Ruchi Doshi has more than 14 years of academic, research and software development experience in Asia and Africa. Currently, she is working as a **Research Supervisor** at the **Universidad Azteca, Mexico** and Adjunct faculty at the Jyoti Vidyapeeth Women's University, Jaipur, Rajasthan, India. She has worked in the BlueCrest University College, Liberia, West Africa as Registrar and Head, Examination, BlueCrest University College, Ghana, Africa, Amity University, Rajasthan, India and Trimax IT Infrastructure and Services, Udaipur, India.

She has been nominated by the IEEE Headquarter, USA for the Chair, Women in Engineering and Secretary Position in Liberia country. She has worked with the Ministry of Higher Education (MoHE) in Liberia and Ghana for the Degree approval and accreditation processes. She is interested in the field of Machine Learning and Cloud computing framework development. She has given many expert talks in the area of Women in Research, Use of Machine Learning Technology in Real-time Applications and Community Based Participatory Action Research at the national and international level. She has published 25 scientific research papers in SCI/Scopus/Web of Science Journals, Conferences, 2 Indian Patents and 4 books with internationally renowned publishers. She is a reviewer, advisor, ambassador and editorial board member of various reputed international journals and conferences. She is an active member in organizing many international seminars, workshops and conferences in Mexico, India, Ghana and Liberia.

LinkedIn Profile: **https://www.linkedin.com/in/dr-ruchi-doshi-96bb63b4/**

Dr. Kamal Kant Hiran is an Assistant Professor, School of Engineering at the **Sir Padampat Singhania University (SPSU),** Udaipur, Rajasthan, India as well as a Research Fellow at the **Aalborg University, Copenhagen, Denmark**. He has more than 16 years of experience as an academic and researcher in Asia, Africa and Europe. He has worked as an Associate Professor and Head, Academics at the BlueCrest University College, Liberia, West Africa, Head of Department at the Academic City College, Ghana, West Africa, Senior Lecturer at the Amity

University, Jaipur, Rajasthan, India, Assistant Professor at the Suresh Gyan Vihar University, Jaipur, Rajasthan, India and Visiting Lecturer at the Government Engineering College, Ajmer.

He has several awards to his credit such as the international travel grant for attending the 114th IEEE Region 8 Committee meeting in Warsaw, Poland, International travel grant for Germany from ITS Europe, Passau, Germany, Best Research Paper Award at the University of Gondar, Ethiopia and SKIT, Jaipur, India, IEEE Liberia Subsection Founder Award, Gold Medal Award in M. Tech (Hons.), IEEE Ghana Section Award - Technical and Professional Activity Chair, IEEE Senior Member Recognition, IEEE Student Branch Award and Elsevier Reviewer Recognition Award. He has published 35 scientific research papers in SCI/Scopus/Web of Science and IEEE Transactions Journal, Conferences, 2 Indian Patents, 1 Australian patent grant and 9 books with internationally renowned publishers. He is a reviewer and editorial board member of various reputed international journals in Elsevier, Springer, IEEE Transactions, IET, Bentham Science and IGI Global. He is an active member in organizing many international seminars, workshops and conferences. He has made several international visits to Denmark, Sweden, Germany, Poland, Norway, Ghana, Liberia, Ethiopia, Russia, Dubai and Jordan for research exposures. His research interests focus on Cloud Computing, Machine Learning and Intelligent IoT.

LinkedIn Profile: **https://www.linkedin.com/in/kamal-kant-hiran-phd-4553b643/**

Mr. Ritesh Kumar Jain works as an Assistant Professor at the **Geetanjali Institute of Technical Studies, (GITS), Udaipur,** Rajasthan, India. He has more than 15 years of teaching and research experience. He has completed his BE and MTech. He has worked as an Assistant Professor and Head of Department at S.S. College of Engineering. Udaipur, Assistant Professor at Sobhasaria Engineering College, Sikar and Lecturer at the Institute of Technology and Management, Bhilwara.

He is a reviewer of international peer-reviewed journals. He is the author of several research papers in peer-reviewed international journals and conferences.

LinkedIn Profile: **https://www.linkedin.com/in/ritesh-jain-b8924345/**

Dr. Kamlesh Lakhwani works as an Associate Professor at the School of Computer Science and Engineering, **JECRC University, Jaipur, Rajasthan**, India. He has an excellent academic background and a rich experience of 15 years as an academician and researcher in Asia. As a prolific writer in the arena of Computer Sciences and Engineering, he has penned down several learning materials on C, C++, Multimedia Systems, Cloud Computing, IoT, Image Processing, etc. He has four published patents to his credit and contributed for more than 50 research papers in the Conferences/Journals/Seminar of International and National repute. His area of interest includes Cloud Computing, Internet of Things, Computer vision, Image processing, Video Processing and Machine Learning.

LinkedIn Profile: **https://www.linkedin.com/in/dr-kamlesh-lakhwani-7119944b/**

About the Reviewer

Dr. Ajay Kumar Vyas has more than 15 years of teaching and research experience and is presently working as an Assistant Professor at Adani Institute of Infrastructure Engineering, Ahmedabad (India). He has completed his Bachelor of Engineering (2005) in Electronics and Communication from Govt. Engineering College, Ujjain and M.Tech (2009) in Optical Communication from Shri Govindram Sakseriya Institute of Tech and Sci., Indore with Honors and PhD (2016) from Maharana Pratap Agri. and Tech. University, Udaipur (Raj). He is a senior member of IEEE and IACSIT (Singapore). He has been awarded certificate of excellence from Elsevier Research Academic and Publons Academy as a certified peer reviewer.

He has worked as a reviewer for renowned journals of Springer, IET, IEEE, OSA, IGI Global, Chinese Journal of Electrical Engineering and many more.

He is the author of several research papers in peer-reviewed international journals and conferences, three books with De-Gruyter and India Publications and has published two Indian patents. He is also the author of many book chapters published by Springer International Publishing, Singapore.

Acknowledgement

First and foremost, we'd like to thank the Almighty for giving us the inspiration and zeal to write this book.

Our sincere thanks goes to our organizations, Universidad Azteca, Mexico, Sir Padampat Singhania University, Geetanjali Institute of Technical Studies, JECRC University, India for providing us with a healthy academic and research environment during work.

Special thanks to the BPB Publications team, especially to Nrip Jain and members for their support, advice and assistance in editing and publishing this book.

The completion of this book could not have been possible without the contribution and support we got from our family, friends and colleagues. It is a pleasant aspect and we express our gratitude to all of them.

— Dr. Ruchi Doshi
Universidad Azteca,
Mexico

— Dr. Kamal Kant Hiran
Sir Padampat Singhania University (SPSU),
India

— Mr. Ritesh Kumar Jain
Geetanjali Institute of Technical Studies (GITS),
India

— Dr. Kamlesh Lakhwani
JECRC University,
India

Preface

Machine learning is an application of Artificial Intelligence (AI). While AI is the umbrella term given to machines emulating human abilities, machine learning is a specific branch of AI where machines are trained to learn how to process and make use of data. The objective of machine learning is not only for effective data collection but also to make use of the ever-increasing amounts being gathered by manipulating and analyzing them without heavy human input.

Machine learning can be defined as a method of mathematical analysis, often using well-known and familiar methods, with a different focus than the traditional analytical practice in applied subjects. The key idea is that flexible and automated methods are used to find patterns within data with a primary focus on making predictions for future data.

There are several real-time applications of machines such as Image Recognition, Biometric Recognition, Speech Recognition, Handwriting Recognition, Medical Diagnosis, Traffic prediction, Text Retrieval, Product recommendations, Self-driving cars, Virtual Personal Assistants, Online Fraud Detection, Natural Language Processing and so on.

Machine Learning paradigms are defined in three types namely Supervised Learning, Unsupervised Learning and Reinforcement Learning. **Supervised learning** algorithms are designed to learn by example. It is called supervised learning because the process of an algorithm learning from the training dataset can be thought of as a teacher supervising the learning process. **Unsupervised learning** deals with unlabelled data which means here we have input data and no corresponding output variable. This is further classified into Clustering and Association. In **Reinforcement Learning**, the machine or agent automatically learns using feedback without any labelled data. Here, the agent learns by itself from its experience.

In this book, the reader will not only find the theoretical concepts but also the practical knowledge needed to quickly and efficiently apply these strategies to challenging problems of machine learning. The reader learns how to understand a problem, be able to represent data, select and correct skills, interpret results correctly and practice effective analysis of outcomes to make strategic decisions.

Organization of the Book

The book consists of six chapters, in which the reader will learn the following:

Chapter 1 introduces the fundamental concepts of machine learning, its applications, types and describes the setup we will be using throughout the book.

Chapter 2 describes supervised machine learning. Different supervised machine learning algorithms such as Linear Regression Model, Naive Bayes classifier Decision Tree, K nearest neighbor, Logistic Regression, Support Vector Machine, and Random forest algorithm are described in this chapter with their practical use.

Chapter 3 describes unsupervised machine learning. Different unsupervised machine learning algorithms such as K-Means Clustering, Hierarchical Clustering, Probabilistic Clustering, Association Rule Mining, Apriori Algorithm, f-p Growth Algorithm, Gaussian Mixture Model are described in this chapter with their practical use.

Chapter 4 describes the various statistical learning theories used in machine learning. This chapter describes statistical learning theories such as Feature Extraction, Principal Component Analysis, Singular Value Decomposition, Feature Selection - feature ranking and subset selection, filter, wrapper and embedded methods, Evaluating Machine Learning Algorithms and Model Selection.

Chapter 5 describes Semi-Supervised Learning and Reinforcement Learning. This chapter describes Markov Decision Process (MDP), Bellman Equations, Policy Evaluation using Monte Carlo, Policy Iteration and Value Iteration, Q-Learning, State Action-Reward-State-Action (SARSA) and Model-Based Reinforcement Learning.

Chapter 6 describes the recommended system and basic introduction to neural networks and deep learning. This chapter includes various techniques used for the recommended system such as Collaborative Filtering and Content-Based Filtering. It also covers the basic introduction of Artificial Neural Network, Perceptron, Multilayer network, Backpropagation and introduction to Deep Learning.

At the end of this book, **practicals** and **model question papers** are included for practice.

Downloading the coloured images:

Please follow the link to download the
Coloured Images of the book:

https://rebrand.ly/dd73a7

Errata

We take immense pride in our work at BPB Publications and follow best practices to ensure the accuracy of our content to provide with an indulging reading experience to our subscribers. Our readers are our mirrors, and we use their inputs to reflect and improve upon human errors, if any, that may have occurred during the publishing processes involved. To let us maintain the quality and help us reach out to any readers who might be having difficulties due to any unforeseen errors, please write to us at :

errata@bpbonline.com

Your support, suggestions and feedbacks are highly appreciated by the BPB Publications' Family.

BPB is searching for authors like you

If you're interested in becoming an author for BPB, please visit **www.bpbonline.com** and apply today. We have worked with thousands of developers and tech professionals, just like you, to help them share their insight with the global tech community. You can make a general application, apply for a specific hot topic that we are recruiting an author for, or submit your own idea.

The code bundle for the book is also hosted on GitHub at **https://github.com/bpbpublications/Machine-Learning**. In case there's an update to the code, it will be updated on the existing GitHub repository.

We also have other code bundles from our rich catalog of books and videos available at **https://github.com/bpbpublications**. Check them out!

PIRACY

If you come across any illegal copies of our works in any form on the internet, we would be grateful if you would provide us with the location address or website name. Please contact us at **business@bpbonline.com** with a link to the material.

If you are interested in becoming an author

If there is a topic that you have expertise in, and you are interested in either writing or contributing to a book, please visit **www.bpbonline.com**.

REVIEWS

Please leave a review. Once you have read and used this book, why not leave a review on the site that you purchased it from? Potential readers can then see and use your unbiased opinion to make purchase decisions, we at BPB can understand what you think about our products, and our authors can see your feedback on their book. Thank you!

For more information about BPB, please visit **www.bpbonline.com**.

Table of Contents

1. Introduction to Machine Learning...1

Introduction..1

Structure 2

Objectives...2

What is Machine Learning?...2

Machine Learning versus Traditional Programming 4

The Seven Steps of Machine Learning...5

Step 1: Data Gathering / Data Collection 6

Step 2: Preparing the data 6

Step 3: Choosing a Model 6

Step 4: Training 7

Step 5: Evaluation..7

Step 6: Hyperparameter Tuning 7

Step 7: Prediction...8

Applications of Machine Learning...8

Social Media Features...8

Product Recommendations 8

Image & Speech Recognition 9

Sentiment Analysis...9

Self-driving cars...9

Email Spam and Malware Filtering 9

Stock Market Trading 10

Medical Diagnosis 10

Online Fraud Detection..10

Automatic language translation...10

Types of Machine Learning...11

Supervised Learning...11

Regression..12

Classification 13

Unsupervised Learning 13

Clustering...14

Association 15

Reinforcement Learning 15

Advantages of Machine Learning 16

Disadvantages of Machine Learning 17

Most Popular Machine Learning Software Tools 18

Summary 20

Exercise (MCQ's) 22

Answers 25

Fill in the blanks 26

Answers 26

Descriptive questions 26

2. Supervised Learning Algorithms 27

Introduction 27

Structure 27

Objectives 28

Introducing Supervised Learning 28

Types of Supervised Learning 29

Regression 30

Terminologies used in Regression 31

Types of Linear Regression 34

Simple Linear Regression 34

Multiple Linear Regression 36

Classification 36

Naïve Bayes classifier algorithm 37

Why is it called Naïve Bayes? 38

Principle of Naive Bayes Classifier 38

Working of Naïve Bayes' Classifier 39

Naive Bayes Example 39

Types of Naïve Bayes 39

Advantages and disadvantages of Naïve Bayes 40

Applications of Naïve Bayes Algorithms 40

Decision Tree 41

Decision-tree working 41

Example of decision-tree 42

Advantages of decision tree 43

Disadvantages of the decision tree 43

K-Nearest Neighbors (K-NN) algorithm 43

Need of the K-NN Algorithm 43

Logistic Regression 46

Comparison between linear and logistic regression 47

Types of Logistic Regression 48

Binary Logistic Regression 48

Multinomial Logistic Regression 49

Ordinal Logistic Regression 49

Examples 49

Support Vector Machine (SVM) Algorithm 49

Hyperplane, Support Vectors, and Margin 51

Working of SVM 52

Types of SVM 52

Applications of Support-Vector Machines 53

Advantages of SVM 53

Disadvantages of SVM 53

Random Forest Algorithm 54

Working of the Random Forest Algorithm 55

Advantages of Random Forest Algorithm 56

Disadvantages of Random Forest Algorithm 56

Applications of Random Forest Algorithm 57

Summary 57

Exercise (MCQ's) 58

Answer 60

Short Answers Questions 60

Long Answers Questions 61

3. Unsupervised Learning 63

Structure 63

Objectives 64

Unsupervised Learning 64

Working of unsupervised learning 64

Need for using unsupervised learning 65

Algorithms 65
Clustering 65
K-means Clustering 67
Algorithm of k-mean clustering 67
Flowchart of k-mean clustering 68
A practical example of k-mean clustering 68
Hierarchical clustering 73
Two approaches to hierarchical clustering 73
Need of hierarchical clustering 73
Agglomerative hierarchical clustering 74
Working of agglomerative hierarchical clustering 74
Measuring the distance between two clusters 75
The dendrogram in hierarchical clustering 77
Creating Dendrogram 78
An example of hierarchical clustering 78
Association rule learning 81
Working of the association rule learning 82
Support 82
Confidence 82
Lift 83
Apriori Algorithm 83
Frequent Itemset 83
Steps of Apriori Algorithm 84
Example of the apriori algorithm 84
FP-Growth Algorithm 87
Frequent Pattern (FP) Tree 87
Steps of the FP-growth algorithm 88
An example of FP-growth algorithm: 88
Difference between Apriori and FP-Growth 93
Applications of the association rule learning 93
Probabilistic clustering 93
Gaussian Distribution 95
Gaussian Mixture Models (GMMs) 95
Expectation-Maximization 97
Expectation Maximization in GMMs 98

E-step ... *98*
M-step ... *98*
An example of Gaussian Mixture Model ... *99*
Summary ... 102
Exercise (MCQ's) ... 102
Answers ... *104*
Short Answers Questions ... 104
Long answers questions ... 105

4. Introduction to Statistical Learning Theory ... 107
Structure ... 108
Objective ... 108
Introduction to statistical learning ... 108
Estimation of unknown function f ... *111*
Prediction ... *111*
Inference ... *112*
Supervised verses unsupervised learning ... *113*
Regression Verses classification ... *113*
Feature selection ... 114
Filters ... *114*
Pearson correlation ... *115*
Chi-squared ... *115*
Linear discriminant analysis (LDA) ... *115*
Analysis of variance (ANOVA) ... *115*
Wrappers ... *115*
Embedded methods ... *116*
Model selection ... 116
Re-sampling methods ... *116*
Probabilistic measures ... *117*
Akaike Information Criterion (AIC) ... *117*
Minimum Description Length (MDL) ... *118*
Model evaluation ... 119
Classification metrics ... *119*
Accuracy ... *119*
Precision ... *120*

Recall *120*
F1 Score *120*
AUC curve *120*
Regression metrics *121*
Mean Squared Error or MSE *121*
Root Mean Squared Error or RMSE *121*
Mean Absolute Error or MAE *121*
Root Mean Squared Log Error or RMSLE *121*
Clustering metrics *122*
Dunn Index *122*
Silhouette Coefficient *122*
Elbow method *122*
Statistical learning algorithms *123*
Supervised learning 124
Regression *124*
Linear Regression *125*
Logistic Regression *125*
Polynomial Regression *126*
Support Vector Regression *126*
Decision Tree Regression *127*
Ridge Regression *127*
Lasso Regression *127*
Classification *127*
Naive Bayes Classifier *128*
Nearest Neighbor algorithm *128*
Logistic Regression *129*
Decision Trees *129*
Neural Network *129*
Unsupervised learning 129
Hierarchical Clustering *130*
K-means Clustering *130*
K- Nearest neighbors *130*
Principal Components Analysis *131*
Summary 131
Exercise (MCQ's) 132
Answers *134*

Short question answer 134
Long question answer 134

5. Semi-Supervised Learning, Reinforcement Learning 135

Introduction 135
Structure 135
Objectives 136
Semi-supervised learning 136
Markov Decision Process (MDP) 137
Markov Chain and Markov Process 137
Applications of Markov Decision Process 139
Bellman Equations 139
Going Deeper into Bellman Expectation Equation 140
Monte Carlo Methods 143
Monte Carlo Policy Evaluation 143
Policy iteration and value iteration 145
Policy Iteration 145
Value Iteration 146
Q-Learning 148
Introducing the Q-Table 148
The Q-Learning algorithm 148
State-Action-Reward-State-Action (SARSA) 150
SARSA Vs Q-learning 151
SARSA Algorithm 151
Model-based Reinforcement Learning 152
What is a Model? 153
Difference between Model-Based and Model-Free 153
Learning the Model 153
Concrete Example 154
Dyna Architecture 156
Summary 156
Exercise (MCQ's) 159
Answers 160
Descriptive Questions 161

6. Recommended Systems 163

Introduction 163

Structure 163

Objective 164

Recommended Systems 164

Importance of recommended systems 165

Types of recommended systems 165

Recommended Systems (RS) functions 167

Applications and Challenges of Recommended Systems 168

Applications 168

Challenges 169

Collaborative Filtering 170

Collaborative Filtering classes 171

Types of collaborative filtering techniques 171

Content-based filtering 173

The architecture of content-based filtering 173

Advantages of content-based filtering 175

Drawbacks of content-based filtering 175

Artificial Neural Network (ANN) 176

Supervised ANN 176

Unsupervised ANN 176

Applications 177

Comparison between Artificial Neural Networks (ANN) and Biological Neural Networks (BNN) 177

Characteristics of the ANN 178

Perceptron 178

Activation Function 179

Multilayer Network 181

Backpropagation algorithm 182

Types of Backpropagation Networks 183

Advantages 184

Disadvantages 184

Introduction to Deep Learning 184

How do deep learning algorithms "learn"? 185

Difference between Machine Learning and Deep Learning *185*
Importance of Deep Learning *186*
Applications of Deep Learning *188*
Summary 188
Exercise (MCQ's) 189
Answers *191*
Short Answers Questions 191
Long Answers Questions 192
Bibliography **193**
Practicals **203**
Test Papers **247**
Index **263-270**

CHAPTER 1

Introduction to Machine Learning

> **"Machine Learning is the field of study that gives computers the ability to learn without being explicitly programmed".**
>
> *~ Arthur l. Samuel, Computer scientist*

Introduction

Machine learning is an application of Artificial Intelligence. While AI is the umbrella term given to machines emulating human abilities, machine learning is a specific branch of AI where machines are trained to learn how to process and make use of data. The objective of machine learning is not only effective data collection but also to make use of the ever-increasing amounts of data being gathered by manipulating and analyzing them without heavy human input.

This chapter takes on a proactive and practical approach to discuss the foundation, background concepts, characteristics as well as the pros and cons of Machine Learning. We will discuss from traditional programming practices to Machine Learning. This chapter includes the types of Machine Learning as well as their advantages & disadvantage.

Structure

In this chapter, we will discuss the following topics:

- What is Machine Learning?
- Machine Learning Vs. Traditional Programming
- The Seven Steps of Machine Learning
- Types of Machine Learning
- Advantages and disadvantages of Machine Learning
- Popular Machine Learning Software Tools
- Tools used for practical with some examples

Objectives

By completing this chapter, you will be able to:

- Understand the concepts of Machine Learning.
- Understand the difference Between Traditional programming & Machine Learning.
- Understand about the state-of-art applications of Machine Learning.
- Understand about the types of Machine Learning.
- Understand the significance of the Machine Learning applications
- Understand about the tools of Machine Learning

What is Machine Learning?

Machine learning is a sub-domain of **artificial intelligence** (**AI**). The goal of machine learning is usually to understand the structure of the data and to match that data to models that can be understood and used by humans.

While artificial intelligence and machine learning are often used together, they are two different concepts. AI is a broad concept – decision-making machines, learning new skills, and problem-solving in the same way for people - and machine learning is an AI set that enables intelligent systems to independently learn new things from the data.

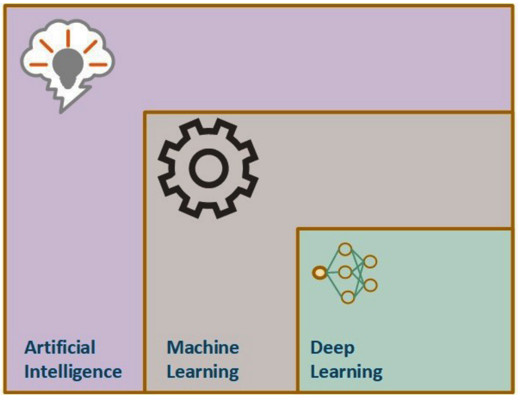


***Figure 1.1**: Machine Learning subset*

The *Figure 1.1* shows that machine learning is a subset of artificial intelligence. Machine learning is a tool for transforming information into knowledge. In the previous 50 years, there has been a blast of information/data. This mass of information is pointless except if we investigate it and discover the examples covered up inside. Machine learning techniques are utilized to consequently locate the significant fundamental examples inside complex information that we would somehow battle to find. Hidden patterns and information about the problem can be used to predict future events and to make all sorts of complex decisions.

We have seen machine learning as a trendy expression for hardly any years, the meaning behind this could be the high rate of data/information creation by applications, the expansion of computation power over the years, and the development of better algorithms.

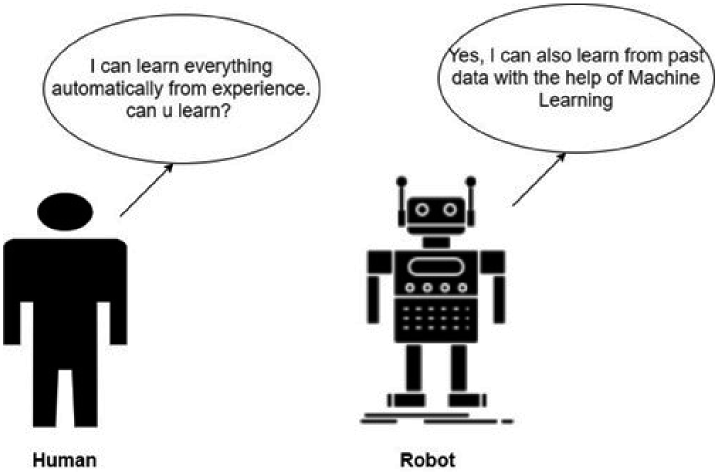

I can learn everything automatically from experience. can u learn?
Yes, I can also learn from past data with the help of Machine Learning
Human
Robot

***Figure 1.2**: Human & Robot*

The *Figure 1.2*, shows that the human learns everything automatically from experience & the robot can also learn from previous experience data with the help of machine learning.

The name machine learning was coined by Arthur Samuel in 1959, an American pioneer in the field of computer gaming and artificial intelligence who stated that "*Machine Learning gives computers the ability to learn without being explicitly programmed*". Arthur Samuel created the first self-study program for playing checkers. You realize that the more the system plays, the better it performs.

And in 1997, *Tom Mitchell* gave a "*well-established*" mathematical and relational definition that "*A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E*".

Example 1: Playing Chess

- **Task (T)**: The task of playing chess
- **Experience (E)**: The experience of playing many games of chess
- **Performance Measure (P)**: The probability of the program which will win the next game of chess

Example 2: Spam Mail Detection

- **Task (T)**: To recognize and classify the emails into '*spam*' or '*not spam*'.
- **Experience (E)**: A set of emails with given labels ('*spam*' / '*not spam*').
- **Performance Measure (P)**: Total percentage of emails being correctly classified as 'spam' (or 'not spam') by the program.

The basic premise of machine learning is to build algorithms that can obtain input data and use mathematical analysis to predict output while reviewing results as new data becomes available.

Currently, **machine learning (ML)** is used for a variety of tasks such as *image recognition, speech recognition like Amazon's Alexa, email filtering like email is spam or not spam, Facebook auto-tagging, recommendation systems like Amazon and Flipkart recommends the product to the user* and many more.

Machine Learning versus Traditional Programming

Traditional programming is a process of hand - meaning that the programmer makes the program. But apart from anyone programming the logic, one has to manually decide the rules or code. We have input data, and the programmer wrote the program/rules that use that data and execute on a computer to generate the output/ answer as shown in the following *Figure 1.3*:

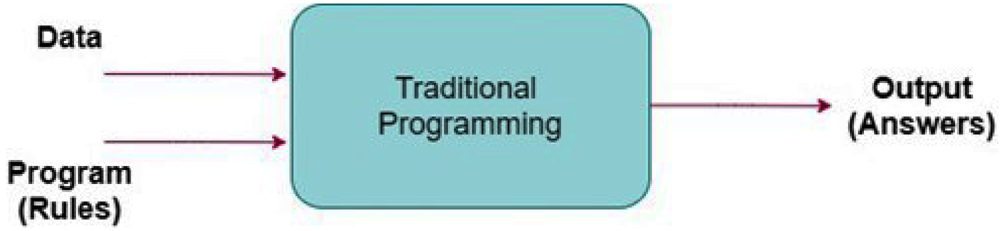


***Figure 1.3**: Traditional Programming*

On the other hand, in case of Machine Learning, data and output/answers (or labels) come in as input and the learning rules (models) come out as output as shown in *Figure 1.4*. The machine learning paradigm is especially important because it allows the computer to learn new rules in a complex and advanced environment, a space difficult to understand by humans.

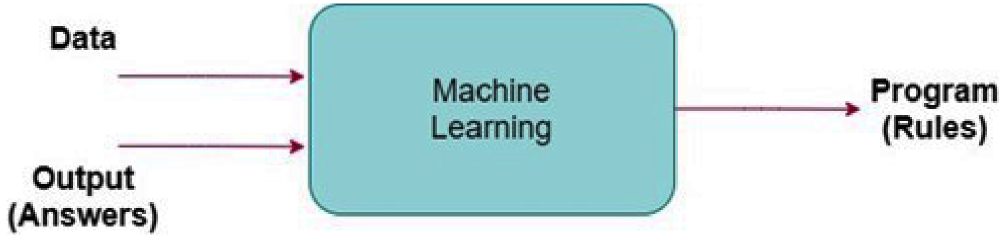


***Figure 1.4**: Machine Learning*

For example, we could write a traditional computer program for activity recognition (walking, running, or cycling) based on a person's speed (data) and an activity description (walk, run, and cycling) based on speed (rules). However, the problem with this method is that different people walk, run, and ride bikes at different speeds depending on age, environment, health, and so on.

If we have to solve the same problem in the field of machine learning, we can find many examples of tasks and their labels (answers i.e., type of activity) and learn or infer the rules for predicting future work.

The Seven Steps of Machine Learning

The process of machine learning can be broken down into 7 steps as shown in *Figure 1.5*. To illustrate the significance and function of each step, we would be using an example of a simple model. This model would be responsible for differentiating between an apple and an orange. Machine learning is capable of much for complex tasks. However, to explain the process in simplistic terms, a basic example is taken to explain the relevant concepts:

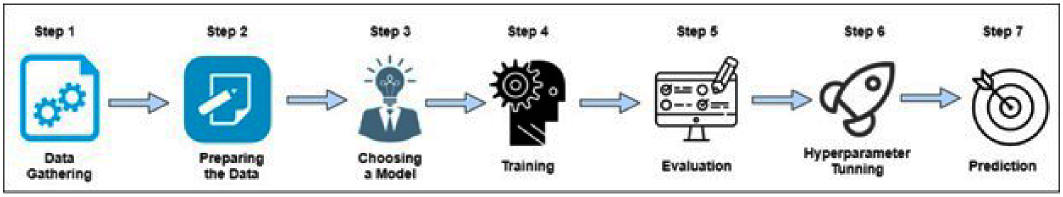


Figure 1.5: *Steps of Machine Learning*

Step 1: Data Gathering / Data Collection

This step is very important because the quality and quantity of data that we gather will directly determine how well or badly your model will work. To develop our machine learning model, our first step would be to gather relevant data that can be used to train the model.

The step of gathering data is the foundation of the machine learning process. Mistakes such as choosing the incorrect features or focusing on limited types of entries for the data set may render the model completely ineffective.

Step 2: Preparing the data

After the training data is gathered, we move on to the next step of machine learning: data preparation, where the data is loaded into a suitable place, and then prepared for use in machine learning training. Here, the data is first put all together, and then the order is randomized as the order of data should not affect what is learned.

In this step, we wrangle the data collected in Step 1 and prepare it for training. We can clean the data by removing duplicates, correct errors, deal with missing values, data type conversions, and so on. We can also do the visualization of the data, as this will help us to see if there are any relevant relationships between the different attributes, how we can take their advantage, and as well as, a show if there are any data imbalances present.

Another major component of data preparation is breaking down the data sets into 2 parts. The larger part (~80%) would be used for training the model while the smaller part (~20%) is used for the evaluation of the trained model's performance. This is important because using the same data sets for both training and evaluation would not give a fair assessment of the model's performance in real-world scenarios.

Step 3: Choosing a Model

The selection of the model type is our next course of action once we are done with the data-centric steps. There are various existing models developed by data scientists that can be used for different purposes. Different classes of models are good at modeling the underlying patterns of different types of datasets. These models are designed with different goals in mind. For instance, some models are more suited to dealing with texts while another model may be better equipped to handle images.

Step 4: Training

At the heart of the machine learning process is the training of the model. The bulk of the "*learning*" is done at this stage. Training requires patience and experimentation. It is also useful to know the field where the model would be implemented.

Training can prove to be highly rewarding if the model starts to succeed in its role. The training process involves initializing some random values for say X and Y of our model, predict the output with those values, then compare it with the model's prediction and then adjust the values so that they match the predictions that were made previously.

This process then repeats, and each cycle of updating is called one training step.

It is comparable to when a child learns to ride a bicycle. Initially, they may have multiple falls but, after a while, they develop a better grasp of the process and can react better to different situations while riding the bicycle.

Step 5: Evaluation

With the model trained, it needs to be tested to see if it would operate well in real-world situations. That is why the part of the data set created for evaluation is used to check the model's proficiency. This puts the model in a scenario where it encounters situations that were not a part of its training.

Evaluation becomes highly important when it comes to commercial applications. Evaluation allows data scientists to check whether the goals they set out to achieve were met or not. If the results are not satisfactory then the prior steps need to be revisited so that the root cause behind the model's underperformance can be identified and, subsequently, rectified. If the evaluation is not done properly then the model may not excel at fulfilling its desired commercial purpose. This could mean that the company that designed and sold the model may lose their goodwill with the client. It could also mean damage to the company's reputation as future clients may become hesitant when it comes to trusting the company's acumen regarding machine learning models. Therefore, evaluation of the model is essential for avoiding the aforementioned ill-effects.

Step 6: Hyperparameter Tuning

Once the evaluation is over, any further improvement in our training can be possible by tuning the parameters. There were a few parameters that were implicitly assumed when the training was done. Another parameter included is the learning rate that defines how far the line is shifted during each step, based on the information from the previous training step. These values play a role in the accuracy of the training model and how long the training will take.

Naturally, the question arises that why we need hyperparameter tuning in the first place when our model is achieving its targets? This can be answered by looking at the competitive nature of machine learning-based service providers. Clients can choose from multiple options when they seek a machine learning model to solve their respective problems. However, they are more likely to be enticed by the one which produces the most accurate results. That is why for ensuring the commercial success of a machine learning model, hyperparameter tuning is a necessary step.

Step 7: Prediction

The final step of the machine learning process is prediction. This is the stage where we consider the model to be ready for practical applications.

This is the point where the value of machine learning is realized. Here we can finally use our model to predict the outcome of what we want.

Applications of Machine Learning

Machine learning helps to improve business decisions, boost productivity, detect diseases, forecast the weather, and much more. A machine learns automatically from the inputs. Some of the best machine learning applications are mentioned as follows.

Social Media Features

Social media platforms like Facebook, Instagram, LinkedIn, and so on, use machine learning algorithms for users. Social media create some attractive and excellent features using machine learning. For example, Facebook notices and records the activities, chats, likes, comments, and the time spent by users on specific kinds of posts, videos, and so on. Machine learning learns from the user's activities & experiences and makes friends and page suggestions for the user's profile.

Let us take another example, when a user uploads a picture of him with a friend and Facebook instantly recognizes that friend. Facebook checks the poses and projections in the picture, notice the unique features, and then match them with the people in the user's friend list and tag that friend automatically.

Product Recommendations

Product recommendation is used in almost every e-commerce website today. This is a very advanced application of machine learning techniques. Using machine learning, e-commerce websites track the user's behavior based on previous purchases, searching patterns, cart history, and so on. Based on this tracking, the e-commerce websites recommend the product to users that somehow matches the user's taste.

Image & Speech Recognition

Image recognition is one of the most common applications of machine learning. Image processing is used to identify objects, persons, places, digital images, and so on in an image. This technique is used for further analysis, such as pattern recognition, character recognition, face detection, or face recognition. *Facebook* provides us with a feature of auto friend tagging suggestions. Whenever a user uploads a photo with his Facebook friends, then the user automatically gets a tagging suggestion with the name of his friends.

Speech recognition is the translation of spoken words into text. It is also known as "*Speech to text*", or "*Computer speech recognition*". *Google Assistant*, *Siri*, *Cortana*, and *Alexa* are using speech recognition technology to follow voice instructions.

Sentiment Analysis

Sentiment analysis is a real-time machine learning application that determines the emotion or opinion of the speaker or the writer. For instance, if someone has written a review or email (or any form of a document), the sentiment analyzer will instantly find out the actual thought and tone of the text. This sentiment analysis application can be used to analyze a review-based website, decision-making applications, and so on.

Self-driving cars

One of the most exciting applications of machine learning is self-driving cars. Machine learning plays a significant role in self-driving cars. Tesla, the most popular car manufacturing company is working on a self-driving car. These self-driven cars are autonomous cars that are safer than cars driven by humans. Things that make these cars safe are that they are not affected by factors like illness or the emotion of the driver.

Email Spam and Malware Filtering

Whenever we receive a new email, it is filtered automatically as important, normal, and spam. We always receive an important email in our inbox with the important symbol and spam emails in our spam box, and the technology behind this is machine learning. Following are some spam filters used by *Gmail*:

- Content filter

- Header filter
- General blacklists filter
- Rules-based filters
- Permission filters

Stock Market Trading

Machine learning is widely used in stock market trading. In the stock market, there is always a risk of fluctuations in shares, so for this, machine learning is used to predict the market trends of stock.

Medical Diagnosis

In medical science, machine learning is used for disease diagnoses. With this, medical technology is growing rapidly and can build 3D models that can predict the exact position of lesions in the brain.

It helps in finding brain tumors and other brain-related diseases easily.

Online Fraud Detection

Machine learning is making online transactions safe and secure by detecting fraud transactions. Whenever we perform some online transactions, there may be various ways that a fraudulent transaction can take place such as fake accounts, fake ids, and steal money in the middle of a transaction. Therefore, to detect this, machine learning helps us by checking whether it is a genuine transaction or a fraud transaction.

Automatic language translation

Nowadays, if we visit a new place and are not aware of the language then it is not a problem, as for this machine learning helps us by converting the text into our known language. Google's **Google Neural Machine Translation** (**GNMT**) provides us with this feature, which is a neural machine learning that translates the text into a familiar language, and it is called **automatic translation**.

The technology behind the automatic translation is a sequence-to-sequence learning algorithm, which is used with image recognition and translates the text from one language to another.

Types of Machine Learning

There are different ways in how a machine learns. In some cases, machines are trained and in some cases, machines learn on their own. Well, primarily, there are three types of machine learning – *Supervised Learning*, *Unsupervised Learning & Reinforcement Learning* as shown in *Figure 1.6*. In this part, we will to discuss the types of machine learning in detail:

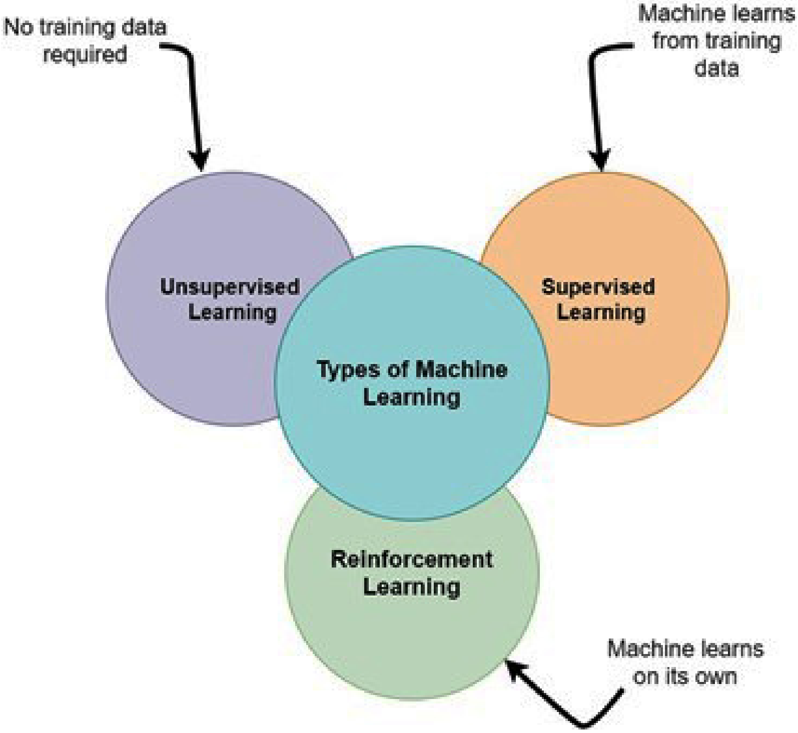

No training data required
Machine learns from training data
Unsupervised Learning
Supervised Learning
Types of Machine Learning
Reinforcement Learning
Machine learns on its own

***Figure 1.6**: Types of Machine Learning*

Supervised Learning

Supervised machine learning algorithms are designed to learn by example. It is called supervised learning because the process of an algorithm learning from the training dataset can be thought of as a teacher supervising the learning process.

In supervised learning, the data consists of an input variable and an output variable or the dataset is labeled. Labeled data means the input data and its associated output is available in the dataset.

During the training process, the algorithm search for the patterns in the input data and correlate with the desired output data. After the training process, a supervised learning algorithm will take the unseen data as inputs and will determine which label the new inputs will be classified as based on prior training data. The objective of a supervised learning model is to predict the correct label for the newly presented input data.

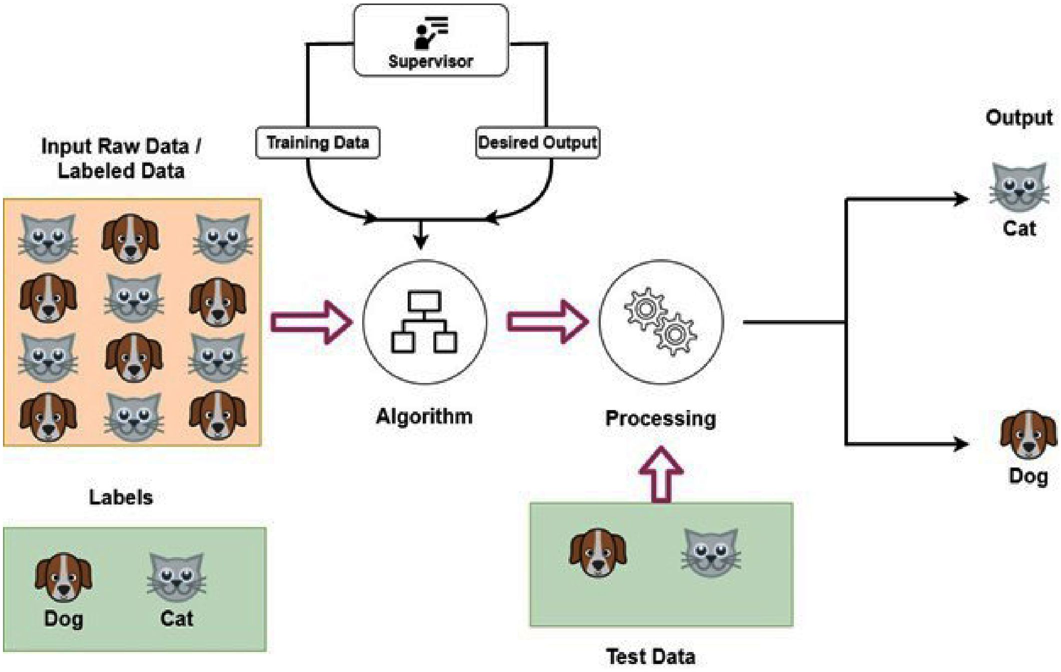


Figure 1.7: Supervised Learning

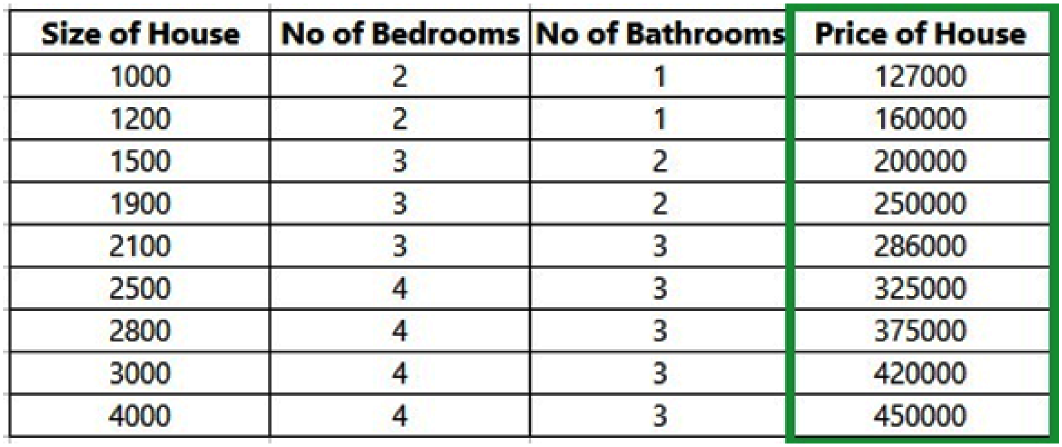
As shown in *Figure 1.7*, we have a dataset of images of cats and dogs. Every image in the dataset is labeled means we know which image represents a cat or which image represents a dog. We give all the images and their corresponding labels to the model/algorithm. Then, it should be able to make the difference on its own.

If everything goes very well, when showing a new image to the model without having its label, it should be able to tell us whether it is a cat or a dog (if the model learned well).

Supervised learning problems can be further classified into regression and classification problems.

Regression

Regression algorithms are used if there is a relationship between input variable and the output variable. It is used when the value of the output variable is continuous or real, such as house price, weather forecasting, stock price prediction, and so on.



Size of House	No of Bedrooms	No of Bathrooms	Price of House
1000	2	1	127000
1200	2	1	160000
1500	3	2	200000
1900	3	2	250000
2100	3	3	286000
2500	4	3	325000
2800	4	3	375000
3000	4	3	420000
4000	4	3	450000

***Figure 1.8**: Regression*

Figure 1.8 shows, the dataset which serves the purpose of predicting the house price, based on different parameters. Here, the input variables are *Size of House*, *No of Bedrooms* & *No of Bathrooms* & the output variable is the *Price of House*, which is a continuous value. Therefore, this is a Regression Problem.

The goal here is to predict a value as much closer to the actual output value as the model can and then evaluation is done by calculating the error value. The smaller the error the greater the accuracy of the regression model.

Classification

Classification is the process to group the output into different classes based on one or more input variables. Classification is used when the value of the output variable is discrete or categorical, such as email is "*spam*" or "*not spam*" or "*disease*" and "*no disease*" or "*rain*" and "*not rain*" or "*Yes*" or "*No*" and 0 or 1, and so on.

If the algorithm tries to classify input variables into two different classes, it is called **binary classification**, such as email is "*spam*" or "*not spam*". When the algorithm tries to classify input variables into more than two classes, it is called **multiclass classification**, such as *handwritten character recognition* where classes go from 0 to 9.

Unsupervised Learning

Unsupervised learning deals with unlabeled data means here we have input data and no corresponding output variable. This is the opposite of supervised machine learning. In unsupervised learning, the users do not need to teach/supervise the model. There is no correct output and no supervisor to teach.

The algorithm itself learns from the input data & discovers the patterns & information from the data to learn & group the data according to similarities.

As shown in the *Figure 1.9*, suppose we have an image dataset of different types of cats and dogs. The unsupervised learning algorithm is given that dataset. The algorithm is never seen/trained upon the given dataset, which means it does not have any idea about the features of the dataset. The task of the unsupervised learning algorithm is to identify the image features on their own. Unsupervised learning algorithms will perform this task by clustering the image dataset into groups according to the similarities between the images.

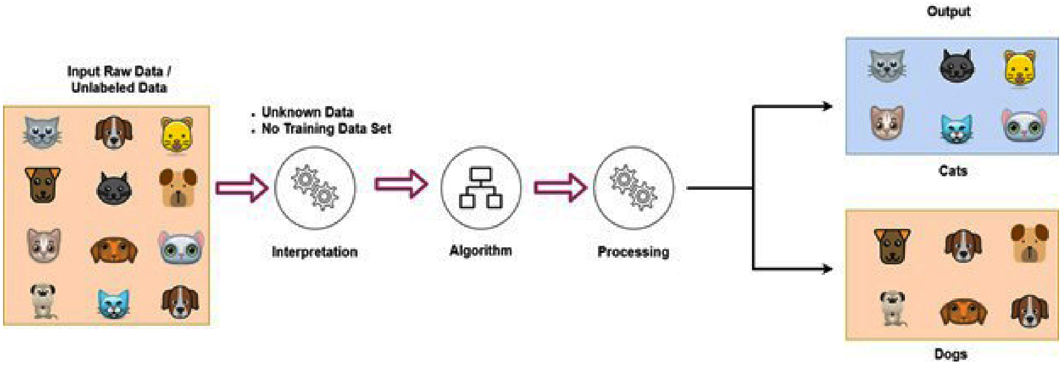


***Figure 1.9**: Unsupervised Learning*

The unsupervised learning algorithm can be further categorized into clustering and Association problems.

Clustering

Clustering is an important concept of unsupervised learning. It is used to find out the hidden structures or patterns in uncategorized data. The clustering algorithm process the uncategorized data and divides them into different clusters (groups) such that objects with many similarities remain in the same group and have fewer or no similarities with the objects of another group. As shown in *Figure 1.10*, it makes the two clusters of cats and dogs.

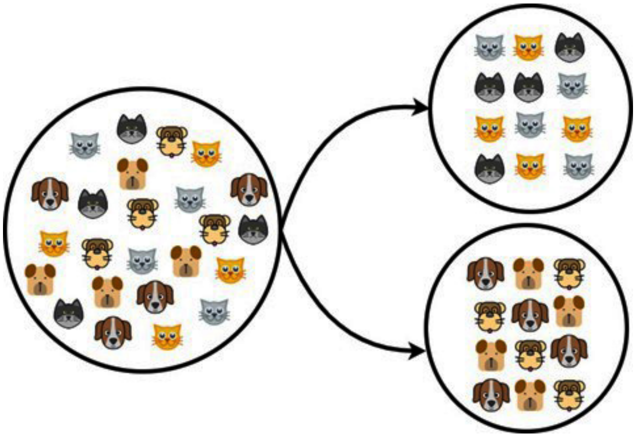


***Figure 1.10:** Clustering*

Association

An association rule is an unsupervised learning method. It is used to find out the relationships between the data in the large dataset. It determines the set of items that occurs together in the dataset. Association rule makes marketing strategy more effective. For example, suppose a person who buys bread also tends to buy butter/ jam as shown in *Figure 1.11*. A typical example of the association rule is Market Basket Analysis.

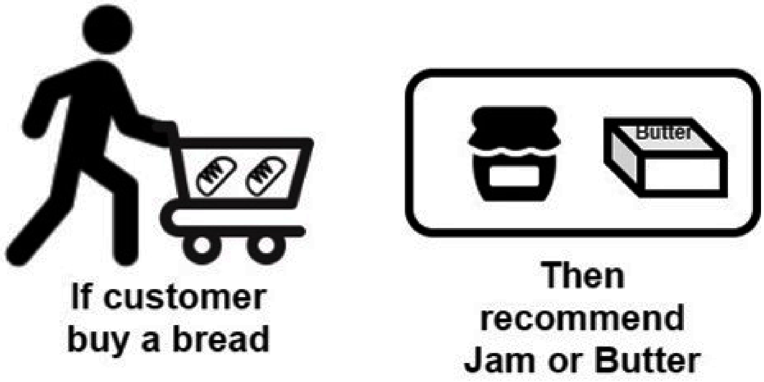


***Figure 1.11**: Association*

Reinforcement Learning

Reinforcement learning is the third and last type of Machine Learning. This is feedback-based Machine Learning. In Reinforcement Learning, the machine or agent atomically learns using feedback without any labeled data. Here, the agent learns itself from its experience.

In Reinforcement Learning, the agent uses the hit and trial process. If the agent tries a successful step, then the agent gets a reward, otherwise for each mistake agent gets a penalty. Its goal is to maximize the total reward.

As shown in *Figure 1.12*, consider the example of teaching a dog. Since the dog does not understand any human language, we cannot tell him exactly what to do. We create a situation, and the dog tries to respond in many different ways. If the dog's responds in a desirable way, we will give him bread. Now, whenever the dog is presented in the same situation again, the dog does the same action by eagerly anticipating a greater reward (bread). That's like learning that a dog gets from "*what* *to do"* from a good experience. At the same time, the dog is learning to do nothing in the face of negative experiences.

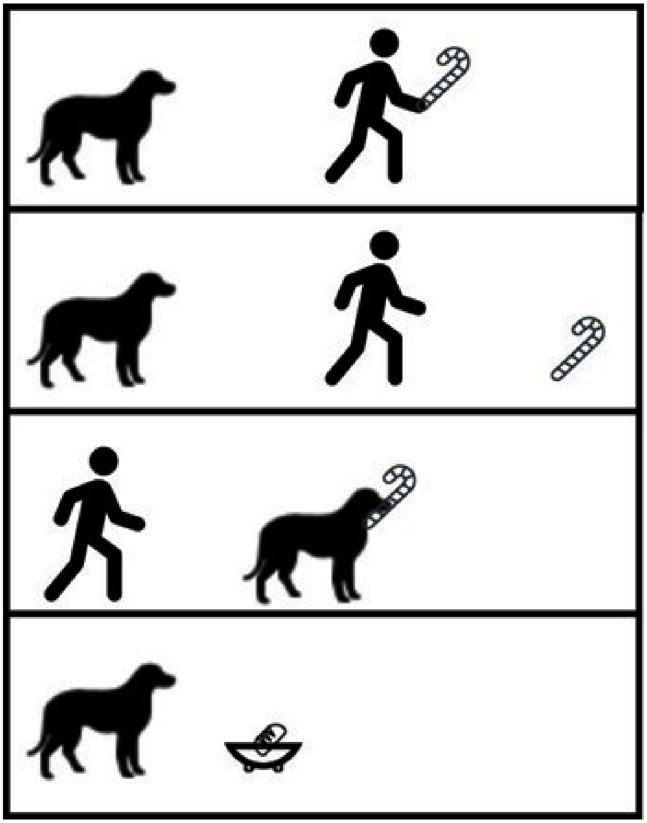


***Figure 1.12**: Reinforcement Learning*

Advantages of Machine Learning

There are many advantages of machine learning. These are some of the helpful advantages:

- **Easily identifies trends and patterns**: Machine learning takes large amounts of data and discovers the hidden structures, specific styles and patterns that are very difficult to find by humans. For example, an e-commerce website like Flipkart, works to understand the browsing methods and purchase records of its users to help target relevant products, deals, and related reminders. It uses results to show relevant ads or products to them.
- **No human intervention needed or Automation of Everything**: Machine learning has a very powerful tool to automate various decision-making tasks. By automating things, we allow the algorithm to perform difficult

tasks. Automation is now practiced almost everywhere. The reason is that it is very reliable. In addition, it helps us to think deeply.

- **Efficient Handling of any type of Data:** Here are many factors that make machine learning reliable. Data handling is one of them. Machine learning plays a big role when it comes to data. It can handle any type and amount of data. It can process and analyze the data that normal systems cannot. Data is the most important part of a machine learning model.
- **Continuous Improvement:** As machine learning algorithms gain experience from data, they keep improving their accuracy and efficiency. This lets them make better decisions. Suppose, we need to make a weather forecast model. As the amount of data we have keeps growing, our algorithms learn to make accurate predictions faster.

Disadvantages of Machine Learning

Similar to the advantages of machine learning, we should also know the disadvantages of machine learning. If we don't know the cons, we won't know the risks of ML. So, let's have a look at the disadvantages.

- **Possibility of High Error:** In machine learning, we can choose algorithms based on the accurate results. Therefore, we must apply the results to all algorithms. A major problem arises with the training and testing of data. The data is large, so sometimes deleting errors becomes impossible. These errors can cause headaches for users. As the data is large, errors take a lot of time to resolve.
- **Data Acquisition**: Machine learning requires a large number of datasets for the training and testing of the model. The machine learning algorithm takes the good quality of data for an accurate result. In many situations, the data constantly keeps on updating. Therefore, we have to wait for the new data to arrive.
- **Time and Resources**: Machine learning algorithms require enough time to learn and develop enough to achieve their goals with a high degree of accuracy and consistency. It also requires great resources to operate. This may mean additional computational power requirements for our computer.
- **Algorithm Selection:** A machine learning problem can implement various algorithms to find a solution. It is a manual and tedious task to run models with different algorithms and identify the most accurate algorithm based on the results.

Most Popular Machine Learning Software Tools

Machine learning software is available in the market. Following are the most popular ones among them. *Figure 1.13* shows the popular machine learning software:

***Figure 1.13:** Popular Machine Learning Software*

The machine learning software marketplace has several applications available. Here is the most popular software among them, enlisted in *table 1.1:*

Tool	Platform	Cost	Written in language	Algorithms or Features
Scikit Learn	Linux, Mac OS, Windows	Free.	Python, Cython, C, C++	Classification Regression Clustering Pre-processing Model Selection Dimensionality reduction.
PyTorch	Linux, Mac OS, Windows	Free	Python, C++, CUDA	Autograd Module Optim Module nn Module
TensorFlow	Linux, Mac OS, Windows	Free	Python, C++, CUDA	Provides a library for dataflow programming.

Weka	Linux, Mac OS, Windows	Free	Java	Data preparation Classification Regression Clustering Visualization Association rules mining
KNIME	Linux, Mac OS, Windows	Free	Java	Can work with large data volume. Supports text mining & image mining through plugins
Colab	Cloud Service	Free	-	Supports libraries of PyTorch, Keras, TensorFlow, and OpenCV
Apache Mahout	Cross-platform	Free	Java Scala	Preprocessors Regression Clustering Recommenders Distributed Linear Algebra.
Accors.Net	Cross-platform	Free	C#	Classification Regression Distribution Clustering Hypothesis Tests & Kernel Methods Image, Audio & Signal. & Vision
Shogun	Windows Linux UNIX Mac OS	Free	C++	Regression Classification Clustering Support vector machines. Dimensionality reduction Online learning and so on.

Keras.io	Cross-platform	Free	Python	API for neural networks
Rapid Miner	Cross-platform	Free plan Small: $2500 per year. Medium: $5000 per year. Large: $10000 per year.	Rapid Miner	Cross-platform

***Table 1.1**: Popular Machine Learning Software Tools*

*(Source: **https://www.softwaretestinghelp.com/machine-learning-tools/**)*

Summary

- Machine learning is the sub-domain of **artificial intelligence** (AI).
- The goal of machine learning is usually to understand the structure of the data and to match that data to models that can be understood and used by humans.
- Arthur Samuel stated that "*Machine Learning gives computers the ability to learn without being explicitly programmed*".
- *Tom Mitchell* gave a "*well-established*" mathematical and relational definition that "*A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E*".
- Traditional programming is a process of hand - meaning that the programmer makes the program.
- In machine learning, data and output come in as input and the learning rules come out as output.
- The seven steps of machine learning:
  - Gathering of data/data collection
  - Preparing that data
  - Choosing a model
  - Training
  - Evaluation
  - Hyperparameter tuning

  - o Prediction

- Applications of machine learning
  - o Social media features
  - o Product recommendations
  - o Image & speech recognition
  - o Sentiment analysis
  - o Self-driving cars
  - o Email spam and malware filtering
  - o Stock market trading
  - o Medical diagnosis
  - o Online fraud detection
  - o Automatic language translation
- **Supervised Learning**: The data consists of an input variable and an output variable or the dataset is labeled. Supervised Learning is further classified into Regression and Classification.
- **Regression algorithms** are used if there is a relationship between the input variable and the output variable. It is used when the value of the output variable is continuous or real.
- **Classification** is the process of grouping the output into different classes based on one or more input variables. Classification is used when the value of the output variable is discrete or categorical
- **Unsupervised Learning**: Unsupervised learning deals with unlabeled data means here we have input data and no corresponding output variable. This is further classified into Clustering and Association.
- **Clustering** is used to find the hidden structure or pattern in uncategorized data. The clustering algorithm process the uncategorized data and divide them into different clusters (groups) based on similarities.
- **Association** is used to find the relationships between data in a large dataset. It determines the set of items that occur together in the dataset.
- **Reinforcement Learning**: This is feedback-based machine learning. In reinforcement learning, the machine or agent atomically learns using feedback without any labeled data. Here, the agent learns itself from its experience.
- Advantages of machine learning:
  - o Easily identifies trends and patterns

  - o No human intervention needed or automation of everything
  - o Efficient handling of any type of data
  - o Continuous improvement

- Disadvantages of machine learning:
  - o Possibility of high error
  - o Data acquisition
  - o Time and resources
  - o Algorithm selection

Exercise (MCQs)

Tick the correct option

1. **What is true about machine learning?**

   a) Machine learning (ML) is a field of computer science

   b) ML is a type of artificial intelligence that extracts patterns out of raw data by using an algorithm or method.

   c) The main focus of ML is to allow computer systems to learn from their experience without being explicitly programmed or involving human intervention.

   d) All of the above

2. **Machine learning is a field of AI consisting of learning algorithms that**

   a) Improve their performance

   b) At executing some task

   c) Overtime with experience

   d) All of the above

3. **Choose the option that is correct regarding machine learning & artificial intelligence.**

   a) Machine learning is an alternate way of programming intelligent machines

   b) AI & ML have very different goals

   c) Machine learning is a set of techniques that turn a dataset into software

   d) AI is software that can emulate the human mind

      i. I, II, IV

      ii. I, III, IV

      iii. II, III, IV

      iv. All are correct

4. **The problem of finding hidden structures in unlabeled data is called**
   - *a)* Supervised learning
   - *b)* Unsupervised learning
   - *c)* Reinforcement learning
   - *d)* None of the above

5. **The task of inferring a model from labeled training data is called**
   - *a)* Unsupervised learning
   - *b)* Supervised learning
   - *c)* Reinforcement learning
   - *d)* None of the above

6. **An e-commerce company wants to segment their customers into distinct groups to send appropriate offers, this is an example of**
   - *a)* Unsupervised learning
   - *b)* Supervised learning
   - *c)* Reinforcement learning
   - *d)* None of the above

7. **Which of the following sentence is false regarding regression?**
   - *a)* It relates inputs to outputs.
   - *b)* It is used for prediction.
   - *c)* It may be used for interpretation.
   - *d)* It discovers causal relationships.

8. **Which of the following is a regression task?**
   - *a)* Predicting the monthly sales of a cloth store in rupees.
   - *b)* Predicting if a user would like to listen to a newly released song or not based on historical data.
   - *c)* Predicting the confirmation probability (in fraction) of your train ticket whose current status is waiting list based on historical data
   - *d)* Predicting if a patient has diabetes or not based on historical medical records.
     - *i.* I, IV
     - *ii.* I, II, IV
     - *iii.* I, III
     - *iv.* II, III, IV

9. **Which of the following is an unsupervised task?**
   a) Grouping images of footwear and caps separately for a given set of images
   b) Learning to play chess
   c) Predicting if an edible item is sweet or spicy based on the information of the ingredients and their quantities.
   d) all of the above

10. **In which of the following type of learning the teacher returns reward and punishment to the learner?**
   a) Active learning
   b) Reinforcement learning
   c) Supervised learning
   d) Unsupervised learning

11. **Which of the following are classification tasks?**
   a) Find the gender of a person by analyzing his writing style.
   b) Predict the price of a house based on the floor area, the number of rooms, and so on.
   c) Predict whether there will be abnormally heavy rainfall next year.
   d) Predict the number of copies of a book that will be sold this month.
      i. I, II
      ii. II, III, IV
      iii. I, III
      iv. I, III, IV

12. **I am a marketing consultant for a leading e-commerce website. I have been given the task of making a system that recommends products to users based on their activity on Facebook. I realize that user interests could be highly variable. Hence, I decide to**
   a) First, cluster the users into communities of like-minded people and
   b) Second, train separate models for each community to predict which product category (e.g., electronic gadgets, cosmetics, and so on.) would be the most relevant to that community.

   The first task is a/an _______________ learning problem while the second is a/an _________________ problem.
   i. Supervised and unsupervised
   ii. Unsupervised and supervised

iii. Supervised and supervised

iv. Unsupervised and unsupervised

13. Which of the following is a supervised learning problem?

a) Predicting the outcome of a cricket match as a win or loss based on historical data.

b) Recommending a movie to an existing user on a website like IMDb based on the search history (including other users)

c) Predicting the gender of a person from his/her image. You are given the data of 1 million images along with the gender

d) Given the class labels of old news articles, predicting the class of a new news article from its content. Class of a news article can be such as sports, politics, technology, and so on

i. I, II, III

ii. I, III, IV

iii. II, III, IV

iv. I, II, III, IV

14. Which of these are categorical features?

a) Height of a person

b) Price of petroleum

c) Mother tongue of a person

d) Amount of rainfall in a day

Answers

1. **d.** All of the above
2. **d.** All of the above
3. **b.** I, III, IV
4. **b.** Unsupervised learning
5. **b.** Supervised learning
6. **a.** Unsupervised learning
7. **d.** It discovers causal relationships.
8. **c.** I, III
9. **a.** Grouping images of footwear and caps separately for a given set of images
10. **b.** Reinforcement learning
11. **c.** I, III
12. **b.** Unsupervised and supervised

13. **d.** I, II, III, IV

14. **c.** Mother tongue of a person

Fill in the blanks

1. Predicting the price of a house based on the floor area, number of rooms and so on is an example of __________________machine learning.
2. Branch of an engineering student is a __________________feature.
3. __________________defines how far the line is shifted during each step, based on the information from the previous training step.
4. In the case of machine learning __________________come in as input and the __________________come out as output.
5. __________________processes the uncategorized data and divides them into different clusters.

Answers

1. Supervised

2. Categorical

3. Learning rate

4. data and output / answers (or labels), learning rules (models)

5. Clustering algorithm

Descriptive questions

1. What is machine learning? Explain the types of machine learning.
2. Explain the different steps used to perform machine learning.
3. Explain the difference between traditional programming & machine learning.
4. Differentiate between supervised, unsupervised & reinforcement learning.
5. Explain supervised learning with an example.
6. Explain unsupervised learning with an example.
7. What is the difference between regression & classification? Explain with example.
8. Explain the various applications of machine learning.

Chapter 2
Supervised Learning Algorithms

Introduction

Supervised machine learning turns data into real, actionable things. It is a type of machine learning which performs the task of implying a function from labelled training data. Each case is a pair of input data items and the desired output value. During training, the algorithm will identify the patterns in the dataset that correlates with the desired outputs. The aim of the supervised learning algorithm is to predict the correct label for the new input data.

This chapter explores different supervised machine learning algorithms with examples.

Structure

In this chapter, we will discuss the following topics:

- Introduction of supervised learning
- Types of supervised learning
- Naïve Bayes classifier
- Decision tree
- K-nearest

- Logistic regression
- Support vector machine
- Random forest algorithm

Objectives

By reading this chapter, you will be able to:

- Understand the concepts and types of supervised machine learning.
- Understand the regression and classifications.
- Understand the principle of Naive Bayes classifier algorithm.
- Understand the decision tree concept and its structure.
- Understand the **K-Nearest Neighbours** (**K-NN**) algorithm, **Support Vector Machine** (**SVM**) and Random Forest Algorithm.

Introducing Supervised Learning

Supervised machine learning algorithms are designed to learn by experience. It is called **supervised learning** because the process of an algorithm learning from the training dataset can be thought of as a teacher supervising the learning process.

In supervised learning, the data consists of an input variable and an output variable or the dataset is labeled. Labeled data means the input data and its associated output is available in the dataset. During the training process, the algorithm searches for the patterns in the input data and correlate them with the desired output data. After the training process, a supervised learning algorithm will take unseen data as input and will determine which label the new input will be classified as based on prior training data. The objective of a supervised learning model is to predict the correct label for the newly presented input data.

Suppose we have a dataset of images of cats and dogs as shown in *Figure 2.1*. Every image in the dataset is labelled means we know which image represents a cat or which image represents a dog. We give all the images and their corresponding labels to the model/algorithm. Then, it will be able to make a difference on its own.

If everything goes very well, when showing a new image to the model without having its label, it should be able to tell us whether it is a cat or a dog (if the model has learned well).

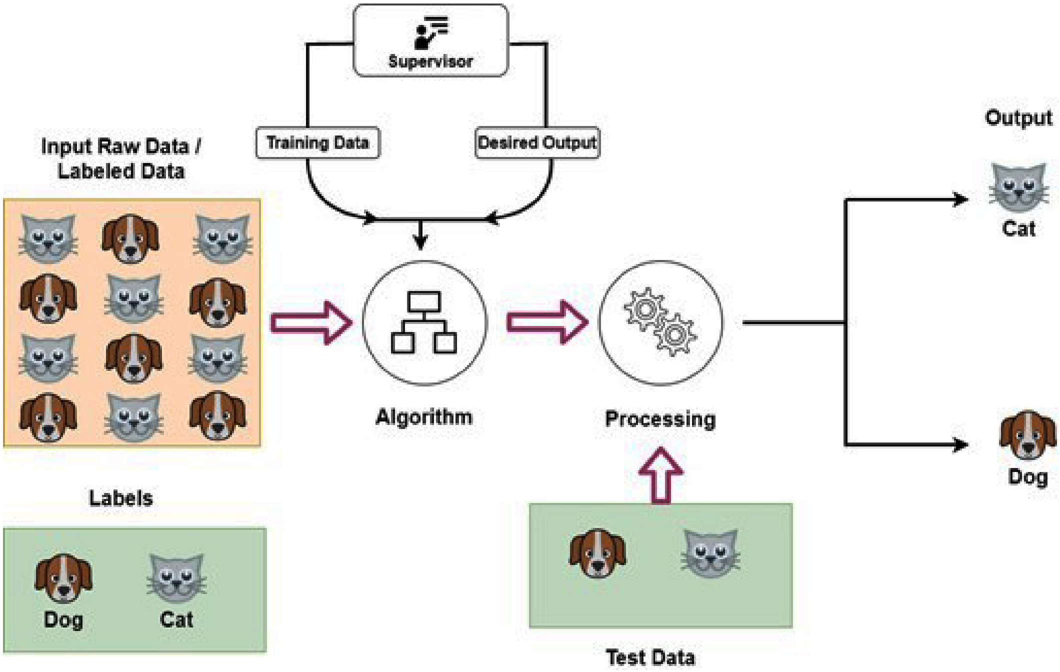


Figure 2.1: *Supervised Learning*

Types of Supervised Learning

Supervised Machine Learning can be further classified into regression and classification as shown in *Figure 2.2*:

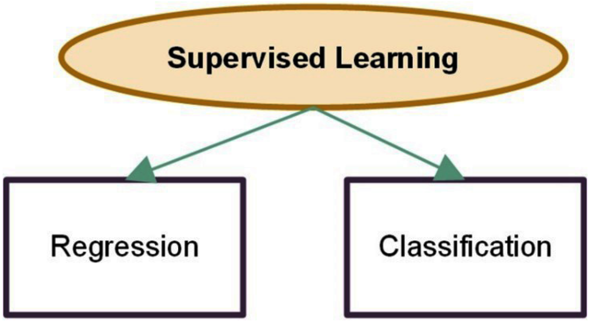


Figure 2.2: *Types of Supervised Learning*

Regression

Regression is a mathematical method used in finance, investment, and other methods that attempt to determine the strength and nature of the relationship between a single dependent/output variable and one or more other independent/input variables.

Regression algorithms are used if there is a relationship between one or more input variables and output variables. It is used when the value of the output variable is continuous or real, such as house price, weather forecasting, stock price prediction, and so on.

The following *Table 2.1* shows, the dataset, which serves the purpose of predicting the house price, based on different parameters:

Size of House	No. of Bedrooms	No. of Bathrooms	Price of House
1000	2	1	127000
1200	2	1	160000
1500	3	2	200000
1900	3	2	250000
2100	3	3	286000
2500	4	3	325000
2800	4	3	375000
3000	4	3	420000
4000	4	3	450000

***Table 2.1**: Dataset of Regression*

Here the input variables are *Size of House*, *No. of Bedrooms* and *No. of Bathrooms* & the output variable is the *Price of House*, which is a continuous value. Therefore, this is a *Regression Problem*.

The goal here is to predict a value as much closer to the actual output value and then evaluation is done by calculating the error value. The smaller the error the greater the accuracy of the regression model.

Regression is used in many real-life applications, such as financial forecasting (house price prediction or stock price prediction), weather forecasting, time series forecasting, and so on.

In regression, we plot a graph between the variables which best fits the given data points, using this plot, the machine learning model can make predictions about the data. In simple words, regression displays an entire line and curve in the goal predictor chart in order to minimize the vertical gap between the datapoints and the regression line. The distance between the data points and the line tells us whether a model has captured a strong relationship or not.

Terminologies used in Regression

Following are the terminologies used in regression:

- **Dependent Variable**: Dependent variable is also known as **target variable**, **output variable,** or **response variable**. The dependent variable in regression analysis is a variable, which we want to predict or understand. It is denoted by 'Y'.

  Example: Size of house, number of bedrooms, and number of bathrooms are dependent variables (refer to *Table 2.1*)

- **Independent Variable**: Independent variable is also known as the **input variable** or **predictor**. Independent variables affect the dependent variables, or these are used to predict the values of the dependent variables. It is denoted by 'X'.

  Example: Price of a house is an independent variable (refer to *Table 2.1*)

- **Outliers**: Outliers are observed data points that are far from the least square line or that differs significantly from other data or observations, or in other words, an outlier is an extreme value that differ greatly from other values in a set of values.

  In *Figure 2.3*, there are a bunch of apples, but one apple is different. This apple is what we call an **outlier**.

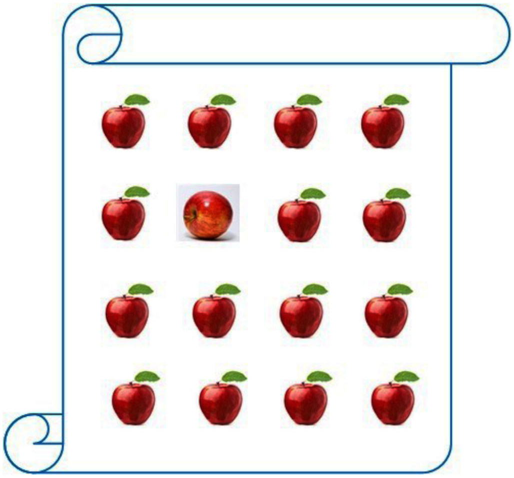
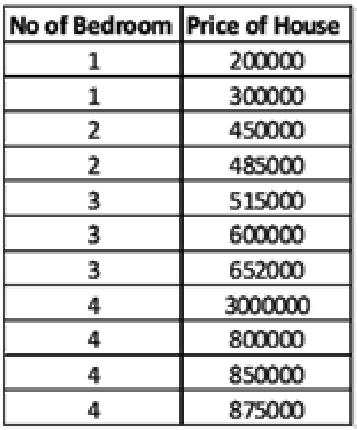


Figure 2.3: *Outlier*

Outliers need to be examined closely. Sometimes, for some reason or another, they should not be included in data analysis. It is possible that an outlier is a result of erroneous data. Other times, an outlier may hold valuable information about the population under study and should remain included in the data. The key is to examine carefully what causes a data point to be an outlier.



No of Bedroom	Price of House
1	200000
1	300000
2	450000
2	485000
3	515000
3	600000
3	652000
4	3000000
4	800000
4	850000
4	875000

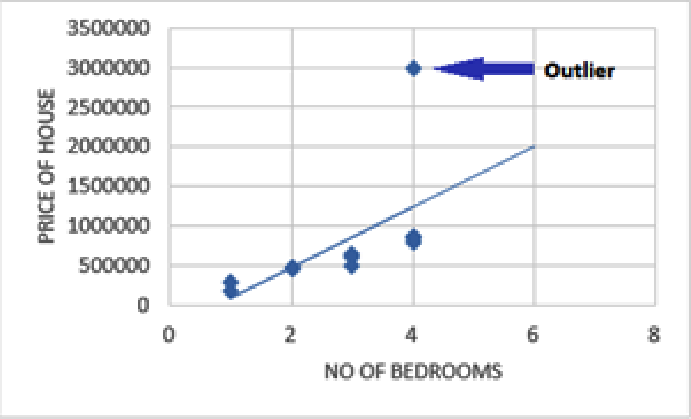


Figure 2.4: Outlier in House Price Dataset

Consider the following example. Suppose as shown in *Figure 2.4*, we sample the number of bedrooms in a house and note the price of each house. We can see from the dataset that ten houses range between 200000 and 875000; but one house is priced at 3000000, that house will be considered as an outlier.

- **Multicollinearity**: Multicollinearity in regression analysis occurs when two or more independent variables are closely related to each other, so as not to provide unique or independent data to the regression model. If the degree of correlation is high enough between variables, it can cause problems when fitting and interpreting the regression model.

  For example, suppose we do a regression analysis using variable height, shoe size, and hours spent in practice in a day to predict high jumps for basketball players. In this case, the height and size of the shoes may be closely related to each other because taller people tend to have larger shoe sizes. This means that multicollinearity may be a problem in this regression.

  Take another example, suppose we have two inputs X1 or X2 :

  $X1 = [0, 3, 4, 9 , 6, 2]$

  $X2 = [0, -1.5, -2, -4.5, -3, -1]$

  $X1 = -2 * X2$

  So X1 & X2 are collinear. Here it's better to use only one variable, either X1 or X2 for the input.

- **Underfitting and Overfitting**: Overfitting and underfitting are two primary issues that happen in machine learning and decrease the performance of the machine learning model.

  The main goal of each machine learning model is to provide a suitable output by adapting the given set of unknown inputs, this is known as **generalization**. It means after providing training on the dataset, it can produce reliable and accurate output. Hence, underfitting and overfitting are the two terms that need to be checked for the reliability of the machine learning model.

  Let us understand the basic terminology for overfitting and underfitting:

  - **Signal**: It is about the true base pattern of the data.
  - **Noise**: Unnecessary and insignificant data that reduces the performance.
  - **Bias**: It is the difference between expected and real values.
  - **Variance**: When model performs well on the training dataset but not on the test dataset, variance exists.

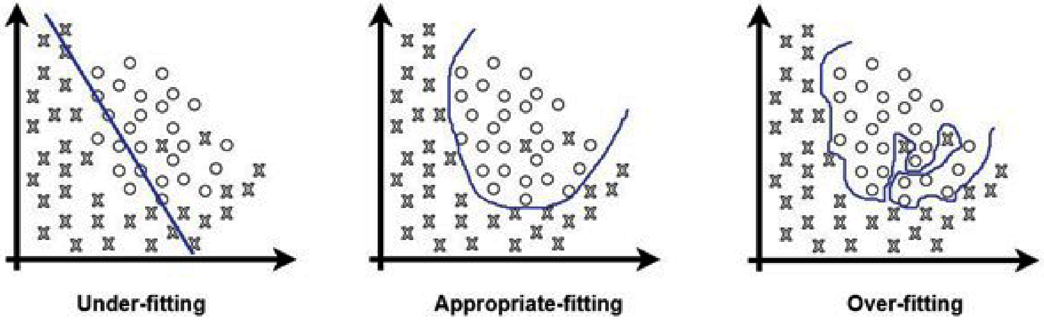


Figure 2.5: *Underfitting and Overfitting*

On the left side of the above *Figure 2.5*, anyone easily predicts that the line does not cover all the points shown in the graph. Such a model tends to cause a phenomenon known as **underfitting** of data. In the case of underfitting, the model cannot learn enough from the training data and from the training data and thus reduces precision and accuracy. There is a high bias and low variance in the underfitted model.

Contrarily, when we consider the right side of the graph in *Figure 2.5*, it shows that the predicted line covers all the points in the graph. In such a situation, we might think this is a good graph that covers all the points, but that is not true. The predicted line of the given graph covers all the points including those, which are noise and outlier. Such a model tends to cause a phenomenon known as **overfitting** of data. This model is responsible for predicting poor results due to its high complexity. The overfitted model has *low bias and high variance*. So, this model is also known as the **High Variance Model**.

Now, consider the middle graph in *Figure 2.5*, it shows a well-predicted line. It covers a major portion of the points in the graph while also maintaining the balance between bias and variance. Such a model tends to cause a phenomenon known as *appropriate fitting of data*.

Types of Linear Regression

As shown in *Figure 2.6*, linear regression is classified into two categories based on the number of independent variables:

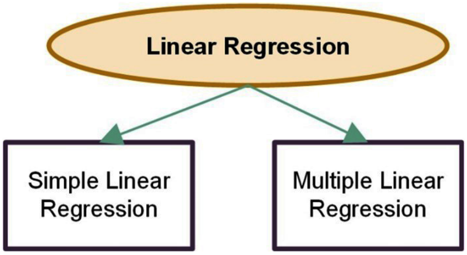


Figure 2.6: *Types of Linear Regression*

Simple Linear Regression

Simple Linear Regression is a type of linear regression where we have only one independent variable to predict the dependent variable. The dependent variable must be a continuous/real value.

The relationship between independent variable (X) & dependent variable (Y) is shown by a linear or a sloped straight line as shown in *Figure 2.7*, hence it is called Simple Linear Regression:

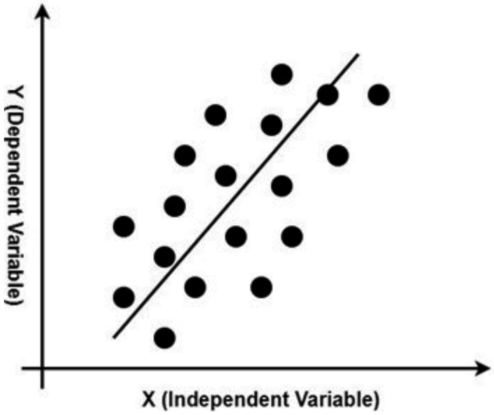


Figure 2.7: *Simple Linear Regression*

The Simple Linear Regression model can be represented using the following equation:

$$y= B_0+ B_1*x$$

Where,

Y: Dependent Variable

X: Independent variable

B_0 is the Y-intercept of the regression line where best-fitted line intercepts with the Y-axis.

B_1 is the slope of the regression line, which tells whether the line is increasing or decreasing.

Therefore, in this graph, the dots are our data and based on this data we will train our model to predict the results. The black line is the best-fitted line for the given data. The best-fitted line is a straight line that best represents the data on a scatter plot. The best-fitted line may pass through all of the points, some of the points or none of the points in the graph.

The goals of simple linear regression are as follows:

- To find out if there is a correlation between dependent & independent variables.
- To find the best-fit line for the dataset. The best-fit line is the one for which total prediction error (all data points) are as small as possible.
- How the dependent variable is changing, by changing the independent variable.

Suppose we have a dataset, which contains information about the relationship between '*Number of Bedrooms*' and '*House's Price*' as shown in *Table 2.2*.

No. of Bedrooms	House's Price
1	175000
2	550300
1	366000
3	575700
4	773500
1	325600
2	357600
3	680700

4	957000
1	275700
3	657500
2	425700
1	300700
1	245500
4	857000
2	475500
3	600000
4	775000

***Table 2.2**: House Dataset*

Here '*Number of Bedrooms*' is the independent variable (X) and '*Price House's*' is the dependent variable (Y). Our aim is to find out the value of B0 & B1 such that it produces the best-fitted regression line. This linear equation is then used for new data.

The *House* dataset is used to train our linear regression model. That is, if we give '*Number of Bedrooms*' as an input, our model should predict '*Price House's*' with minimum error.

Here Y' is the predicted value of '*House Price*'.

The values B_0 and B_1 must be chosen so that they minimize the error. If the sum of squared error is taken as a metric to evaluate the model, then the goal to obtain a line that best reduces the error is achieved.

$$\text{Error} = \sum_{i=1}^{n} (\text{Actual\_Output} - \text{Predicted\_Output})^2$$

$$\text{Error} = \sum_{i=1}^{n} (Y - Y')^2$$

If we do not square the error, then positive and negative points will cancel out each other.

Multiple Linear Regression

If there is more than one independent variable for the prediction of a dependent variable, then this type of linear regression is known as multiple linear regression.

Classification

Classification is the process to group the output into different classes based on one or more input variables. Classification is used when the value of the output variable is discrete or categorical, such as email is "*spam*" or "*not spam*" or "*disease*" and "*no disease*" or "*rain*" and "*not rain*" or "*Yes*" or "*No*" and 0 or 1, and so on.

If the algorithm tries to classify input variables into two different classes, it is called **binary classification**, such as email is "*spam*" or "*not spam*". When the algorithm tries to classify input variables into more than two classes, it is called **multiclass classification**, such as handwritten character recognition where classes go from 0 to 9. *Figure 2.8* shows the classification example with a message if an email is a spam or not spam:

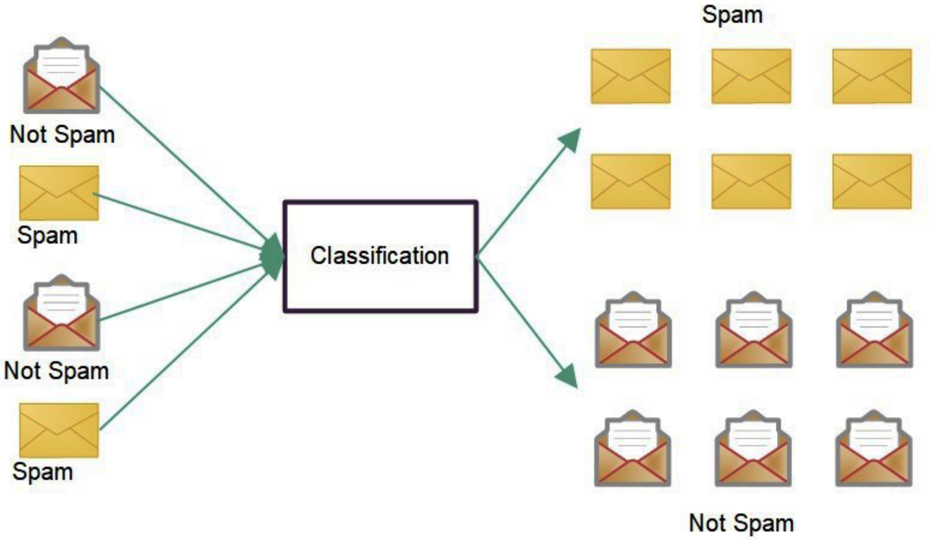


***Figure 2.8**: Example of Classification*

Naïve Bayes classifier algorithm

Naive Bayes classifiers are a group of classification algorithms that are based on Bayes' Theorem. It is a family of algorithms that shares a common principle, namely that every pair of the characteristics being classified is independent of each other.

- The Naïve Bayes algorithm is a supervised learning algorithm that solves classification problems and is based on the Bayes theorem.
- It is primarily used in text classification with a large training dataset.
- The Naïve Bayes Classifier is a simple and effective *Classification* algorithm that aids in the development of fast machine learning models capable of making quick predictions.

- It is a probabilistic classifier, which ensures it predicts based on an object's probability.
- Spam filtration, Sentimental analysis, and article classification are some popular applications of the Naïve Bayes Algorithm.

For example, if a fruit is red, round, and about 3 inches in diameter, it is classified as an apple. Even if these features are dependent on each other or on the presence of other features, all of these properties independently contribute to the likelihood that this fruit is an apple, which is why it is called '*Naive*.'

The Naive Bayes model is simple to construct and is especially useful for very large data sets. In addition to its simplicity, Naive Bayes has been shown to outperform even the most sophisticated classification methods.

Why is it called Naïve Bayes?

The Naive Bayes algorithm is made up of the words Naive and Bayes, which can be interpreted as:

Naïve: It is called Naïve because it assumes that the occurrence of one feature is unrelated to the occurrence of others. For example, if the fruit is identified based on color, shape, and taste, then a red, spherical, and sweet fruit is identified as an apple. As a result, each feature contributes to identifying it as an apple without relying on the others.

Bayes: It is known as Bayes because it is based on the principle of Bayes' Theorem.

Principle of Naive Bayes Classifier

A Naive Bayes classifier is a type of probabilistic machine learning model that is used to perform classification tasks. The classifier's crux is based on the Bayes theorem.

The Bayes theorem allows us to calculate the posterior probability P(c|x) from P(c), P(x), and P(x|c). Consider the following equation:

$$P(c|x) = \frac{P(x|c)\ \ P(c)}{(P(x)}$$

$$P(c|X)= P(x_1|c) * P(x_2|c) *.....* \ P(x_n|c)\ \ X\ P(C)$$

Where,

Posterior probability of class c (target) given predictor x(attributes) - *P(c|x)*

The prior probability of class - *P(c)*

The likelihood which is the probability of predictor given class - *P(x|c)*

The prior probability of predictor - *P(x)*

Working of Naïve Bayes' Classifier

The following steps demonstrate how the Nave Bayes' Classifier works.

Step 1: Convert the given dataset into frequency tables.

Step 2: Create a likelihood table by calculating the probabilities of the given features.

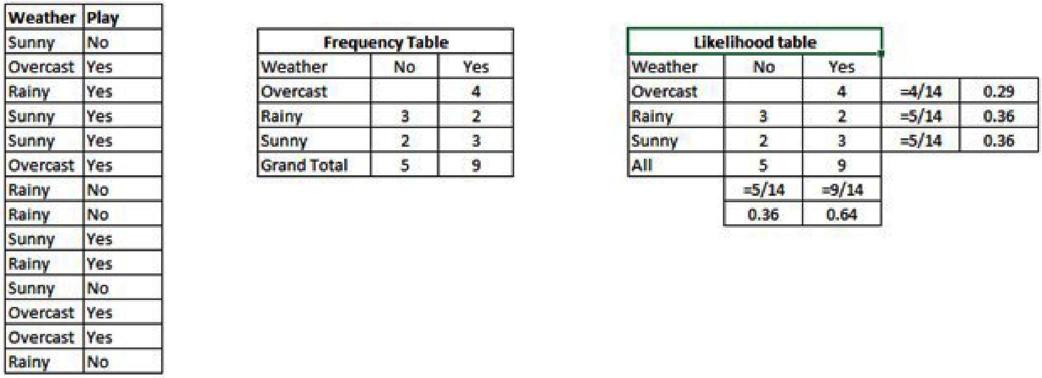
Step 3: Use Bayes theorem to calculate the posterior probability.

Naive Bayes Example

The weather training data set and corresponding target variable '*Play*' are shown in *Figure 2.9*. We must now categorize whether or not players will play based on the weather.

Step 1: Make a frequency table out of the data set.

Step 2: Make a likelihood table by calculating probabilities such as overcast probability and probability of playing.



Weather	Play
Sunny	No
Overcast	Yes
Rainy	Yes
Sunny	Yes
Sunny	Yes
Overcast	Yes
Rainy	No
Rainy	No
Sunny	Yes
Rainy	Yes
Sunny	No
Overcast	Yes
Overcast	Yes
Rainy	No

Frequency Table		
Weather	No	Yes
Overcast		4
Rainy	3	2
Sunny	2	3
Grand Total	5	9

Likelihood table				
Weather	No	Yes		
Overcast		4	=4/14	0.29
Rainy	3	2	=5/14	0.36
Sunny	2	3	=5/14	0.36
All	5	9		
	=5/14	=9/14		
	0.36	0.64		

***Figure 2.9**: Weather training data set and corresponding frequency and likelihood table*

Step 3: Calculate the posterior probability for each class using the Naive Bayesian equation. The outcome of prediction is the class with the highest posterior probability.

Naive Bayes employs a similar method to predict the likelihood of various classes based on various attributes. This algorithm is commonly used in text classification and multi-class problems.

Types of Naïve Bayes

The Naive Bayes Model is classified into three types, which are listed as follows:

- **Gaussian Naïve Model**: The Gaussian model is based on the assumption that features have a normal distribution. This means that if predictors take continuous values rather than discrete values, the model assumes these values are drawn from the Gaussian distribution.

- **Multinomial Naïve Model**: When the data is multinomially distributed, the Multinomial Naive Bayes classifier is used. It is primarily used to solve document classification problems, indicating which category a particular document belongs to, such as sports, politics, education, and so on. The predictors of the classifier are based on the frequency of words.
- **Bernoulli Naïve Model**: The Bernoulli classifier operates similarly to the multinomial classifier, except that the predictor variables are independent Boolean's variables. For example, whether or not a specific word appears in a document. This model is also well-known for performing document classification tasks.

Advantages and disadvantages of Naïve Bayes

Following are the advantages:

- Easy and fast algorithm to predict a class of datasets.
- Used for binary as well as multi-class classifications.
- When compared to numerical variables, it performs well with categorical input variables.
- It is the most widely used test for text classification problems.

Following are the disadvantages:

- Naive Bayes is based on the assumption that all predictors (or features) are independent.
- The '*zero-frequency problem*' is confronted by this algorithm.

Applications of Naïve Bayes Algorithms

Following are the applications of Naïve Bayes Algorithms:

- Text classification
- Spam Filtering
- Real-time Prediction
- Multi-class Prediction
- Recommendation Systems
- Credit Scoring
- Sentiment Analysis

Decision Tree

A decision tree is a supervised learning technique that can be applied to classification and regression problems.

- Decision trees are designed to mimic human decision-making abilities, making them simple to understand.
- Because the decision tree has a tree-like structure, the logic behind it is easily understood.

Decision-tree working

Following steps are involved in the working of Decision-tree:

Step-1: Begin the tree with node T, which contains the entire dataset.

Step-2: Using the Attribute Selection Measure, find the best attribute in the dataset.

Step-3: Divide the T into subsets that contain the best possible values for the attributes.

Step-4: Create the decision tree node with the best attribute.

Step-5: Make new decision trees recursively using the subsets of the dataset created in step-3.

Continue this process until you reach a point where you can no longer classify the nodes and refer to the final node as a leaf node.

General DT-structure: Following is the general decision tree structure is shown in *Figure 2.10*:

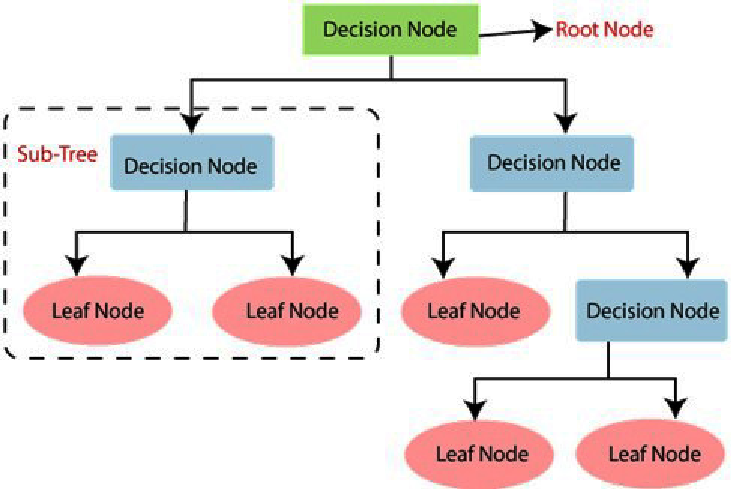


Figure 2.10*: General Decision Tree structure*

Note: A decision tree can contain both categorical (Yes/No) and numerical data.

Example of decision-tree

Assume we want to play badminton on a specific day, say Saturday - how will you decide whether or not to play? Assume you go outside to see if it is hot or cold, the speed of the wind and humidity, and the weather, i.e., whether it's sunny, cloudy, or rainy. You consider all of these factors when deciding whether or not to play. *Table 2.3* shows the weather observations of the last ten days.

Day	Weather	Temperature	Humidity	Wind	Play?
1	Sunny	Hot	High	Weak	No
2	Cloudy	Hot	High	Weak	Yes
3	Sunny	Mild	Normal	Strong	Yes
4	Cloudy	Mild	High	Strong	Yes
5	Rainy	Mild	High	Strong	No
6	Rainy	Cool	Normal	Strong	No
7	Rainy	Mild	High	Weak	Yes
8	Sunny	Hot	High	Strong	No
9	Cloudy	Hot	Normal	Weak	Yes
10	Rainy	Mild	High	Strong	No

***Table 2.3**: Weather Observations of the last ten days*

A decision tree is a great way to represent the data because it follows a tree-like structure and considers all possible paths that can lead to the final decision.

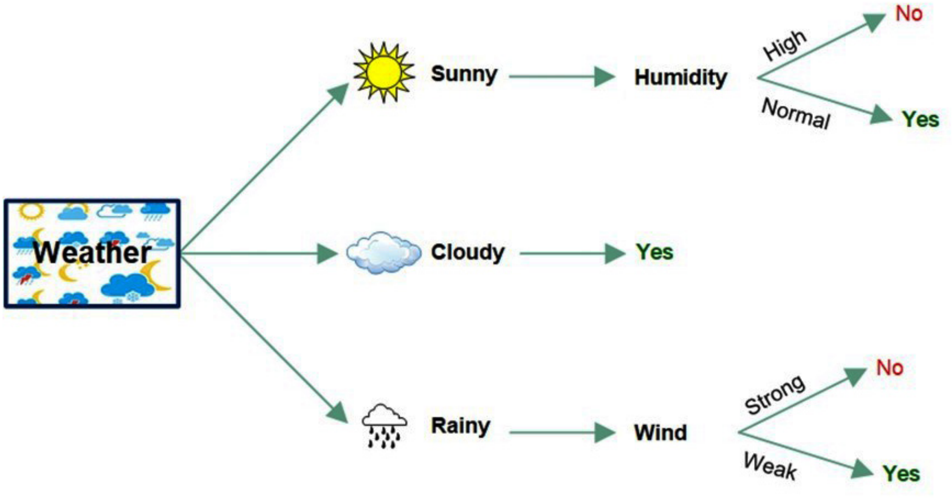


***Figure 2.11:** Play Badminton decision tree*

Figure 2.11 depicts a learned decision tree. Each node represents an attribute or feature, and the branch from each node represents the node's outcome. Finally, the final decision is made by the tree's leaves.

Advantages of decision tree

Following are the advantages of decision tree:

- Simple and easy to understand.
- Popular technique for resolving decision-making problems.
- It aids in considering all possible solutions to a problem.
- Less data cleaning is required.

Disadvantages of the decision tree

Following are the disadvantages of decision tree:

- Overfitting problem
- Complexity

K-Nearest Neighbors (K-NN) algorithm

The **K-Nearest Neighbors** (**K-NN**) algorithm is a clear and simple supervised machine learning algorithm that can be used to solve regression and classification problems.

The K-NN algorithm assumes that the new case and existing cases are similar and places the new case in the category that is most similar to the existing categories. The K-NN algorithm stores all available data and classifies a new data point based on its similarity to the existing data. This means that when new data appears, the KNN algorithm can quickly classify it into a suitable category.

K-NN is a non-parametric algorithm, which means it makes no assumptions about the data it uses. It's also known as a *lazy learner algorithm* because it doesn't learn from the training set right away; instead, it stores the dataset and performs an action on it when it comes time to classify it.

Pattern recognition, data mining, and intrusion detection are some of the demanding applications.

Need of the K-NN Algorithm

Assume there are two categories, Category A and Category B, and we have a new data point x1. Which of these categories will this data point fall into? A K-NN algorithm is required to solve this type of problem. We can easily identify the category or class of a dataset with the help of K-NN as shown in *Figure 2.12*:

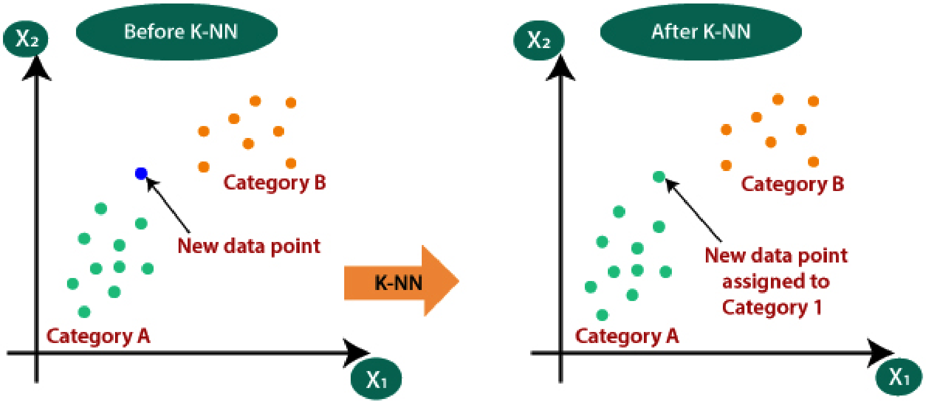


***Figure 2.12:** Category or class of a dataset with the help of K-NN*

The following algorithm can be used to explain how KNNs work:

Step-I: Select the no. K of the neighbors.

Step-II: Determine the Euclidean distance between K neighbors.

Step-III: Take the K closest neighbors based on the Euclidean distance calculated.

Step-IV: Count the number of data points in each category among these K neighbors.

Step-V: Assign the new data points to the category with the greatest number of neighbors.

Step-VI: Our model is complete.

Example: Let's say we have a new data point that needs to be placed in the appropriate category. Consider the following illustration:

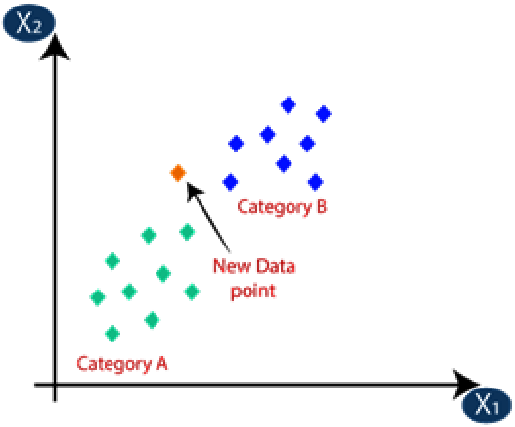


***Figure 2.13:** K-NN example*

First, we'll decide on the number of neighbors, so we'll go with *k=5*.

The Euclidean distance between the data points will then be calculated as shown in *Figure 2.14*. The Euclidean distance is the distance between two points that we learned about in geometry class. It can be calculated using the following formula:

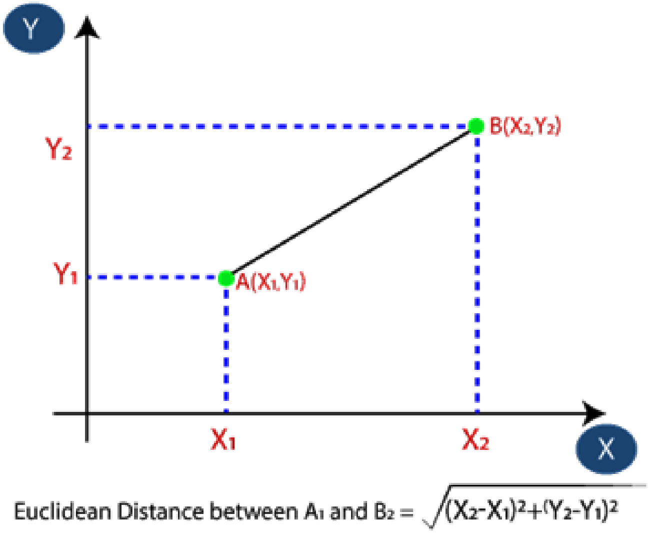

Y
Y2
Y1
B(X2,Y2)
A(X1,Y1)
X1
X2
X
Euclidean Distance between A1 and B2 = $\sqrt{(X_2-X_1)^2+(Y_2-Y_1)^2}$

Figure 2.14: *The Euclidean distance between the data points*

We found the closest neighbors by calculating the Euclidean distance, which yielded three closest neighbors in category A and two closest neighbors in category B as shown in *Figure 2.15*. Consider the following illustration:

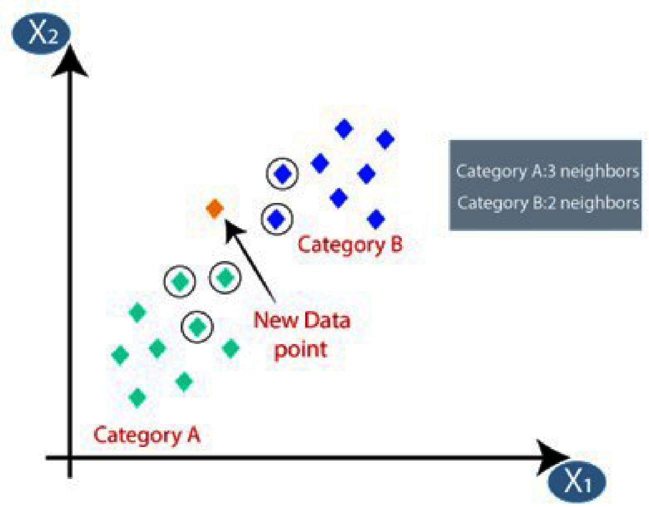


Figure 2.15: *Closest neighbors for the Category A and B*

As can be seen, the three closest neighbors are all from category A, so this new data point must also be from that category.

Logistic Regression

The classification algorithm logistic regression is used to assign observations to a discrete set of classes. Unlike linear regression, which produces a continuous number of values, logistic regression produces a probability value that can be mapped to two or more discrete classes using the logistic sigmoid function.

A regression model with a categorical target variable is known as logistic regression. To model binary dependent variables, it employs a logistic function.

The target variable in logistic regression has two possible values, such as yes/no. Consider how the target variable y would be represented in the value of "yes" is 1 and "no" is 0. The log-odds of y being 1 is a linear combination of one or more predictor variables, according to the logistic model. So, let's say we have two predictors or independent variables, x_1 and x_2, and p is the probability of y equaling 1. Then, using the logistic model as a guide:

$$\ln \frac{p}{1 - p} = a + bx_1 + cx_2$$

We can recover the odds by exponentiating the equation:

$$\frac{p}{1 - p} = e^{(a + b_{x1} + c_{x2})}$$

$$p = \frac{p = \mathrm{e}(\mathrm{a} + \mathrm{b}_{\mathrm{x1}} + \mathrm{c}_{\mathrm{x2}})}{1 + e^{(\mathrm{a}+\mathrm{b}_{\mathrm{x1}} + \mathrm{c}_{\mathrm{x2}})}}$$

$$p = \frac{1}{1 + e^{-(\mathrm{a}+\mathrm{b}_{\mathrm{x1}} + \mathrm{c}_{\mathrm{x2}})}}$$

As a result, the probability of y is 1. If p is closer to 0, y equals 0, and if p is closer to 1, y equals 1. As a result, the logistic regression equation is:

$$y = \frac{1}{1 + e^{-(\mathrm{a}+\mathrm{b}_{\mathrm{x1}} + \mathrm{c}_{\mathrm{x2}})}}$$

This equation can be generalized to n number of parameters and independent variables as follows:

$$y = \frac{1}{-(\beta_0 + \beta_1 x_1 + \ldots + \beta_n x_n)}$$

Comparison between linear and logistic regression

Different things can be predicted using linear regression and logistic regression as shown in *Figure 2.16*.

Linear Regression: The predictions of linear regression are continuous (numbers in a range). It may be able to assist us in predicting the student's test score on a scale of 0 to 100.

Logistic Regression: The predictions of logistic regression are discrete (only specific values or categories are allowed). It might be able to tell us whether the student passed or failed. The probability scores that underpin the model's classifications can also be viewed.

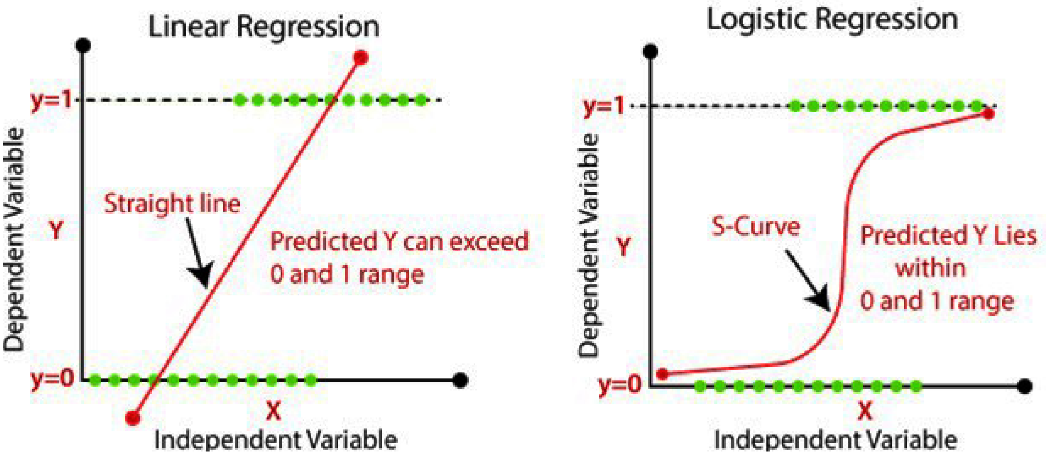


Figure 2.16: *Linear regression vs Logistic regression*

While linear regression has an infinite number of possible outcomes, logistic regression has a set of predetermined outcomes.

When the response variable is continuous, linear regression is used, but when the response variable is categorical, logistic regression is used.

A continuous output, such as a stock market score, is an example of logistic regression and predicting a defaulter in a bank using transaction details from the past is an example of linear regression.

The following *Table 2.4* shows the difference between Linear and Logistic Regression:

Sr.No	Linear Regression	Logistic Regression
1	Target is an interval variable.	Target is a discrete variable.
2	Predicted values are the mean of the target variable at the given values of the input variables.	Predicted values are the probability of a particular level of the target variable at the given values of the input variables.
3	Solves Regression problems.	Solves Classification problems.
4	The graph is a straight line.	The graph is an S-curve.
5	A numeric variable is used .	A categorical variable is used.
6	It requires the relationship between dependent and independent variables.	It doesn't require the relationship between dependent and independent variables.

Table 2.4: *Difference between Linear and Logistic Regression*

Types of Logistic Regression

Logistic regression is mainly categorized into three types as shown in *Figure 2.17*:

1. Binary Logistic Regression
2. Multinomial Logistic Regression
3. Ordinal Logistic Regression

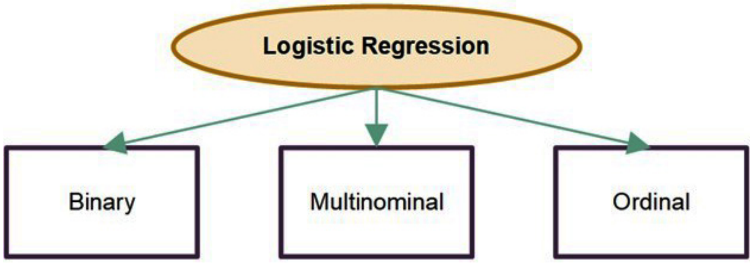


Figure 2.17: *Types of Logistic Regression*

Binary Logistic Regression

The target variable or dependent variable in binary logistic regression is binary in nature, meaning it has only two possible values.

There are only two possible outcomes for the categorical response.

For example, determining whether or not a message is a spam.

Multinomial Logistic Regression

In a multinomial logistic regression, the target variable can have three or more values, but there is no fixed order of preference for these values.

For instance, the most popular type of food (Indian, Italian, Chinese, and so on.)

Predicting which food is preferred more is an example (Veg, Non-Veg, Vegan).

Ordinal Logistic Regression

The target variable in ordinal logistic regression has three or more possible values, each of which has a preference or order.

For instance, restaurant star ratings range from 1 to 5, and movie ratings range from 1 to 5.

Examples

The following are some of the scenarios in which logistic regression can be used.

- **Weather Prediction :**Logistic regression is used to make weather predictions. We use the information from previous weather reports to forecast the outcome for a specific day. However, logistic regression can only predict categorical data, such as whether it will rain or not.

- **Determining Illness:** We can use logistic regression with the help of the patient's medical history to predict whether the illness is positive or negative.

Support Vector Machine (SVM) Algorithm

The Support Vector Machine, or SVM, is a popular Supervised Learning algorithm that can be used to solve both classification and regression problems. However, it is primarily used in machine learning for classification problems as shown in *Figure 2.18*.

Many people prefer SVM because it produces significant accuracy while using less computing power. The extreme points/vectors that help create the hyperplane are chosen by SVM. Support vectors are the extreme cases hence the algorithm is called a **Support Vector Machine**.

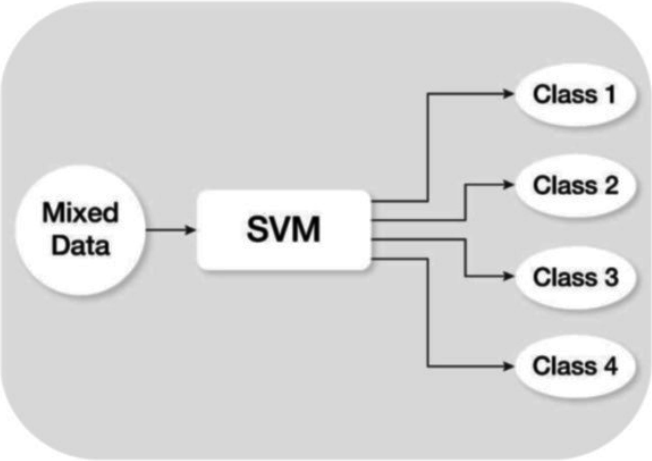


Figure 2. 18: *Support Vector Machine (SVM) concept*

The support vector machine algorithm's goal is to find a hyperplane in N-dimensional space (N - the number of features) that categorizes the data points clearly.

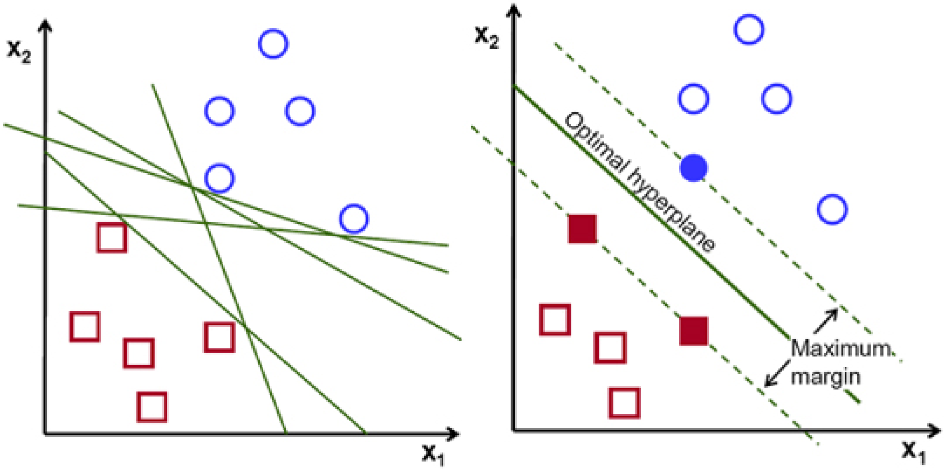


Figure 2.19: *SVM hyperplanes with Maximum margin*

There are numerous hyperplanes to choose from to separate the two classes of data points. Our goal is to find a plane with the greatest margin, or the greatest distance between data points from both classes as shown in *Figure 2.19*. Maximizing the margin distance provides some reinforcement, making it easier to classify future data points.

Hyperplanes are decision boundaries that aid in data classification. Different classes can be assigned to data points on either side of the hyperplane. The hyperplane's dimension is also determined by the number of features. If there are only two input features, the hyperplane is just a line. The hyperplane becomes a two-dimensional plane when the number of input features reaches three. When the number of features exceeds three, it becomes difficult to imagine.

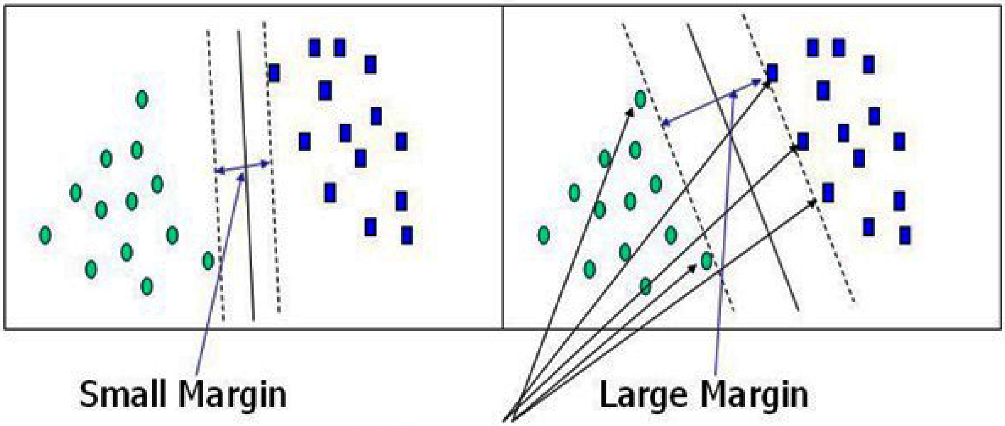


Figure 2.20: *Support Vectors with small and large margin*

Support vectors are data points that are closer to the hyperplane and have an influence on the hyperplane's position and orientation as shown in *Figure 2.20*. We maximize the classifier's margin by using these support vectors. The hyperplane's position will be altered if the support vectors are deleted. These are the points that will assist us in constructing SVM.

The sigmoid function is used in logistic regression to squash the output of the linear function within the range of [0,1]. If the squashed value exceeds a threshold value (0.5), it is labeled as 1, otherwise, it is labeled as 0. In SVM, we take the output of a linear function and, if it is greater than 1, we assign it to one class, and if it is less than 1, we assign it to another. We get this reinforcement range of values ([-1,1]) which acts as a margin because the threshold values in SVM are changed to 1 and -1.

Hyperplane, Support Vectors, and Margin

The Hyperplane, Support Vectors, and Margin are described as follows:

- **Hyperplane**: In n-dimensional space, there can be multiple lines/decision boundaries to separate the classes, but we need to find the best decision boundary to help classify the data points. The hyperplane of SVM refers to the best boundary. The hyperplane's dimensions are determined by the features in the dataset; for example, if there are two features, the hyperplane will be a straight line. If three features are present, the hyperplane will be a two-dimensional plane. We always make a hyperplane with a maximum margin, which refers to the distance between data points.

- **Support Vectors:** Support vectors are the data points or vectors that are closest to the hyperplane and have an effect on the hyperplane's position. These vectors are called **support vectors** because they support the hyperplane.
- **Margin**: It's the distance between two lines on the closest data point of different classes. The perpendicular distance between the line and the support vectors can be calculated. A large margin is regarded as a good margin, while a small margin is regarded as a bad margin.

Working of SVM

In multidimensional space, an SVM model is essentially a representation of different classes in a hyperplane. SVM will generate the hyperplane in an iterative manner in order to reduce the error. SVM's goal is to divide datasets into classes so that a maximum marginal hyperplane can be found.

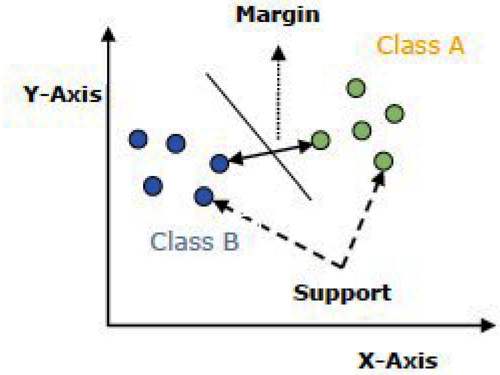


Figure 2.21: *Working of SVM*

SVM's main goal is to divide datasets into classes in order to find a **maximum marginal hyperplane** (**MMH**), which can be accomplished in two steps:

First, SVM will iteratively generate hyperplanes that best separate the classes.

The hyperplane that correctly separates the classes will then be chosen.

Types of SVM

Support Vector Machine (SVM) types are described below:

- **Linear SVM**: Linear SVM is used for linearly separable data, which means that if a dataset can be classified into two classes using only a single straight line, it is called **linearly separable data**, and the classifier used is called Linear SVM.

- **Non-linear SVM**: Non-Linear SVM is used to classify **non-linearly separated data**, which means that if a dataset cannot be classified using a straight line, it is classified as non-linear data, and the classifier used is the Non-Linear SVM classifier.

Applications of Support-Vector Machines

The following are few of the applications of Support-Vector Machines:

- Facial expressions classifications
- Pattern classification and regression problems
- In the military datasets
- Speech recognition
- Predicting the structure of proteins
- In image processing - handwriting recognition
- In earthquake potential damage detections

Advantages of SVM

The following are the advantages of SVM:

- SVM classifiers are highly accurate and work well in high-dimensional environments. Because SVM classifiers only use a subset of training points, they require very little memory.
- Solve the data points that are not linearly separable.
- Effective in a higher dimension space.
- It works well with a clear margin of separation.
- It is effective in cases where the number of dimensions is greater than the number of samples.
- Better utilization of memory space.

Disadvantages of SVM

The following are the disadvantages of SVM:

- Not suitable for the larger data sets.
- Less effective when the data set has more noise.

- It doesn't directly provide probability estimates.
- Overfitting problem

Random Forest Algorithm

Random Forest is a well-known machine learning algorithm that uses the supervised learning method. In machine learning, it can be used for both classification and regression problems. It is based on ensemble learning, which is a method of combining multiple classifiers to solve a complex problem and improve the model's performance.

Random Forest is a classifier that combines a number of decision trees on different subsets of a dataset and averages the results to improve the dataset's predictive accuracy. Instead of relying on a single decision tree, the random forest takes the predictions from each tree and predicts the final output based on the majority votes of predictions.

The greater the number of trees in the forest, the more accurate it is and the problem of overfitting is avoided.

Each tree in the random forest produces a class prediction, and the class with the most votes becomes the prediction of our model (see figure). *Figure 2.22* is showing the random forest algorithm visualization and having the Six 1's and three 0's therefore Prediction is 1:

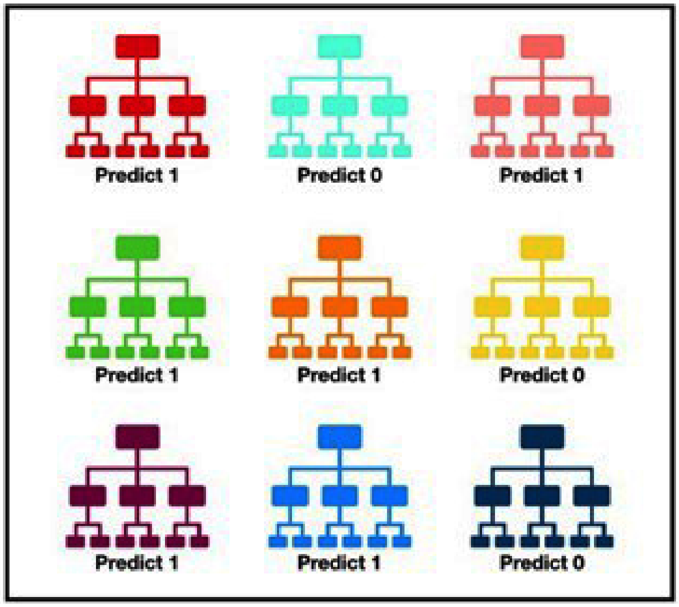


Figure 2.22: *Random Forest Algorithm Visualization*

Working of the Random Forest Algorithm

The random forest is created in two phases: the first is to combine N decision trees to create the random forest, and the second is to make predictions for each tree created in the first phase.

The following steps and *Figure 2.23* can be used to explain the working process:

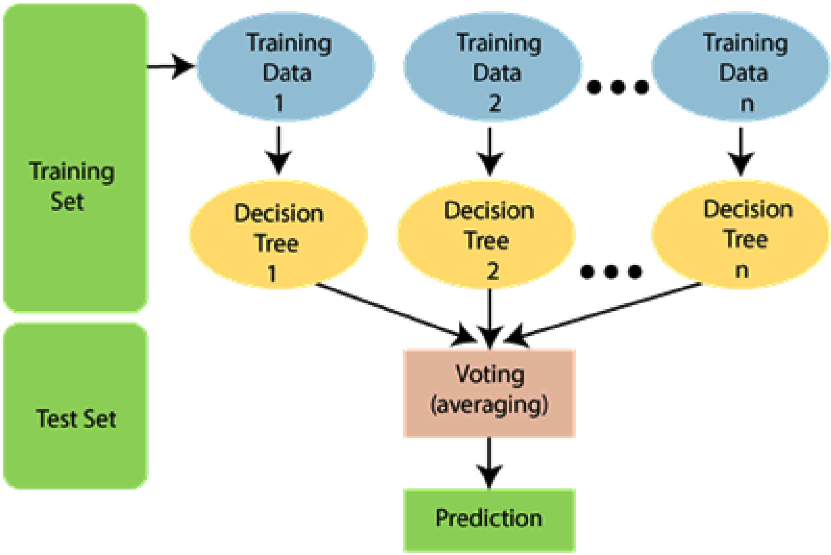


***Figure 2.23:** Working of the Random Forest algorithm*

Step-I: Choose K data points at random from the training data set.

Step-II: Create decision trees for the data points chosen (subsets).

Step-III: Choose N for the number of decision trees you want to create.

Step-IV: Repeat Steps I and II.

Step-V: Find the predictions of each decision tree for new data points and assign the new data points to the category with the most votes.

Example: Assume you have a dataset with a variety of fruit images. As a result, the random forest classifier is given this dataset. Each decision tree is given a subset of the dataset to work with. During the training phase, each decision tree generates a prediction result, and when a new data point appears, the random forest classifier predicts the final decision based on the majority of results. *Figure 2.24* is showing the random forest fruit's instance example:

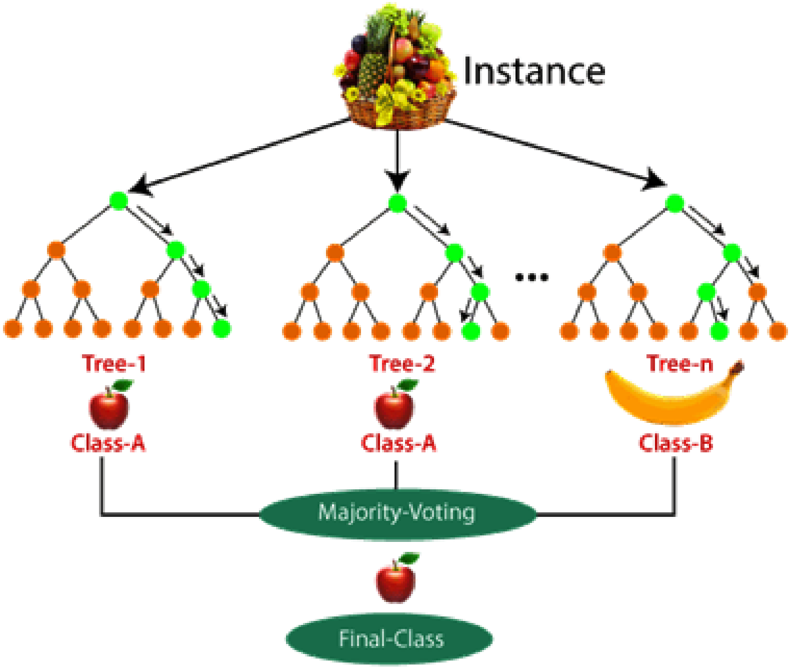


Figure 2.24: *Random Forest Fruit's instance example*
Source: https://www.javatpoint.com/machine-learning-random-forest-algorithm

Advantages of Random Forest Algorithm

The following are some reasons why we should use the random forest algorithm:

- Less training time required as compared to other algorithms.
- Output with high accuracy.
- Effective for the large dataset.
- Used for the classification and regression.
- Prevents the overfitting problems.
- It can be used for identifying the most important features from the training dataset i.e., feature engineering.

Disadvantages of Random Forest Algorithm

Despite the fact that random forest can be used for both classification and regression tasks, it is not better suited to regression.

Applications of Random Forest Algorithm

The random forest algorithm concept is most commonly used in four sectors as follows:

- **Banking Sector**: This algorithm is primarily used in the banking industry to identify loan risk.
- **Medical Sector**: The disease trends and risks can be identified with the help of this algorithm.
- **Land Development Sector**: This algorithm can identify areas with similar land use.
- **Marketing Sector**: This algorithm can be used to spot marketing trends.

Summary

- Supervised learning is distinguished by the use of labeled datasets to train algorithms that accurately classify data or predict outcomes.
- Supervised machine learning can be further classified into regression and classification.
- Regression algorithms are used if there is a relationship between one or more input variables and the output variable.
- Simple linear regression is a type of linear regression where we have only one independent variable to predict the dependent variable.
- If there is more than one independent variable for the prediction of a dependent variable, then this type of linear regression is known as multiple linear regression.
- Classification is the process to group the output into different classes based on one or more input variables.
- The Naïve Bayes classifiers algorithm is a supervised learning algorithm that solves classification problems and is based on the Bayes theorem.
- The Naive Bayes Model is classified into three types: Gaussian Naïve Model, Multinomial Naïve Model and Bernoulli Naïve Model.
- A decision tree is a supervised learning technique that can be applied to classification and regression problems.
- The K-Nearest Neighbors (K-NN) algorithm stores all available data and classifies a new data point based on its similarity to the existing data.

- K-NN also known as a lazy learner algorithm because it doesn't learn from the training set right away.
- A regression model with a categorical target variable is known as logistic regression.
- The predictions of linear regression are continuous (numbers in a range).
- The predictions of logistic regression are discrete (only specific values or categories are allowed).
- Logistic regression is mainly categorized into three types: Binary, Multinomial, and Ordinal.
- Support Vector Machine (SVM) Algorithm produces significant accuracy while using less computing power.
- There are two types of SVM: Linear and Non-linear.
- Random forest is a well-known machine learning algorithm that uses the supervised learning method.

Exercise (MCQs)

1. **Supervised learning and unsupervised clustering both require at least one.**

   a) hidden attribute.

   b) output attribute.

   c) input attribute.

   d) categorical attribute.

2. **Supervised learning differs from unsupervised clustering in that supervised learning requires:**

   a) at least one input attribute.

   b) input attributes to be categorical.

   c) at least one output attribute.

   d) output attributes to be categorical.

3. **A regression model in which more than one independent variable is used to predict the dependent variable is called:**

   a) a simple linear regression model

   b) a multiple regression models

   c) an independent model

   d) none of the above

4. **A ____________________is a decision support tool that uses a tree-like graph or model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility.**

   a) Decision tree

   b) Graphs

   c) Trees

   d) Neural Networks

5. **Decision trees can be used for classification tasks.**

   a) True

   b) False

6. **Sentiment analysis is an example of:**

   a) Regression

   b) Classification

   c) Clustering

   d) Reinforcement learning

      i. 1 Only

      ii. 1 and 2

      iii 1 and 3

      iv. 1, 2 and 4

7. **In supervised learning**

   a) Classes are not predefined

   b) Classes are predefined

   c) Classes are not required

   d) Classification is not done

8. **What are the two types of learning?**

   a) Improvised and unimprovised

   b) Supervised and unsupervised

   c) Layered and unlayered

   d) None of the above

9. **____________________algorithm stores all available data and classifies a new data point based on its similarity to the existing data.**

   a) Naive Bayes

   b) K-Nearest Neighbors

c) Decision Tree

d) Support Vector Machine

10. **Random forest is a well-known machine learning algorithm that uses:**

    a) Supervised learning

    b) Unsupervised learning

    c) Hybrid learning

    d) None of these

Answer

1. **a**
2. **b**
3. **c**
4. **a**
5. **a**
6. ***iv***
7. **b**
8. **b**
9. **b**
10. **a**

Short Answers Questions

1. Explain the concept of supervised learning with an example.
2. Difference between supervised and unsupervised machine learning.
3. What is Naive Bayes algorithm and how does it work?
4. Write the applications of the Naive Bayes Algorithm
5. What is the random forest algorithm and explain how it works?
6. What is a decision tree and explain how it is useful?
7. Define support vector, hyperplane, and margin.
8. Explain the types of supervised learning.
9. What is the Logistic Regression?
10. Explain in detail about the Linear Regression.

Long Answers Questions

1. What are the pros and cons of using Naive Bayes?
2. Explain the random forest algorithm with a real-life example.
3. Difference between classification and regression with example.
4. What do you mean by overfitting? How to avoid this?
5. Explain the K-Nearest Neighbor with help of an example.
6. What do you mean by SVM? Explain its types?

CHAPTER 3
Unsupervised Learning

Unsupervised learning is an ML technique that deals with unlabeled data. This unlabeled data is used by unsupervised learning algorithms to discover patterns of correlated data and information. These algorithms don't require any supervision, instead, it works on its techniques to discover the patterns. As compared to other learning methods, unsupervised learning methods are more unpredictable. Moreover, as compared to supervised learning, unsupervised learning performs more complex tasks, that includes: neural network, anomaly detection, clustering, and so on.

Structure

In this chapter, you will learn the following topics:

- Unsupervised Learning
- Clustering
- Hierarchical clustering
- K-mean clustering
- Probabilistic clustering
- Apriori Algorithm

- Association rule mining
- Gaussian Mixture Model (GMM)
- FP-Growth Algorithm

Objectives

After reading this chapter, you will be able to understand the concepts of various algorithms of unsupervised learning techniques. Knowledge of these algorithms will help you to analyze, evaluate, and group the unlabeled data items using various clustering techniques.

Unsupervised Learning

This is a powerful machine learning technique. In this technique, the designed model doesn't require any observation or supervision to process the unlabeled data. Instead, this technique accepts unlabeled data items and discovers new patterns of information that were previously unknown or undetected. We can understand this technique with the help of its working process that is shown in *Figure 3.1*.

Working of unsupervised learning

It is easy to understand the working of unsupervised learning through the following steps:

i. An unsupervised learning model accepts unlabeled raw data items as input. Unlike supervised learning, it doesn't require any training, testing, and labeled output data.

ii. After the interpretation of data, it is required to choose an appropriate or desired algorithm to discover new patterns (that were previously unknown) of information.

iii. Identified new patterns must be processed for assigning new labels. In *Figure* 3.1, identified patterns have been shown with different colors:

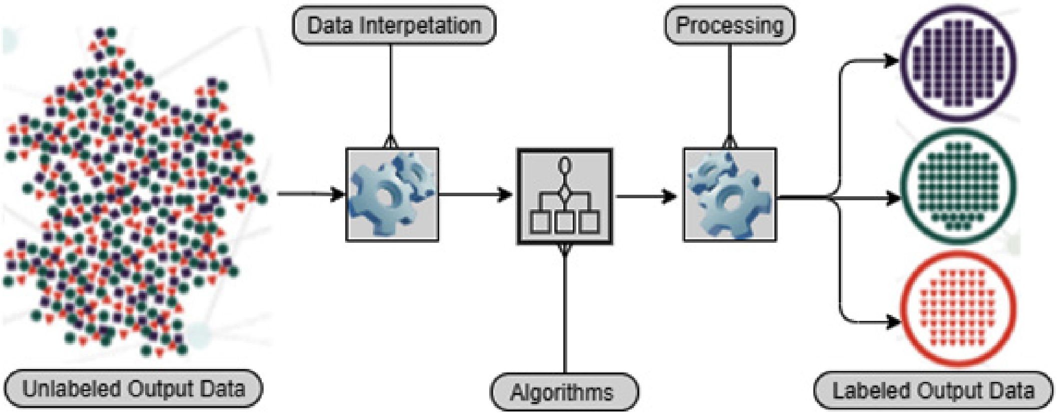


***Figure 3.1**: Working of unsupervised learning*

Need for using unsupervised learning

There are several reasons for which unsupervised learning is a perfect choice:

- Unsupervised ML finds all sorts of unexplained data patterns.
- Unsupervised learning techniques are used to identify properties that can help to make groups.
- Everything is carried out in real-time, so in presence of learners, all the input data is to be evaluated and labeled.
- Unlabeled data is easier to read from the computer than the labeled data, which requires manual operations.

Algorithms

As compared to supervised learning, an algorithm used in unsupervised learning performs a more complex task. Moreover, algorithms used in unsupervised learning are more unpredictable than the other natural learning algorithms. Unsupervised learning algorithms include:

- Clustering,
- Anomaly detection,
- Neural networks, etc.

Clustering

Clustering is an unsupervised ML technique. The task of clustering is to find patterns or structures from unstructured data. The process of clustering is to divide the data points or population into some groups, such that data points of the same groups having similar characteristics while data points belonging to the different groups must have dissimilar properties. These natural groups are called clusters. A natural example of clustering is shown in *Figure 3.2*, where the two groups or clusters have been created from the mixed population of birds and animals. The population belongs to the same group having similar characteristics such as one cluster is having a population of birds and another cluster having a population of animals. In unsupervised ML, we can adjust the required number of clusters by altering the clustering algorithms. It helps us to change these groups' granularity.

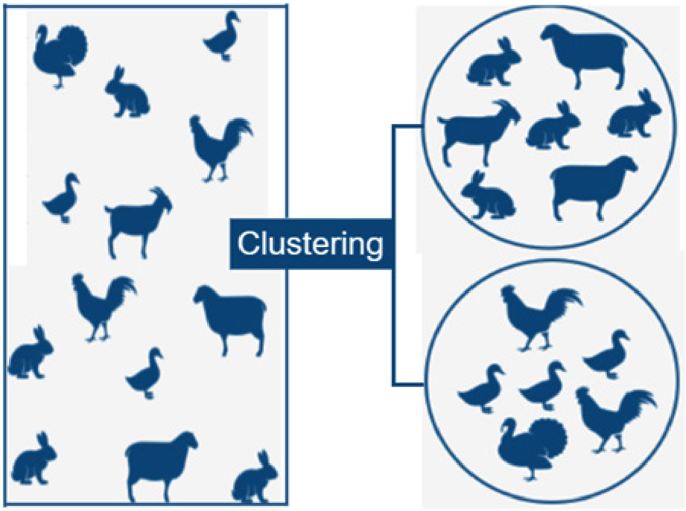


Figure 3.2: *(a) Before Clustering* *(b) After Clustering*

Clustering can be performed in the following four ways:

- A method of clustering to partition the raw data into some groups such as a particular data belongs to only one group called **Exclusive(partitioning)**. For example: K-means clustering.

- In the **Agglomerative clustering** method, in the beginning, each data point will be considered as a cluster. After that, the number of clusters decreases through iterative unions between the two closest clusters. For example: Hierarchical clustering.

- **Overlapping clustering** is a method of clustering whereas each data point may belong to more than one cluster with a separate degree of membership. Data would be connected to a suitable membership attribute. For example: Fuzzy C-Means.

- **Probabilistic clustering** is another method of clustering in which the probability distribution technique is used to form the clusters. An example of probabilistic clustering is: Following keywords.

K-means Clustering

K-mean clustering is an exclusive partitioning technique to classify the raw data into a k-number of groups, where k is a positive integer. Grouping of data is based on calculating and minimizing the distance of data points from the selected centroids. Centroids are the heart of this algorithm. The number of centroids depends on the number of required groups of clusters. Centroid catches and adds the points nearest to them to the cluster through an iterative process.

Algorithm of k-mean clustering

A step-by-step process of k-mean clustering has been shown in an algorithm given as follows:

Input: Raw data items (n elements)

Output: Labeled groups of data items (k clusters)

Step 1: Start by choosing the value of k (number of clusters).

Step 2: For initial partitioning of raw data into k clusters, choose k elements as centroids that classify data. We can choose the initial centroids systematically or randomly as per the following steps:

- **a. Take the first k training sample (centroids)**
- **b. For n-k elements, each element will be added to a cluster of the nearest centroid. After each assignment, the centroids of the clusters must be recomputed.**

Step 3: By putting every sample in sequence calculate the distance of elements from the centroid of each cluster. Put the sample element into the cluster which is having the closest distance with centroids. After switching the samples into the nearest centroid's cluster update the centroid.

Step 4: Repeat step 3 until convergence, i.e. during the iteration, not a single sample moves from one cluster to another.

- If the data count is less than the cluster count, each data will be considered the centroid of the cluster and each centroid will be assigned a cluster number.
- If the data count is greater than the number of clusters then for every sample data calculate the distance from each centroid. The data which is having the minimum distance will belong to the cluster which is having the minimum distance with that data.

- As the location of the centroid is uncertain, therefore the location of the centroid must be recalculated based on the recently updated cluster's data. This process will be repeated until convergence.

Flowchart of k-mean clustering

For more illustration of the k-mean algorithm, a flow chart of this algorithm is shown in *Figure 3.3*:

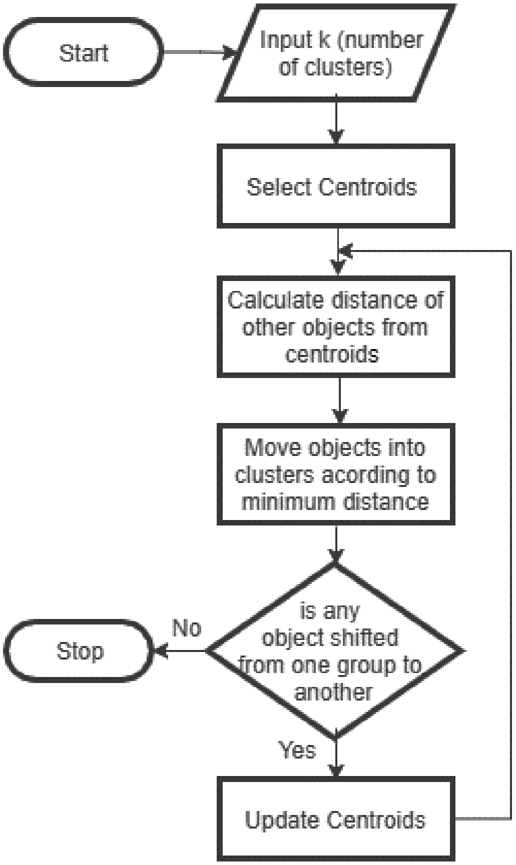

Start
Input k (number of clusters)
Select Centroids
Calculate distance of other objects from centroids
Move objects into clusters acording to minimum distance
is any object shifted from one group to another
No
Stop
Yes
Update Centroids

***Figure 3.3**: Flowchart of k-mean clustering algorithm*

A practical example of k-mean clustering

For example, we have four objects, each object is has 2 attributes. The first attribute of each object is a weight-index, represented by label W and another attribute is pH represented by label P. Objects, attributes, and their values are shown in *Table 3.1*. Our objective is to clutch these objects into two clusters based on the values of their attributes.

Objects	W	P
E	1	1
F	2	1
G	4	3
H	5	4

***Table 3.1:** Input dataset for k-mean clustering*

Every object denotes one point with two attributes (W, P). Objects and corresponding values of their attributes have been plotted and shown in *Figure 3.4* using the graph:

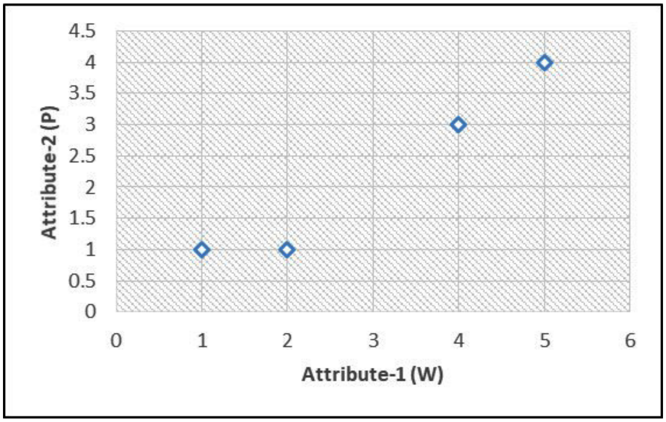


***Figure 3.4:** Scatter plot of the input dataset*

1. **Choose centroid**: Assume object E and object F have been chosen as initial centroids. The coordinates of the centroid E and F are represented through symbols Cn1 and Cn2, respectively. *Cn1= (1, 1)* and *Cn2= (2, 1)*, as shown in *Figure 3.5:*

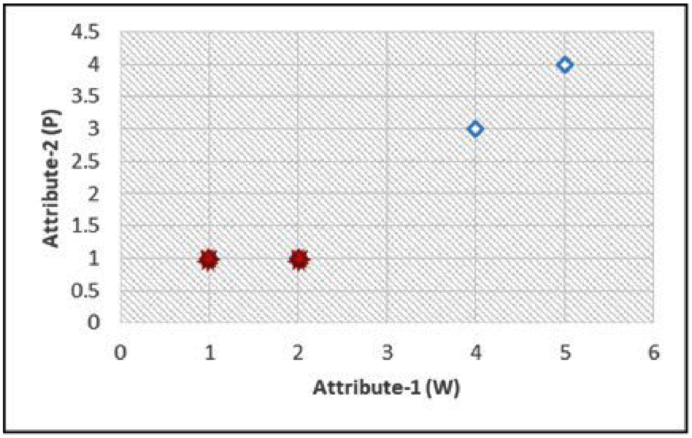


***Figure 3.5:** Representation of centroid with different color*

2. **Objects-centroids distance**: The Euclidean distance between each object and cluster-centroid is calculated and represented through the distance matrix in *Figure 3.6*:

$$\text{DistanceMatrix} = \begin{array}{cl} \begin{bmatrix} 0 & 1 & 3.61 & 5 \\ 1 & 0 & 2.83 & 4.24 \end{bmatrix} & \begin{array}{l} \text{Cn1=(1,1) Centroid-group\_1} \\ \text{Cn2=(2,1) Centroid-group\_2} \end{array} \\ \begin{array}{cccc} \boxed{E} & \boxed{F} & \boxed{G} & \boxed{H} \end{array} & \end{array}$$

$$\text{Objects Matrix} = \begin{array}{cl} \begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4 \end{bmatrix} & \begin{array}{l} \boxed{W} \\ \boxed{P} \end{array} \end{array}$$

Figure 3.6: *Representation of distance matrix*

Every element of the distance matrix represents the Euclidean distance between the object and centroid. Elements of row-1 of the distance matrix are the distance between centroid Cn1 and objects E, F, G, and H respectively, whereas elements of row-2 of the distance matrix are the distance between centroid Cn2 and objects E, F, G, and H respectively as shown in *Figure 3.6*.

Example: Calculate the distance of object G from centroid Cn1 and Cn2, respectively.

1. Distance between G(4,3) and centroid Cn1 (1, 1)

   $= \sqrt{(4-1)^2 + (3-1)^2}\ |$

   $= 3.61$

2. Distance between G(4,3) and centroid Cn2 (2, 1)

   $= \sqrt{(4-2)^2 + (3-1)^2}$

   $= 2.83$

1. **Objects clustering**: Each object must be assigned based on the minimum distance. So, object E has been assigned to group-1, object F to group 2, object G to group 2, and object H to group 2. The element of the group matrix has been shown in *Figure 3.7*:

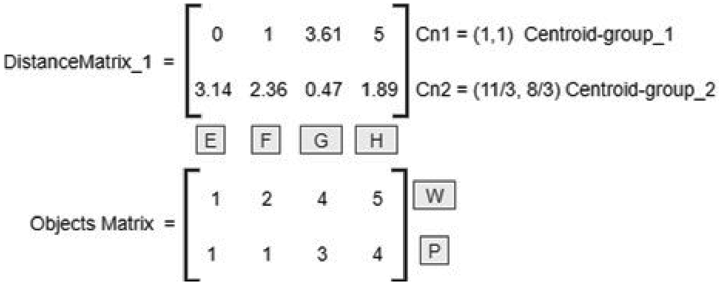
$$\text{GroupMatrix\_0} = \begin{array}{cl} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix} & \begin{array}{l} \text{Cluster\_1 (group\_1)} \\ \text{Cluster\_2 (group\_2)} \end{array} \\ \begin{array}{cccc} \boxed{E} & \boxed{F} & \boxed{G} & \boxed{H} \end{array} & \end{array}$$

Figure 3.7: *Elements of the group matrix*

2. **Iterations, calculate new centroids**: After assigning members to each group, a new centroid must be calculated. For every group, a new centroid will be calculated based on the values of its members. In this example, group-1 is having only one member therefore the centroid remains in Cn1= (1, 1). Whereas group-2 is having three members, therefore the centroid will be calculated by averaging the coordinates of all three members as follows:

$$Cn2 = \left(\frac{2+4+5}{3}, \frac{1+3+4}{3}\right) = (\frac{11}{3}, \frac{8}{3})$$

3. **Objects-centroid distances**: After calculating the new centroid, the distance of all objects from the new centroids have been calculated, and like step-2 a distance matrix has been created and shown as follows:



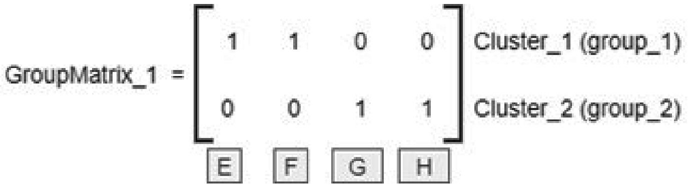
$$\text{DistanceMatrix\_1} = \begin{array}{c} \\ \\ \end{array}\begin{bmatrix} 0 & 1 & 3.61 & 5 \\ 3.14 & 2.36 & 0.47 & 1.89 \end{bmatrix} \begin{array}{l} \text{Cn1 = (1,1) Centroid-group\_1} \\ \text{Cn2 = (11/3, 8/3) Centroid-group\_2} \end{array}$$

$$\begin{array}{cccc} \boxed{E} & \boxed{F} & \boxed{G} & \boxed{H} \end{array}$$

$$\text{Objects Matrix} = \begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4 \end{bmatrix} \begin{array}{l} \boxed{W} \\ \boxed{P} \end{array}$$

***Figure 3.8**: Distance matrix and object matrix*

4. **Object clustering**: Like step 3, each object must be assigned to the updated group based on the minimum distance. Based on the new distance matrix, object F will be moved to group-1 and all the other objects will remain in the previously assigned group. The updated group matrix has been shown as follows:



$$\text{GroupMatrix\_1} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \begin{array}{l} \text{Cluster\_1 (group\_1)} \\ \text{Cluster\_2 (group\_2)} \end{array}$$

$$\begin{array}{cccc} \boxed{E} & \boxed{F} & \boxed{G} & \boxed{H} \end{array}$$

***Figure 3.9:** Group matrix*

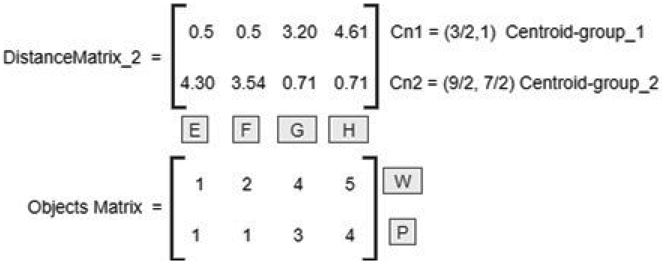
5. **Iteration2, determine centroids**: We have to repeat step-4 to determine the new centroid. Based on the clustering of the previous iteration, the coordinates of the new centroid must be calculated. As we can see group-1 and group-2 both have two members, thus the new centroids of centroids are as follows:

$$Cn1 = \left(\frac{1+2}{2}, \frac{1+1}{2}\right) = (1\frac{1}{2}, 1)$$

and

$$Cn2 = \left(\frac{4+5}{2}, \frac{3+4}{2}\right) = (4\frac{1}{2}, 3\ \frac{1}{2})$$

6. **Iteration2, objects-centroids distances**: We have to repeat step 2 to calculate the distance between the objects and updated centroids. Calculated distances are represented through the distance matrix shown as follows:

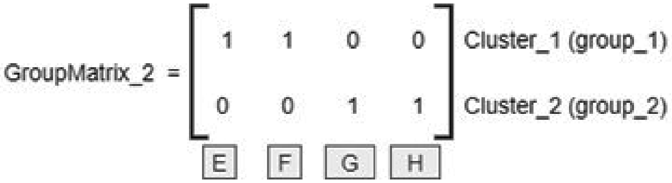


$$\text{DistanceMatrix\_2} = \begin{array}{c} \left[\begin{array}{cccc} 0.5 & 0.5 & 3.20 & 4.61 \\ 4.30 & 3.54 & 0.71 & 0.71 \end{array}\right] \\ \begin{array}{cccc} \boxed{E} & \boxed{F} & \boxed{G} & \boxed{H} \end{array} \end{array} \begin{array}{l} Cn1 = (3/2,1)\ \text{Centroid-group\_1} \\ Cn2 = (9/2,\ 7/2)\ \text{Centroid-group\_2} \\ \ \end{array}$$

$$\text{Objects Matrix} = \left[\begin{array}{cccc} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4 \end{array}\right] \begin{array}{c} \boxed{W} \\ \boxed{P} \end{array}$$

***Figure 3.10**: Distance matrix and object matrix*

7. **Iteration2, object clustering**: As per the distance matrix created in the previous step, group-1 or group-2 is assigned to each object based on the minimum distance. The resultant group matrix is shown as follows:



$$\text{GroupMatrix\_2} = \begin{array}{c} \left[\begin{array}{cccc} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{array}\right] \\ \begin{array}{cccc} \boxed{E} & \boxed{F} & \boxed{G} & \boxed{H} \end{array} \end{array} \begin{array}{l} \text{Cluster\_1 (group\_1)} \\ \text{Cluster\_2 (group\_2)} \\ \ \end{array}$$

***Figure 3.11**: Group matrix*

As we can see the groupmatrix_2 and groupmatrix_11 (calculated in the previous iteration) are the same i.e. no change in the group membership of objects in the second iteration. So we can say that there is no movement of objects in the current iteration thus computation of the k-mean clustering algorithm has reached permanency and no more iterations are needed. So final grouping is the result calculated through the k-mean clustering algorithm.

Object	Weight(W)	pH(P)	Group (result)
E	1	1	G1
F	2	1	G1
G	4	3	G2
H	5	4	G2

Hierarchical clustering

Hierarchical clustering is an unsupervised machine learning technique of data clustering that constructs the hierarchy of clusters. It takes an unlabeled dataset as an input and groups it into hierarchical clusters as an output therefore it is also known as **Hierarchical Cluster Analysis** (**HCA**). It starts with all the knowledge and data as an initial cluster. Two clusters similar to each other would be in the same cluster here. When there is only one cluster remaining, this algorithm stops. The output of the hierarchical clustering can be seen as a tree-like structure. That tree-like structure is called a **dendrogram**.

The effects and outcomes of hierarchical clustering and k-means clustering may often look identical; however, the working of both algorithms are different. Unlike a k-mean cluster algorithm, there is no need to pre-identify the number of groups or clusters.

Two approaches to hierarchical clustering

There are two approaches to hierarchical clustering, the first one is Agglomerative and another is Divisive.

In case of agglomerative hierarchical clustering, a bottom-up approach is used to collect the data as input. In the beginning, each data point of collected input is considered as a single cluster. Later, each cluster is merged with a similar cluster, this iteration is done until one cluster is left.

Whereas the divisive algorithm follows an inverse approach of the agglomerative algorithm. It uses a top-down approach. It takes the whole data as a single cluster and groups it into multiple clusters in a hierarchical manner.

Need of hierarchical clustering

Since we already have other algorithms for clustering, including the k-means clustering, why do we require hierarchical clustering? The answer is, in the k-mean clustering there are some limitations in the algorithm:

i. Need to choose a fixed number of clusters by choosing the value of parameter k.

ii. The algorithm always tries to create groups/clusters of the same size.

Hierarchical clustering can be used to overcome the limitations of k-means clustering because in this clustering there is no need to fix the number of clusters in the beginning.

In this chapter, we will discuss agglomerative hierarchical clustering in detail with an example.

Agglomerative hierarchical clustering

A common instance of HCA is the agglomerative hierarchical clustering algorithm. It uses the bottom-up approach to group the raw data into clusters. At the beginning of this algorithm, each dataset will be considered a single cluster, afterwards, the nearest pair of clusters will be combined to build a clubbed cluster. This process will be repeated until all the clusters are merged into a single cluster that contain datasets.

Working of agglomerative hierarchical clustering

A step-by-step working of the **Agglomerative Hierarchical clustering** (**AHC**) algorithm is shown as follows:

Step-1: Suppose there are N datasets, each dataset will be considered a single cluster. In the beginning, the number of clusters is equal to the number of datasets. In *Figure 3.12* each dot represents a cluster:

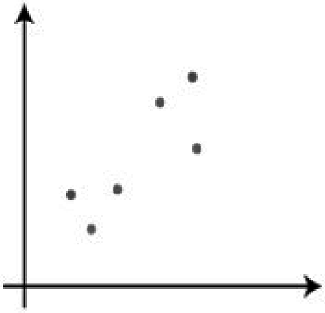


***Figure 3.12**: Each dot represents a cluster*

Step-2: Take two clusters that are closest to each other, and merge them to make a single cluster. Now there are N-1 clusters left, as shown in *Figure 3.13:*

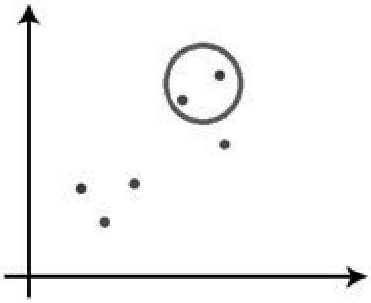


***Figure 3.13**: The merging of two clusters is shown through the circle*

Step-3: Similar to step-2, choose another two nearest clusters and merge them to form one cluster. Now there are N-2 clusters left, as shown in *Figure 3.14*:

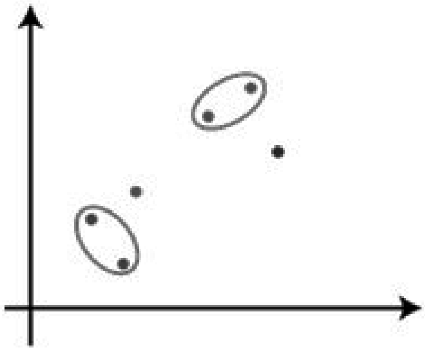
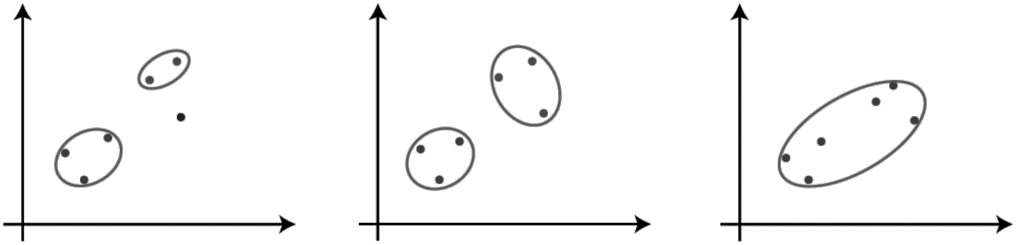


***Figure 3.14**: The merging of two clusters is shown through the oval*

Step-4: Repeat step 3 until one cluster is left. So, we will get the following clusters. Consider the images shown in *Figure 3.15*:



(a)

(b)

(c)

***Figure 3.15**: Step by step merging of clusters (a),(b),(c)*

Step-5: After combining all the clusters into one big cluster, a dendrogram has to be created. The dendrogram is used to divide the clusters as per the requirement of the problem. A dendrogram of the above-mentioned problem is shown in *Figure 3.16*.

Measuring the distance between two clusters

As we know, for hierarchical clustering, the closest distance between the two clusters is important. There are different ways to measure the distance between two clusters and these methods establish the rules of clustering and are called linkage methods. Some important linkage methods are illustrated below:

Single Linkage: The shortest distance between the two closest points of two different clusters is considered as a single linkage. A pictorial representation of a single linkage is shown in the below image:

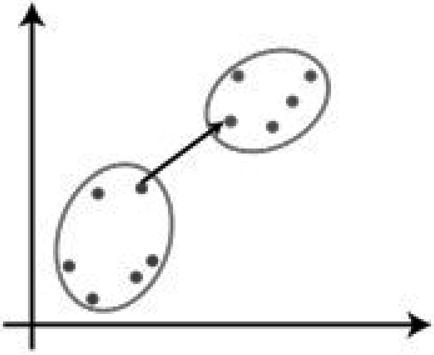


Figure 3.16: Single linkage

Complete Linkage: The farthest distance between the two points of two different clusters is considered as a complete linkage. This method is popular than the single linkage method because it makes a tighter cluster. A pictorial representation of a complete linkage is shown in the following image*:*

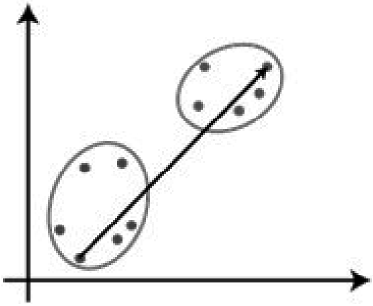


Figure 3.17: Complete linkage

Average Linkage: In this linkage method, the average distance between two clusters has to be calculated. It is also a very popular method of linkage.

Centroid Linkage: In this linkage method centroid of each cluster has to be calculated and then the distance between the centroids has to be calculated. A pictorial representation of a centroid linkage is shown in the following image:

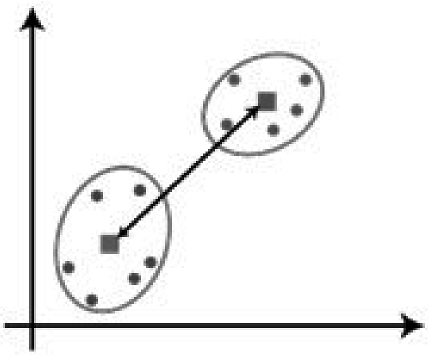


Figure 3.18: *Centroid linkage*

From the above-mentioned linkage methods, any method can be applied to calculate the distance between two clusters as per the requirement of business and type of problems.

The dendrogram in hierarchical clustering

In hierarchical clustering, a dendrogram represents a tree-like structure of clusters. The main use of the dendrogram is to store each step performed by the hierarchical clustering algorithm. In a dendrogram plot, the x-axis represents all datasets, and the y-axis represents the Euclidean distance between the data points of all datasets.

The functioning of the dendrogram can be illustrated through the following diagram:

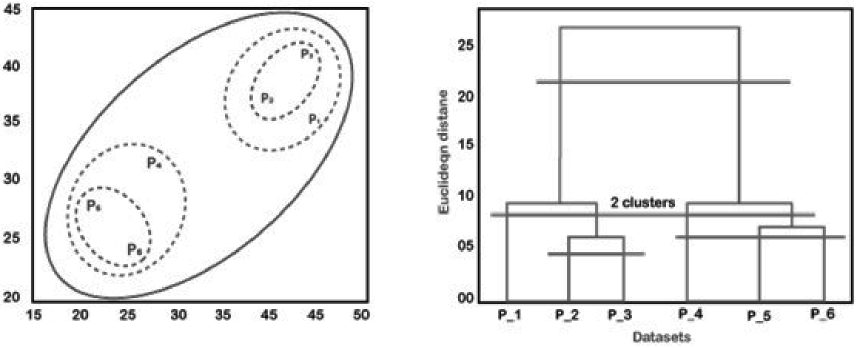


Figure 3.19: *(a) cluster (b) dendrogram*

The left part of the above-shown diagram shows the clusters created through the agglomerative clustering algorithm, and the dendrogram of corresponding clusters has been shown in the right (b) part of *Figure 3.19*.

Creating Dendrogram

A step-by-step process to create a dendrogram of the above-mentioned problem is discussed below:

I. As per the above-discussed problem, initially the points having the shortest distance i.e. point P2 and point P3 has been clubbed to create a cluster, and simultaneously their dendrogram is created that is having a rectangular shape by joining points P2 and P3, and the height of the rectangle is equivalent to the Euclidean distance between P2 and P3.

II. In the further step, the points P5 and P6 have been clubbed into a cluster, and the equivalent dendrogram is created. The height of this dendrogram is higher than the previous dendrogram of points P2 and P3 because the Euclidean distance between the points P5 and P6 is a little bit more than the points P3 and P2.

III. The same process has been repeated to draw two new dendrograms clubbing points P1, P2, and P3 in one dendrogram and points P4, P5, and P6 in another dendrogram. The final dendrogram that integrates all the data points is finally formed (as shown in *Figure 3.19(b)*). As per our requirements, we can cut the dendrogram tree-structure at any stage.

An example of hierarchical clustering

Example 2: Perform the hierarchical clustering using single linkage and complete linkage for the following dataset and plot the dendrogram for both the methods.

```
Dataset {8, 11, 21, 29, 36};
```

Solution: Step-1 visualize the data set:

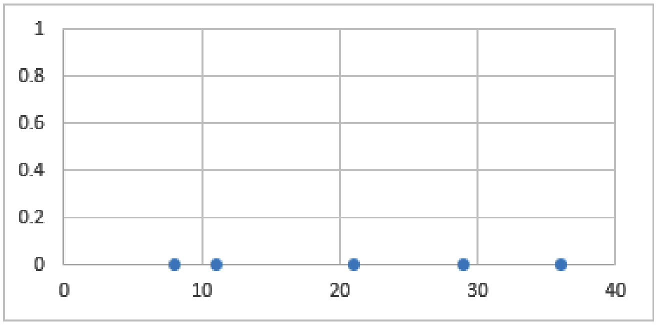


***Figure 3.20**: data visualization*

As we can see, it’s linear one-dimensional data.

Let's start to solve it first through the single-linkage method of agglomerative hierarchical clustering.

Using linkage method: In this method, two objects with the least distance will be merged in a cluster. In each step, two clusters with the least distance will be merged:

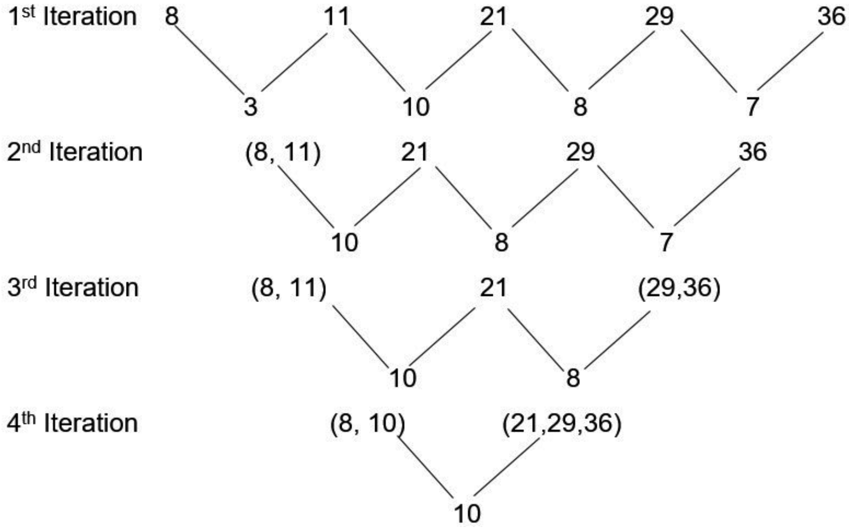


Figure 3.21: Single-linkage method of agglomerative hierarchical clustering

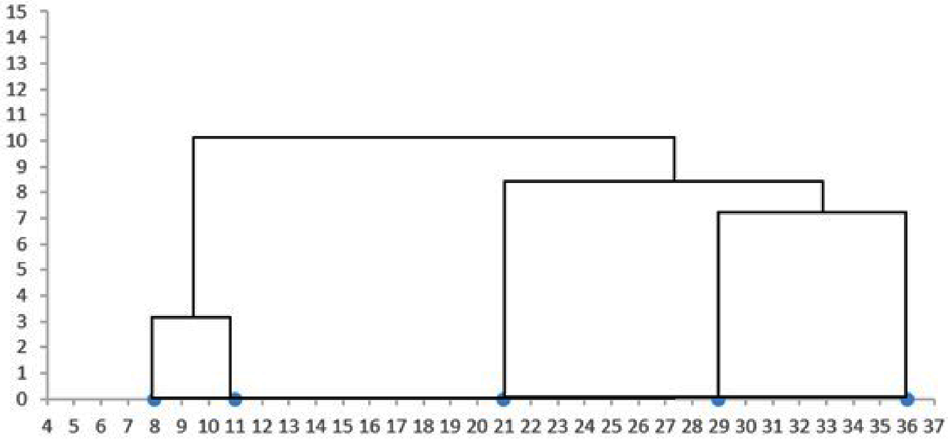


Figure 3.22: The dendrogram of the single-linkage method of agglomerative hierarchical clustering.

By using single linkage two clusters are formed :

Cluster 1 : (8, 11)

Cluster 2 : (21, 29, 36)

Using the complete linkage method: In each step, two clusters providing the smallest maximum pairwise distance will be merged:

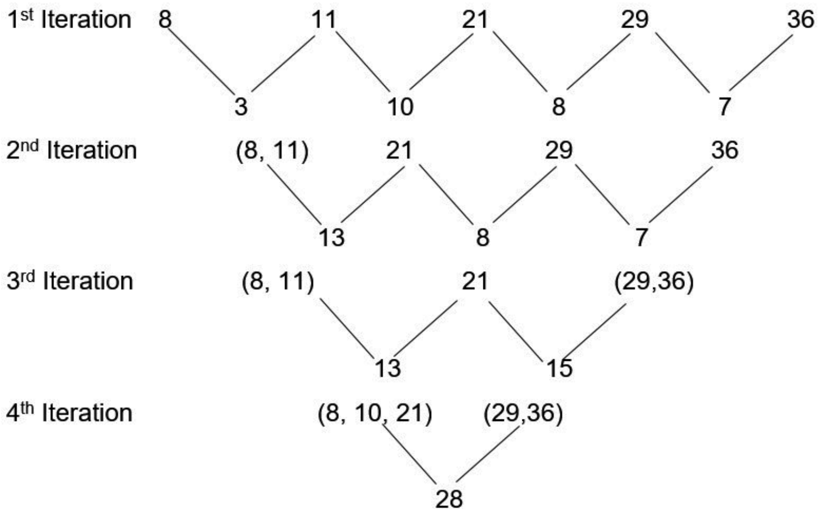


__Figure 3.23__: The complete-linkage method of agglomerative hierarchical clustering draw dendrogram:

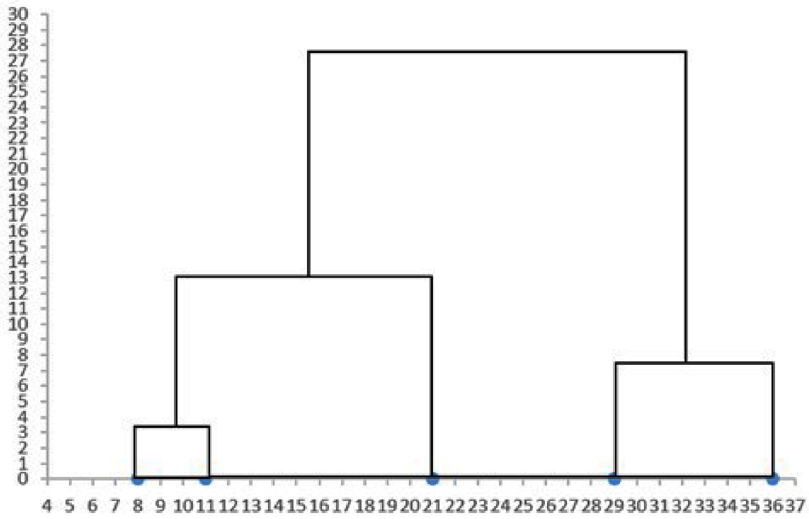


__Figure 3.24__: The dendrogram of the complete-linkage method of agglomerative hierarchical clustering

Using complete linkage two clusters are formed:

Cluster 1 : (8,11,21)

Cluster 2 : (29,36)

Hierarchical clustering is often used if a hierarchy is required by the application, like taxonomy design. In terms of storage and computation, such clustering techniques are expensive.

Association rule learning

Association rule learning is another technique of unsupervised learning. This technique is used to find the relation, association, and/or dependencies between the data items. This technique uses various rules to find useful relations between the dataset in a database. Association rule learning is beneficial for the product-based business organization because mapping relations between different products can improve their sale and they can earn more profit. Web usage mining, market basket analysis, and continuous production analysis are some important applications of association rule mining.

Market basket analysis is a popular application of association rule mining. Almost in all the supermarkets, this approach is used to improve sales. We can understand this by observing the product display strategy of a supermarket. All the products that can be purchased together are put in the same or nearby place. For example, if a customer purchases bread, he will possibly buy butter, eggs, or milk as well, so these items are thus stored inside or often near a shelf. Consider the following diagram:

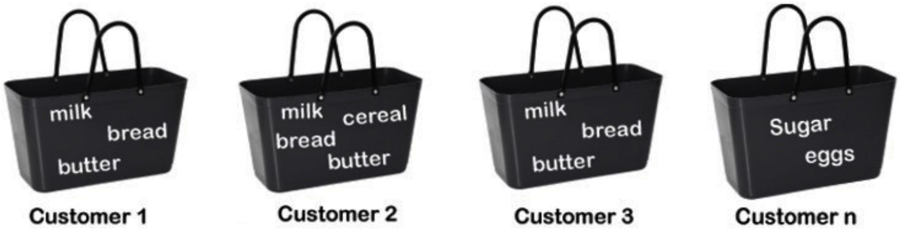


***Figure 3.25**: Market basket analysis*

There are three types of association rule learning algorithms:

1. Apriori
2. Eclat
3. F-P Growth Algorithm

Before learning these algorithms, we will understand the working of association rule learning.

Working of the association rule learning

Working of the association rule learning is based on the if and else statements, like:

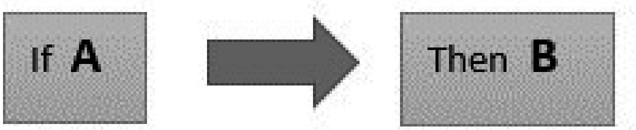


***Figure 3.26**: Association rule learning*

In this technique, the statement 'If' is referred to as antecedent, and the 'Then' statement is referred to as consequent. The relationship or association between two items, that can be found through the If-Then relation is called the single cardinality relationship. If the number of items increases. It's all about establishing rules and as the number of things increases, it also increases cardinality. So, there are many metrics to quantify the associations between thousands of data objects.

Below, these metrics are given:

- Support
- Confidence
- Lift

Let's grasp each of them:

Support

It can be defined as a frequency of elements X that appears in a dataset in a fraction of transaction T. It can be express through the following equation:

$$Support(X) = \frac{Freq\ (x)}{T}$$

Confidence

Confidence is a property to identify the probability of trueness of the association rules. In another word, we can say that the confidence property tells us how often the elements X and Y take place together in the database when the frequency of element X is given. It can be calculated through the following equation:

$$Cofidence = \frac{Freq\ (X,Y)}{Freq\ (X)}$$

Lift

Lift is another property of the association rule which is used to understand the strength of association rule. It can be calculated by finding the ratio of the observed support of elements X and Y and the expected support of elements X and element Y when they are independent. The mathematical representation of the lift property is shown through the following equation:

$$Lift = \frac{Supp\ (X,Y)}{Supp(X) \times Supp(Y)}$$

The value of lift property can be categorized into three ways:

- If Lift > 1, it signifies the degree of dependency up to which the element X and element Y is dependent.
- If Lift <1, it means that one element is harming another one or we can say item X is the replacement of item Y and vice-versa.
- If Lift= 1: it means the probability of the occurrence of antecedent (If) and consequent (Then) is independent of each other.

Apriori Algorithm

Apriori algorithm was proposed by *R. Agarwal* and *Shrikant* in the year 1994. Generally, this algorithm is used for market basket analysis. It helps to find the products that can be bought together by the consumer from the market. It also plays an important role in the healthcare sector to find the reaction of drugs on the patients.

The databases which contain the transactions are used by the Apriori algorithm to generate the association rules by analyzing the '*frequent itemsets*' available in the database. Association rules generated by the algorithm are used to find how weakly/ strongly two items are connected. This technique uses two approaches to find the 'frequent itemsets' from databases, one is **breadth-first search** (**BFS**) and another is **Hash Tree** (**HT**). Finding the '*frequent itemsets*' from databases is an iterative process.

Frequent Itemset

In a transactional database, the itemsets whose support value is greater than the predefined threshold or user-defined minimum support value is called '*frequent itemsets*'.

For example, in a database, two transactions T1 and T2 are such that:

$$T1 = \{1,3,4,5,6,7\} \text{ and } T2 = \{2,5,7\};$$

In these two transactions, 5 and 7 are the '*frequent itemsets*'.

In a transactional dataset, if P1 and P2 together is a frequent itemset then P1 and P2 individually should be frequent itemset.

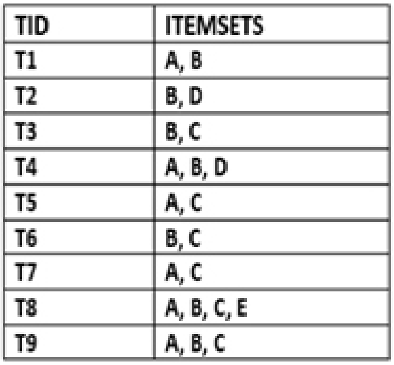
Steps of Apriori Algorithm

The Apriori algorithm uses the following steps:

Step 1	**Start**
Step 2	Input a transactional database
Step 3	Determine the *Support* value of the itemsets in a transactional database.
Step 4	Choose minimum *Support* and *Confidence* for threshold.
Step 5	From the transactions select all *Support* whose support value is higher than the minimum or threshold support.
Step 6	Find the rules for subsets having a higher '*Confidence*' value than the minimum or threshold value of '*Confidence*'.
Step 7	Perform sorting on the rules and arrange the rules in a lift or descending order.
Step 8	Stop.

Example of the apriori algorithm

Consider the following transactional dataset and find the frequent itemsets and generate association rules using the Apriori algorithm. Consider: Minimum support=2, Minimum confidence=50%.



TID	ITEMSETS
T1	A, B
T2	B, D
T3	B, C
T4	A, B, D
T5	A, C
T6	B, C
T7	A, C
T8	A, B, C, E
T9	A, B, C

Solution: In the first step, we will calculate the candidacy table (C1) and item list (L1) by counting individual items in the transactional dataset.

1. **Generate C1 and L1:**

   Create a table with the support (frequency) of each item:

List C1:

Item	Support_count
A	6
B	7
C	5
D	2
E	1

Now we will find a list L1 by removing the items from the list C1 which is having a support count less than the given minimum support.

List L1:

Item	Support_count
A	6
B	7
C	5
D	2

Now, the above list has the items whose support count is greater or equal to the minimum support value.

2. **Generate C2 and L2.**

To generate C2, we will take help of L1. We will generate C2 by creating pairs of items contained in list L1 with their count available in the actual transactional dataset.

List C2:

Item	Support_count
{A, B}	4
{A, C}	4
{A, D}	1
{B, C}	4
{B, D}	2
{C, D}	0

Now, we will generate list L2, by removing the items whose support count is less than the given minimum support.

List L2:

Item	Support_count
{A, B}	4
{A, C}	4
{B, C}	4
{B, D}	2

3. **Generate C3 and L3.**

   For C3, we will repeat the same process by taking three items in a set.

Item	Support_count
{A, B, C}	2
{A, B, D}	0
{A, C, D}	0
{B, C, D}	1

Now, we will create L3 by removing the items that are having a count less than the given minimum support count.

List L3:

Item	Support_count
{A, B, C}	2

Here, we can see list L3 is having only one set of items as {A, B, C}, so now there is no need to calculate C4 and L4.

4. **Find the association rules**

   To find the association rules, first, we need to create a table that will have all combinations of the itemset that can be created from itemset {A, B, C} with its confidence values. Confidence value can be calculated by using the following formula: sup(A^B)/sup(A), where 'sup' is the support value of that item. After that, the items that has less confidence value than the given minimum confidence value will be removed from the list and the remaining rules will be considered as strong association rules for the given problem.

The following table shows association rules and their confidence values:

Rules	Support_count	Confidance_value
A^B→ C	2	Sup{(A^B)^C}/sup(A^B)=2/4 = 0.5= 50%
A^C→ B	2	Sup{(A^C)^B}/sup(A^C) =2/4 = 0.5= 50%
B^C→ A	2	Sup{(B^C)^A}/sup(C^C)=2/4 = 0.5= 50%

A→ B^C	2	Sup{(A^(B ^C)}/sup(A)= 2/6=0.33=33.33%
B→A^C	2	Sup{(B^(A ^C)}/sup(B)= 2/7=0.28=28%
C→ A^B	2	Sup{(C^(A ^B)}/sup(C)= 2/5=0.4=40%

***Table 3.3**: Association rules and their confidence values*

For strong association rules, we will consider the rules that has a confidence value greater or equal to the given minimum confidence value i.e. 50%.

Strong association rule:

From the above-mentioned table rules A^B→ C, A^C→ B, and B^C→ A having the confidence value of 50%, i.e., equal to the given threshold of confidence value. Therefore these rules will be considered as strong association rules for the given problem.

Cons of the Apriori algorithm:

- This algorithm is simple and easy to understand.
- For large databases, the Join and Prune steps of this algorithm can be implemented easily.

Pros of the Apriori algorithm:

- As compared to other algorithms this algorithm works slowly.
- This algorithm scans the database multiple times therefore the overall performance of this algorithm can be reduced.
- The space complexity and the time complexity of this algorithm is very high. It's an order of 2D i.e. O(2D). whereas D is the horizontal width of the given database.

FP-Growth Algorithm

The FP-growth algorithm is another important algorithm of the association rule learning. In this algorithm, the word F-P means Frequent-Pattern. This algorithm represents the data in a tree-like structure that is called a frequent pattern tree. The main purpose of the frequent tree is fetching the most frequent pattern from the database. The relation between the item sets will be preserved by this tree structure. Frequent pattern reduces the search time therefore, we can say the FP-growth algorithm is an improved version of the Apriori algorithm.

Frequent Pattern (FP) Tree

The initial items of the database makes a tree-like structure which is called a **frequent pattern tree**. The main purpose of the tree is to find the most frequent patterns from the database. Items of an itemset are represented by the lower nodes of the FP tree, whereas, the root node of the tree has a NULL value. The association of the lower itemsets with the other itemsets of the tree must be maintained during the generation of that tree.

Steps of the FP-growth algorithm

The following steps are used to mine the frequent patterns from the transactional database through the FP-growth algorithm.

Step-1: Scan the transactional database to find the frequency/support count of each itemset of the database. (This step is similar to the first step of the Apriori algorithm)

Step-2: Build an FP-tree by creating a root node with the null value.

Step-3: Scan the database to examine the first transaction. During the examination of the first transaction, find the itemset that has the maximum count and place it on the top of the tree and the itemset with the lower-count will be placed next to it, and so on. The branches of the constructed tree will have the transections-itemset in descending order in counting.

Step-4: Examine the database for the next transaction. Arrange the itemsets in descending order. For repeated itemset of the transaction (i.e. was already present in the previous transaction), the branch of this transaction will share the collective prefix to the root node. This means in this transaction, the mutual itemsets are connected to the new node of the alternative item-set.

Step-5: Increase the itemset count as it happens in the transaction. Also, increase the count of the new node and the common by one, as these linked and generated conferring to transections.

Step-6: Mine the generated FP Tree. For this step, the lowest node along with its link will be examined first. The frequency patterns with length-1 will be represented by the lowest node of the tree. According to that, the path of the FP tree has to be traversed. These paths are called **conditional pattern bases**.

Step-7: Build a conditional FP-Tree: This is formed by the number of items in the path. In the conditional FP tree, the itemsets which meets the threshold support will only be considered.

Step-8: Generate frequent patterns: From the conditional FP (build in the previous step). Frequent Patterns will be generated.

An example of FP-growth algorithm:

Consider the following transactional dataset and find the frequent patterns and generate the association rules using the FP-growth algorithm.

Consider: Minimum support=50%, Minimum confidence=60%.

Sr. No.	Transections	Item list
1.	Tr1	It1, It2, It3
2.	Tr2	It2, It3, It4
3.	Tr3	It4, It5
4.	Tr4	It1, It2, It4
5.	Tr5	It1, It2, It3, It5
6.	Tr6	It1, It2, It3, It4

Solution:

Given: support-threshold = 50%

The total transaction is given 6 (Tr1 to Tr6);

So, the minimum support count threshold = 0.5 x 6= 3

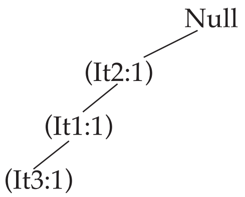
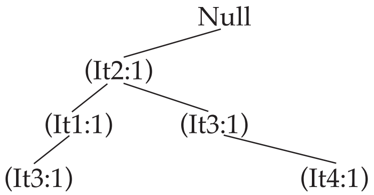
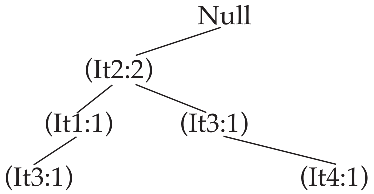
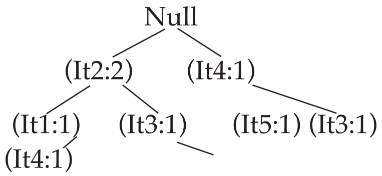
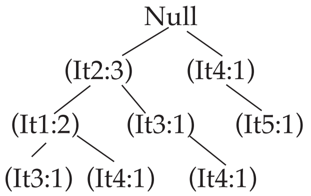
Step 1: Table for support count of items

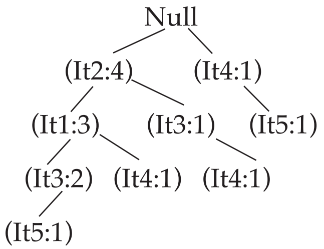
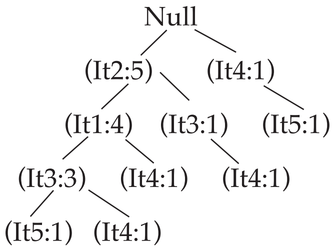
Item	Support_count
It1	4
It2	5
It3	4
It4	4
It5	2

Step 2: Table for sorted after applying the threshold of support count

Item	Count
I2	5
I1	4
I3	4
I4	4

1. Construct FP Tree

Steps	Operation	Action
Step 1:	Take a root node as Null.	*Null*
Step 2:	Scan the transaction Tr1: It contains 3 items with frequency such as, {It1:1}, {It2:1}, {It3:1}, from the above table we know that It2 is having the highest support count so It2 will be the child node of root. It1 will be linked to the It2 and It3 will be linked to the It1.	
Step 3:	Scan transaction Tr2: It contains It2, It3, and It4. Here It2 is connected to the root node, It3 is connected to It2, and It4 will be connected to It3. But this branch would share the It2 node as common as it is already used in Tr1.	
Step 4:	The count of It2 will be incremented by 1, as It3 becomes a child of It2 and It4 becomes a child of It3. So count become like that {It2:2}, {It3:1}, {It4:1}.	
Step 5:	Scan transaction Tr3: It4, It5. Connect It5 as a child of It4.	 Null (It2:2) (It4:1) (It1:1) (It3:1) (It5:1) (It3:1) (It4:1)
Step 6:	Scan transaction Tr4: It1, It2, It4. This transaction sequence will be followed as It2, It1, and It4. As we can see It2 is previously connected to root so its count will increase by 1. The count of It1 will also increment as it is connected to It2 in transaction Tr1. So it looks like: {It2:3}, {It1:2}, {It4:1}	 Null (It2:3) (It4:1) (It1:2) (It3:1) (It5:1) (It3:1) (It4:1) (It4:1)

Step 7:	Scan transaction Tr5: It1, It2, It3, It5. It will be considered in the following order: It2, It1, It3, and It5. So the count will be updated like {It2:4}, {It1:3}, {It3:2}, {It5:1}.	 Null (It2:4) (It4:1) (It1:3) (It3:1) (It5:1) (It3:2) (It4:1) (It4:1) (It5:1)
Step 8:	Scan transaction Tr6: It1, It2, It3, It4. It will be considered in the following order: It2, It1, It3, and It4. So the count will be updated like {It2:5}, {It1:4}, {It3:3}, {It4 1}.	 Null (It2:5) (It4:1) (It1:4) (It3:1) (It5:1) (It3:3) (It4:1) (It4:1) (It5:1) (It4:1)

2. Mining of FP-Tree:

1. As we know, that item It5 is the lowest node and it doesn't satisfy the threshold i.e. the minimum support count so it will not be considered and will be deleted.

2. After It5 the next item is It4, and it arises in two branches, first (It2, It1, It4:1) and second {It2, It3, It4:1}. Consider I4 as a suffix and calculate prefix path that will be {It2, It1, It3:1} and {It2, It3:1} it can be considered a conditional pattern.

3. The conditional pattern base is considered a transaction database, an FP-tree is constructed. This will contain {It2:2, It3:2}, It1 is not considered as it does not meet the min support count.

4. All arrangement of the Frequent-Patterns will be generated through this path: {It2,It4:2}, {It3,It4:2}, {It2,It3,It4:2}

5. For It3, the prefix path would be: {It2,It1:3},{It2:1}, this will generate a 2 node FP-tree : {It2:4, It1:3} and frequent patterns are generated: {It2,It3:4}, {It1:It3:3}, {It2,It1,It3:3}.

6. For It1, the prefix path would be: {It2:4} this will generate a single node FP-tree: {It2:4} and frequent patterns are generated: {It2, It1:4}.

Item	Conditional Pattern Base	Conditional FP-tree	Frequent Patterns Generated
It4	{It2,It1,It3:1},{It2,It3:1}	{It2:2, It3:2}	{It2,It4:2},{It3,It4:2}, {It2,It3,It4:2}
It3	{It2,It1:3},{It2:1}	{It2:4, It1:3}	{It2,It3:4}, {It1:It3:3}, {It2,It1,It3:3}
It1	{It2:4}	{It2:4}	{It2,It1:4}

The diagram given below in *Figure 3.27*, depicts the conditional FP tree associated with the conditional node It3:

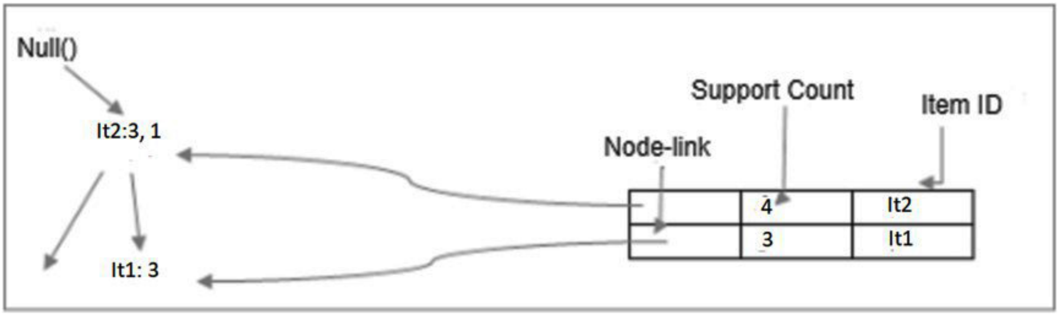


***Figure 3.27**: Conditional FP-tree*

Advantages:

1. The FP growth algorithm is faster than the Apriori algorithm because the Apriori algorithm scans the complete dataset in each iteration whereas FP Growth scans the dataset only twice.
2. For short and long frequent patterns, this technique is efficient and scalable.
3. This technique uses a compact version of memory to store datasets.
4. This technique doesn’t perform the pairing of nodes (data items); therefore it is faster than the other technique.

Disadvantages:

1. Building an FP-tree is a clumsy and difficult task.
2. The use of this technique may be expensive than the Apriori technique.
3. For a very large dataset, shared memory may not be suited for this algorithm.

Difference between Apriori and FP-Growth

The following are difference between Apriori and FP-Growth based on parameters:

Parameters	Apriori	FP-Growth
Pattern Generation	It performs the pairing of items and generates patterns of singletons, pairs, and triplets.	It constructs FP-tree to generate patterns.
Candidate Generation	It has a process of candidate generation.	It doesn't have a process of candidate generation.
Process	The process is slower. For a large number of items, runtime increases exponentially.	The process is faster. For a large number of items, runtime increases linearly.
Memory Usage	Candidate combinations are saved in memory.	Compact datasets are saved in memory.

Applications of the association rule learning

Association rule learning has a large number of applications in the field of data mining and machine learning. Some popular applications of association rule learning are given as follows:

In **Medical diagnosis**: Association rule learning can be used to find the probability of illness due to a particular disease. It can help doctors to choose the right direction of treatment so that the patients can be cured easily.

Determining Protein Sequence: Association rule learning helps to identify the protein structure from a big amount of genome data, which helps to synthesize the artificial protein.

Market Basket Analysis: Finding an association between the different items is called marker basket analysis. Various techniques of the association rule learning can be used for market basket analysis. It is widely used by big retailers to improve their sales. Therefore, we can say that the market basket analysis is an important and popular application of association rule learning. Other than the application mentioned above, there are some more applications of association rule learning such as catalog design, loss-leader, and many more.

Probabilistic clustering

In this clustering technique, the assignment of data points to clusters is soft, i.e. the membership of a data point in a cluster is having a probabilistic value. Therefore, the probabilistic clustering is also known as soft clustering. Let us understand the concept with the help of an example:

Let us consider three clusters as shown in a scatter plot in *Figure 3.28*. All these three clusters have been denoted through different colors i.e. cyan, blue, and green:

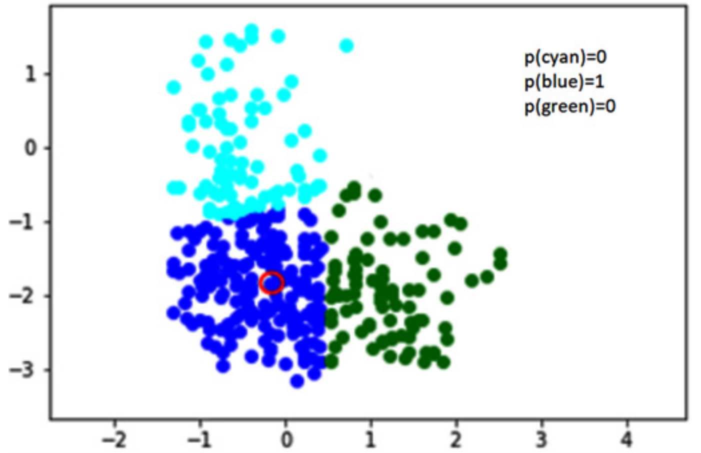


***Figure 3.28**: Three clusters as shown in a scatter plot*

Now consider a data point that has been represented through the red circle in the given plot, we can say that the probability of belonging to this data point to a blue cluster is 1, to a green cluster is 0, and to a cyan cluster is also 0.

Now consider another plot is shown in *Figure 3.29*. This plot looks towards a data point somewhere between the cyan and blue color that has been circled with a red color. Here, 80% portion of that data point belongs to a cyan cluster and 20% portion belongs to a blue cluster. So we can say the probability of belonging to this data point to a blue cluster is 0.2, to a cyan cluster is 0.8, and to a green cluster is 0.0.

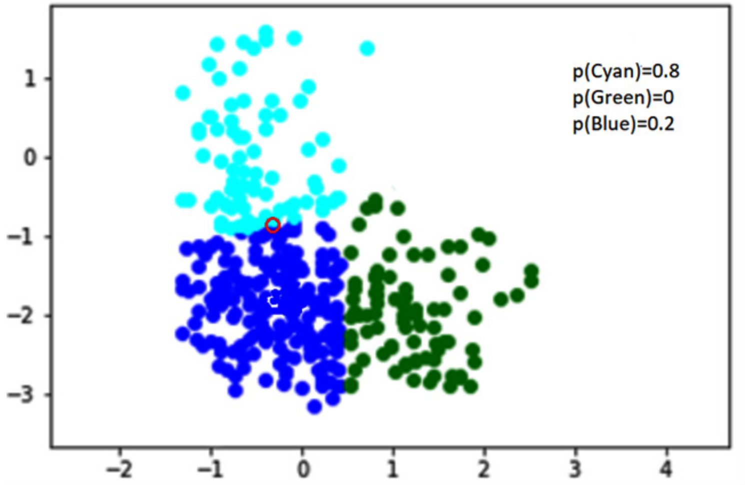


***Figure 3.29**: Three clusters as shown in a scatter plot*

The most popular probabilistic clustering technique is the Gaussian Mixture Models. The Gaussian distribution model uses the soft clustering approach for distributing the points in different clusters.

Gaussian Distribution

In statistics, the normal distribution is an important type of probabilistic method. It is suitable for many normal spectacles like blood pressure, IQ score, height, and measurement errors. In this statistic, the median, mode, and mean of the data are equal. The plot of the curve is symmetric around the μ(mean). Correctly, half of the values are on the right side of the center μ and the rest half values are to the left side of the center. The curve looks like a bell curve as shown in *Figure 3.30*:

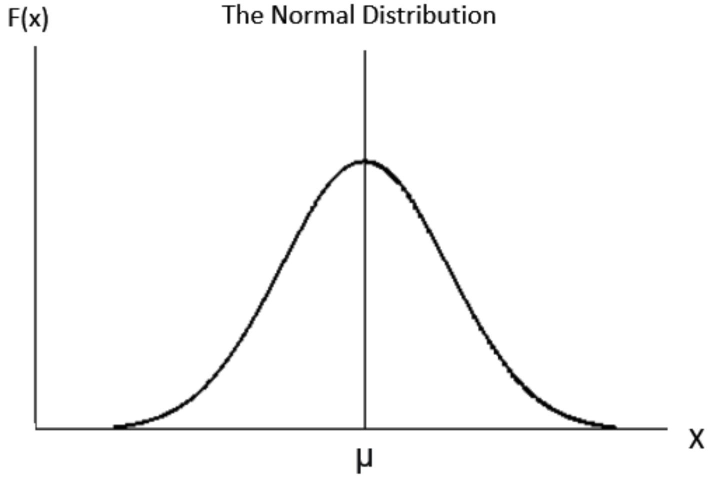


Figure 3.30*: Normal distribution*

The area under the bell curve is 1, this a type of probabilistic distribution is known as **Gaussian Distribution**.

Gaussian Mixture Models (GMMs)

In the Gaussian Mixture Model, it is assumed that there are a variety of Gaussian distributions in a dataset, every Gaussian distribution is assumed as a cluster. Moreover, a Gaussian Mixture Model aims to group the data points associated with a single distribution.

For example, assume there are 3 Gaussian distributions say Gd1, Gd2, and Gd3, having the definite mean μ1, μ2, μ3, and variance σ1, σ2, σ3 respectively. So a Gaussian Mixture Model can be used to find the probability of every data point (of a given dataset) belonging to each of these Gaussian distributions.

A mixture of some Gaussian distributions with a different mean (μ) and variance (σ2) is shown below in *Figure 3.31*. The value of variance σ represents the spread of the bell curve:

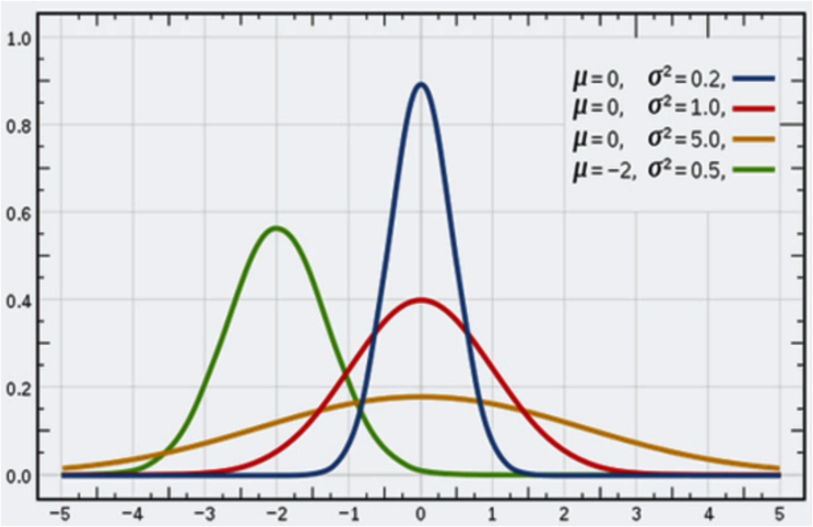


***Figure 3.31**: A mixture of some Gaussian distributions with a different mean (μ) and variance (σ2)*

For one-dimensional data probability, distribution function a can be represented via a function given as follows:

$$p(x) = \sum_{i=0}^{k} \emptyset_i N(x|\, \mu_i, \sigma_i)$$

$$N(x|\, \mu_i, \sigma_i) = \frac{1}{\sigma_i\sqrt{2\pi}} \exp\left(-\frac{(x-\mu_i)^2}{2\sigma_i^2}\right)$$

$$\sum_{i=0}^{k} \emptyset_i = 1$$

The above-mentioned function is true for only one-dimensional data. For multidimensional data it can be represented by the following function:

$$p\,(\vec{x}) = \sum_{i=1}^{k} \emptyset_i \, N(\vec{x} \mid \vec{\mu_i}, \Sigma_i)$$

$$N(\vec{x} \mid \vec{\mu_i}, \Sigma_i) = \frac{1}{\sqrt{(2\pi)^k \, |\Sigma_i|}} \exp\left(-\frac{1}{2}\,(\vec{x} - \vec{\mu_i})^T \, \Sigma_i^{\;-1} \, (\vec{x} - \vec{\mu_i})\right)$$

$$\sum_{i=0}^{k} \emptyset_i = 1$$

Where x_bar is the input vector, μ_bar is the 2D mean vector, and Σ is the 2×2 covariance matrix. The shape of the curve would be defined with the help of the covariance matrix. For a d-dimensional dataset, x_bar and μ_bar are the vectors of length d and Σ would be the d x d covariance matrix. For a dataset with k-Gaussian distributions and d-features, each cluster (Gaussian Distribution) will have a certain mean and variance matrix but how can these values be assigned for each Gaussian Distribution?

These values are estimated using a process called **Expectation-Maximization**. This technique will help to understand the Gaussian Mixture Model.

Expectation-Maximization

To find the proper model parameters, **Expectation-Maximization** (**EM**) is a mathematical algorithm. When the dataset has incomplete and/or missing values, we use EM. Such missing or incomplete variable is referred to as latent variables.

To find the right model of parameters, a statistical algorithm named EM is used. The EM is used when data is has latent variables, in other words, we can say that EM will be used when the data has missing or incomplete values. Because of these latent variables, it's hard to decide the proper model parameter.

Expectation-Maximization aims to determine the optimal values for these variables using the actual data and then finds the parameters of the model. Depending on this model, parameters of the values of latent parameters can be updated.

The process of Expectation Minimization has two basic steps:

- **E-step**: In this step, to approximate (guess) the values of the missing variables, the available data is used.
- **M-step**: The entire data is used to update the parameters based on the approximate values produced in the E-step.

Expectation-Maximization is a basic process that is used by many algorithms including the Gaussian Mixture Model of clustering. Now, let us understand how the process of EM can be applied to data for the **Gaussian Mixture Model** (**GMM**).

Expectation Maximization in GMMs

For a dataset choose a number k, which represents the number of clusters that is, the number of Gaussian distributions. Let’s assume µ1, µ2, .. µk and Σ1, Σ2, .. Σk are the mean and covariance of these k Gaussian distributions respectively. Let’s say, we need to assign k a number of clusters. We will consider another parameter Πi, that represents the density i.e. the number of points for distribution.

To define the Gaussian distribution, we need to find these parameters. As the value of k has already been defined, the values of mean, covariance, and density have to be chosen randomly. Now, we will apply the E-step and M-step.

E-step

For each xi: calculate the probability that xi belongs to c1, c2, c3,….ck. It can be calculated with the help of the following formula:

$$r_{ic} = \frac{\textit{Probability } x_i \textit{ belongs to } c}{\textit{sum of probability } x_i \textit{ belongs to } c_1, c_2, \dots, c_k}$$

$$r_{ic} = \frac{\pi_c \, N\,(x_i;\ \mu_c, \Sigma_c)}{\Sigma_{c'}\, \pi_{c'}\, N\,(x_{i'};\ \mu_{c'}, \Sigma_{c'})}$$

M-step

After the E-step, a feedback operation is performed to update the values of Π, µ, and Σ. It can be performed in the following ways:

1. The updated density can be defined by calculating the ratio of points in the cluster and the total number of points in the dataset.

$$\pi = \frac{\textit{Number of points assigned to cluster}}{\textit{Total number of points}}$$

2. The covariance and mean can be updated with the help of the following formula:

$$\mu = \frac{1}{\substack{\textit{number of points} \\ \textit{assigned to cluster}}} \Sigma_i \, r_{ic} \, x_i$$

$$\mu_c = \frac{1}{\substack{\textit{number of points} \\ \textit{assigned to cluster}}} \Sigma_i \, r_{ic}(x_i - \mu_c)^T \, (x_i - \mu_c)$$

Based on the updated values generated from the above-mentioned step, the new probability will be calculated for each data point and the same process is repeated iteratively to maximize the log-likelihood function. In the k-mean algorithm, only the mean value is used to update the centroid. Whereas, in GMM both mean and variance is considered.

An example of Gaussian Mixture Model

Due to the statistical complexity, it is difficult to show the process of GMM mathematically. Therefore, the implementation of GMM has been shown through the following Python code:

```
#Loading and Plotting of dataset
# A Sample dataset file ‘Sample_GMM.csv’ is used here.
import pandas as pd
data = pd.read_csv('Sample_GMM.csv')
plt.figure(figsize=(7,7))
plt.scatter(data["Weight"],data["Height"])
plt.xlabel('Weight')
plt.ylabel('Height')
plt.title('Data Distribution')
plt.show()
```

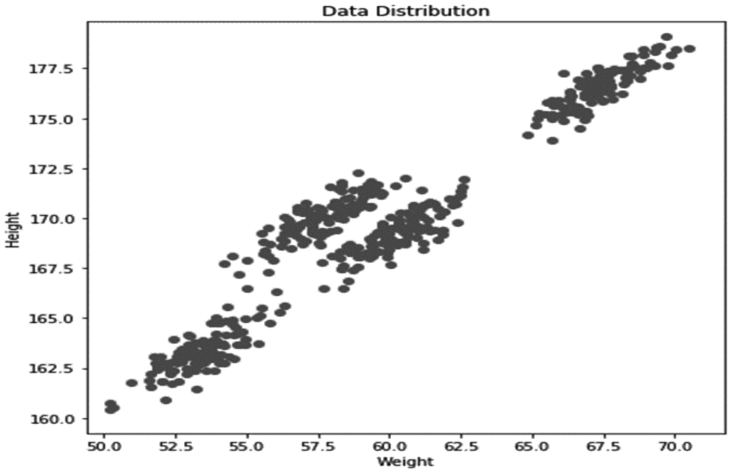


***Figure 3.32**: Data representation*

Let's implement k-mean first to show the difference between k-mean and GMM

# training k-means model

```
from sklearn.cluster import KMeans
kmeans = KMeans(n_clusters=4)                  //here k=4 for k-mean.
kmeans.fit(data)
```

# predictions from kmeans

```
pred = kmeans.predict(data)
frame = pd.DataFrame(data)
frame['cluster'] = pred
frame.columns = ['Weight', 'Height', 'cluster']
```

# plotting results

```
color=['blue','green','cyan', 'black']
for k in range(0,4):
    data = frame[frame["cluster"]==k]
    plt.scatter(data["Weight"],data["Height"],c=color[k])
plt.show()
```

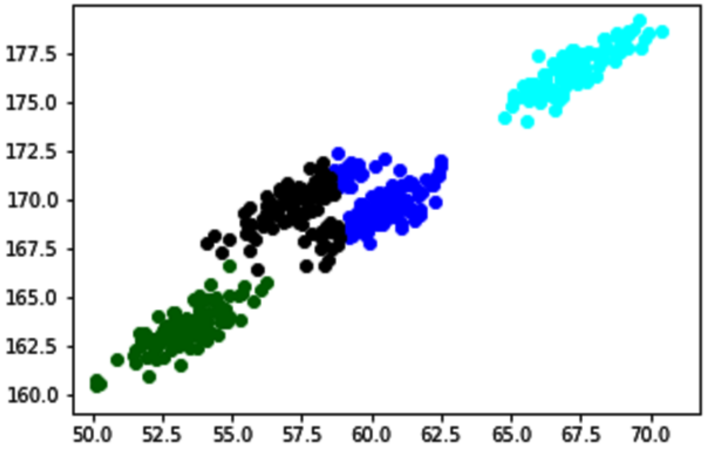


Figure 3.33: Output of k-mean technique

Here, we can see that the k-mean model struggled to classify the proper clusters. Even if the data flow is elliptical, k-means attempted to construct a circular cluster. This is the drawback of the k-mean algorithm.

Now, let's create a Gaussian Mixture Model on the same data to see if the result can be improved:

```
# Training Gaussian Mixture Model
from sklearn.mixture import GaussianMixture
gmm = GaussianMixture(n_components=4)
gmm.fit(data)


#Predictions from GMM
labels = gmm.predict(data)
frame = pd.DataFrame(data)
frame['cluster'] = labels
frame.columns = ['Weight', 'Height', 'cluster']
color=['blue','green','cyan', 'black']
for k in range(0,4):
    data = frame[frame["cluster"]==k]
    plt.scatter(data["Weight"],data["Height"],c=color[k])
plt.show()
```

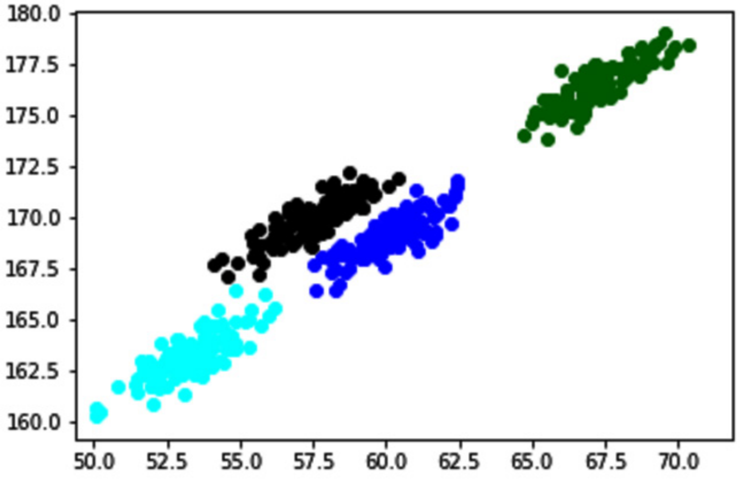


***Figure 3.34**: Output of GMM*

Brilliant! These are precisely the clusters that we had wished for. GMM has given an improved result than the k-means algorithm.

Summary

The complete chapter has been summarized in the following points:

- Unsupervised learning is a methodology for machine learning, where the model does not need to be supervised.
- Almost all types of the patterns can be identified through unsupervised machine learning.
- Association and clustering are the two types of unsupervised machine learning.
- Exclusive, agglomerative, overlapping, and probabilistic are the four methods of clustering.
- Hierarchical clustering, k-means clustering, and the Gaussian Mixture Model are some important types of clustering.
- For a large dataset, association rules allow us to create associations between data objects.
- Labeled data is used to train the model in supervised learning, whereas in unsupervised learning unlabeled data is used to train the model.
- In a dataset that is valuable for detecting suspicious transactions, anomaly detection may uncover essential data points.
- The greatest downside to the unsupervised learning is that the detailed knowledge about data sorting cannot be accessed.

Exercise (MCQs)

1. **Technique used in unsupervised learning accepts ______________________ ____________ , discovers new patterns of information that were previously unknown or undetected.**
   a) Unlabeled data items
   b) Labeled data items
   c) Classified data items
   d) Clustered data items

2. **Which of the following is/are the application/s of unsupervised learning?**
   a) Clustering
   b) Anomaly detection
   c) Neural networks
   d) All of the above

3. **Choose a correct statement regarding clustering:**
   a) Clustering is an unsupervised ML technique
   b) Clustering is a supervised ML technique
   c) Labeled data is used for the clustering technique
   d) All of the above are correct

4. **In the k-mean clustering algorithm k is a:**
   a) Positive integer
   b) Negative integer
   c) Random number
   d) Complex number

5. **The hierarchical clustering is an example of:**
   a) Exclusive partitioning
   b) Agglomerative clustering
   c) Overlapping clustering
   d) Probabilistic clustering

6. **The k-mean clustering is an example of:**
   a) Exclusive partitioning
   b) Agglomerative clustering
   c) Overlapping clustering
   d) Probabilistic clustering

7. **The shortest distance between the two closest points of two different clusters is considered as:**
   a) Single linkage
   b) Complete linkage
   c) Average linkage
   d) Centroid linkage

8. **The farthest distance between the two points of two different clusters is considered as:**
   a) Single linkage
   b) Complete linkage
   c) Average linkage
   d) Centroid linkage

9. **In a transactional database, the itemsets whose support-value is greater than the predefined threshold or user-defined minimum support value is called:**
   a) Frequent itemsets
   b) Average itemsets
   c) Regular itemset
   d) Poor itemsets

10. **Probabilistic clustering is also known as:**
    a) Soft clustering
    b) Hard clustering
    c) Lenient clustering
    d) Overlapping clustering

Answers

1. **a**
2. **d**
3. **a**
4. **a**
5. **b**
6. **a**
7. **a**
8. **b**
9. **a**
10. **a**

Short Answers Questions

1. Explain the difference between supervised and unsupervised learning.
2. Define clustering also explain the difference between clustering and classification.
3. What are the various applications of the association rule learning?
4. Explain the difference between the single linkage and complete linkage methods.
5. What are the various advantages and disadvantages of the Apriori algorithm?

Long answers questions

1. How to measure the distance between two clusters? Explain its various methods in detail.
2. Write the various steps of the FP-growth algorithm. Explain in detail.
3. Explain the step-by-step working of the agglomerative hierarchical clustering.
4. Define GMM. Explain the working of the Gaussian mixture model in detail.
5. Consider the following transactional dataset and find the frequent patterns and generate the association rules using the FP-growth Algorithm.

Sr. No.	Transactions	Item list
1.	Tr1	It1, It2, It3
2.	Tr2	It2, It3, It4
3.	Tr3	It4, It5
4.	Tr4	It1, It2, It4
5.	Tr5	It1, It2, It3, It5
6.	Tr6	It1, It2, It3, It4

Consider: Minimum support=50%, Minimum confidence=60%.

CHAPTER 4

Introduction to Statistical Learning Theory

Statistical learning was introduced in the 1960s and during that time it was a purely theoretical concept of analyzing the various problems using the available datasets. In the 1990s a new type of learning, the algorithm was introduced using the theoretical concept of statistical learning and it became popular with the name support vector machine. This new technique gave a new direction to statistical learning and since then statistical learning is not only a theoretical approach for analysis but it is also a tool for creating an algorithm for handling and estimating multidimensional function. Statistical learning has a major role in different area such as industry, medical science, and other finance-related industries. Now we will discuss some examples of statistics learning problems:

- Predicting if the patient who is hospitalized due to a heart attack will get a second heart attack or not. This prediction is based on the patient's age, gender, past clinical history, diet, etc. and so on.
- Predicting the stock market price in six months from now according to the past performance and economic status of the company.
- Determining the numbers in a handwritten ZIP code, from a digitized image.
- Estimation of the glucose level in the blood of a patient.

- Detection of the risk of prostate cancer, brain tumor, and lung cancer according to the clinical reports of the patient.

Structure

In this chapter, we will discuss the following topics:

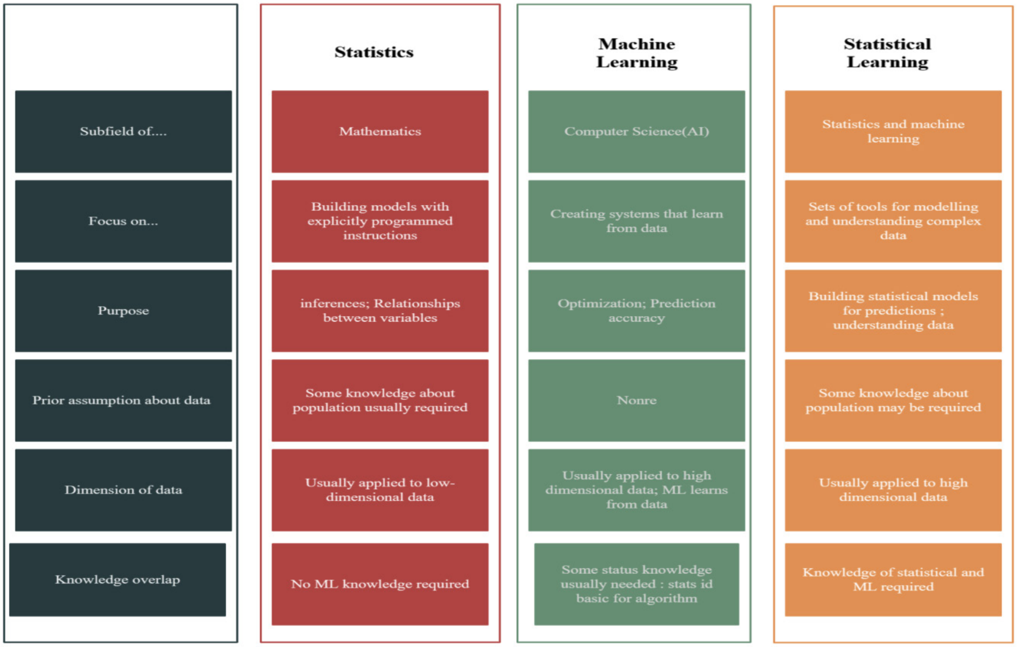
- Statistical learning concepts
- Feature selection
- Model selection
- Model evaluation
- Supervised machine learning
- Unsupervised machine learning

Objective

Statistical learning plays an important role in data mining, artificial intelligence, and other disciplinary areas of science and engineering. This chapter gives a general overview of the statistical theory and different methods of feature selection and many more.

Introduction to statistical learning

The term machine learning brings up images of robots, artificial intelligence, and flying cars, while statistics draws forth charts with bell curves and tracking of sports game outcomes. In reality, however, these two fields overlap significantly, as they both deal with the analysis of data. Statistical modeling and machine learning can even be applied to similar situations and work together to provide solutions to a variety of questions. Statistical learning is a combination of statistics and machine learning as shown in the *Figure 4.1*:



	Statistics	Machine Learning	Statistical Learning
Subfield of....	Mathematics	Computer Science(AI)	Statistics and machine learning
Focus on...	Building models with explicitly programmed instructions	Creating systems that learn from data	Sets of tools for modelling and understanding complex data
Purpose	inferences; Relationships between variables	Optimization; Prediction accuracy	Building statistical models for predictions ; understanding data
Prior assumption about data	Some knowledge about population usually required	Nonre	Some knowledge about population may be required
Dimension of data	Usually applied to low-dimensional data	Usually applied to high dimensional data; ML learns from data	Usually applied to high dimensional data
Knowledge overlap	No ML knowledge required	Some status knowledge usually needed : stats id basic for algorithm	Knowledge of statistical and ML required

Figure 4.1: *Overview of statistical learning*

The concept of t-tests, ANOVA, p-values testing, and so on., belongs to classical statistics which was introduced almost a hundred years ago. These methods were designed at a time when mathematics was purely based on pen and paper and analysis of the data was done for the small samples only. But things have changed drastically and the computer has taken part everywhere in human lives. Computational speed and memory have become available, affordable, and accessible. Increased usage of the internet and information transformation became a regular practice for computer users but the one thing which has not changed are the classical statistical methods for the analysis of the data and its important tool for the researchers. Slowly many areas of industry migrated to an alternative statistical framework which is popular with the name "*Statistical learning*". Statistical learning is easily adopted by engineers, computer science users, and people related to the physics background. The statistical learning approach is more flexible compared to the classical statistical-based approach as there is more scope of learning in itself. The comparative representation of both approaches is represented in *Figure 4.2:*

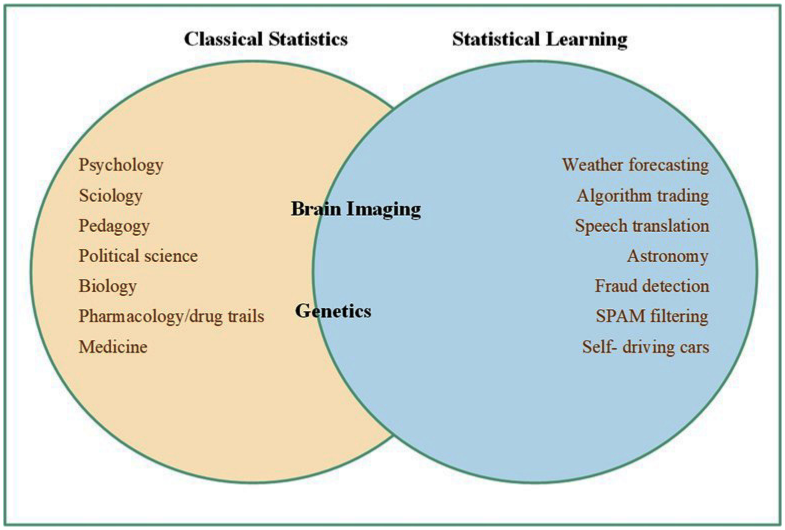


***Figure 4.2**: Classical Statistics Vs Statistical learning*

Statistical learning is a collection of tools that are used to understand different data. The tools can be categorized as supervised and unsupervised tools. Supervised statistical learning is used to build a statistical model for predicting the output using the given input datasets. Supervised statistical learning is used in various areas such as medical, business-related applications and public policies, and so on. In unsupervised statistical learning, input will be given but there is no supervised output and this type of learning is best suitable to understand the structure and the relationship between data. For a better understanding of statistical learning, let us take an example. Suppose a client hired a statistical advisor who gives new ideas to improve the sales of the product. To improve the sale of the product client has assigned some budget for the advertisement purpose and the mode of the advertisements is television, radio, and newspaper. Therefore, the statistical advisor has information about the budget and sales of the product in different cities which is displayed in *Figure 4.3*. It is difficult to increase the sale of the product directly. So the statistical advisor has to do some analysis. If we observe, the advertising budget can be controlled and distributed according to the requirements. It is visible that there is a relationship between the advertising budget and the sales of the product. If the statistical advisor gives proper advice about the distribution of the advertisement budget, there is a possibility of an increase in the sales of the product. So the main goal of the statistical advisor is to develop a model that can predict the sales of the product according to the assigned budget for different advertising media.

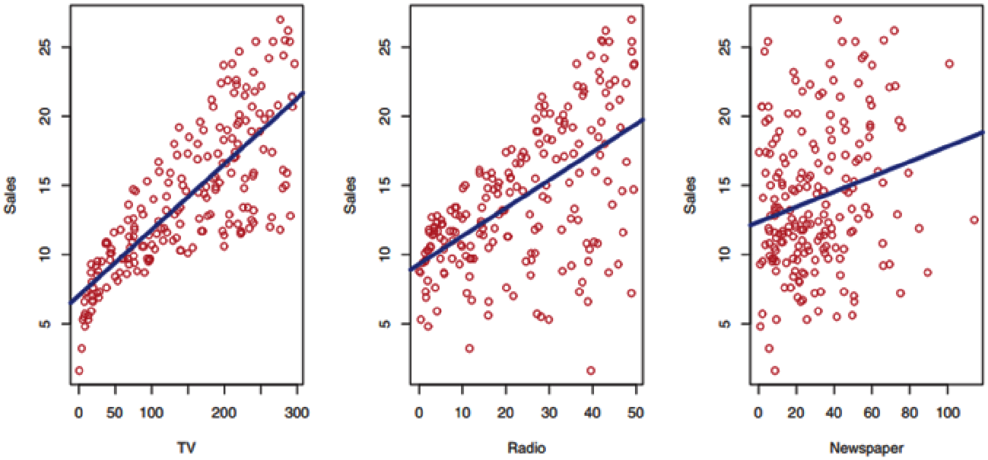


***Figure 4.3**: Example of Advertising dataset*

In the above example, the advertising budget is an input variable which is denoted by X and the sales information is an output variable which is denoted by Y. The input variable is also called features, independent variable, predictors, and so on. The output variables are sometimes referred to as dependent variables or response variables, and so on. To distinguish between the different input variables, subscripts could be used such as X1 can represent the budget of TV advertisement. In the same way, X2 and X3 can be used to represent the budget of radio and newspaper advertisements.

Suppose there is a relationship between X and Y then it can be represented as:

$$Y=f(X) + € \quad (1)$$

Here f is an unknown function that gives systematic information.

€ represents the random error and it is independent of X with means value zero.

Estimation of unknown function f

f is used to estimate either prediction or inference.

Prediction

In some scenarios, there is a possibility that input sets are available but still it is difficult to obtain the output. Therefore, to estimate the value of Y the following can be used:

$$\hat{Y} = \hat{f}(X) \quad (2)$$

Where ꞙ represents the estimation of f and Ŷ is represents the resultant predicted value of Y. The accuracy of Ŷ for the prediction of Y always depends on two things: reducible error and irreducible error. It is always possible that ꞙ is not the perfect estimation of f and this imperfection may introduce error. The error can be reduced if we use appropriate statistical learning algorithms and the accuracy of can be improved. Therefore it is possible that the estimated response could be in this form Ŷ = f(X). But there could still be some error because Y is a function of € and it cannot be predicted by X. So the variability which is associated with € can also affect the accuracy of the predicted value and this type of error is known as an irreducible error and it is introduced by €.

Inference

If the value of X changes, it affects the value of Y. In that case, if we want to estimate the value of f, we need to determine the reassociation between X and Y. In that situation f will not be considered as a black box and we need to find the answer of following questions:

- Find the particular predicator which is associated with the dependent variable.
- Type of relationship between the predictor and the dependent variable.
- Can an identified relation be represented using a linear equation?
- Consider the above-discussed sales example and try to relate the possibilities and try to answer to the following questions:
- The type of media is contributing to sales.
- Find the name of the media which is generating the highest sales.
- The relationship between sales and TV advertisement.

This type of scenario falls into the inference category. Now we will discuss another example:

If a person wants to buy a new house, the prediction variables can be the availability of schools nearby, pollution, air quality, distance from the office, size of the house, gated community or an independent house, and so on. In this scenario, each variable affects the cost of the house and the most important thing is to find the value of each independent variable that affects the cost of the house. A person might be interested to find the extra cost of the house if it is east facing or river facing. This is an example of inference. If a person is simply interested to find that the cost of the house is undervalued or overestimated then this is an example of a prediction based problem.

Therefore, according to our estimation goal (prediction or inference), there are different ways of estimating the value of f. There are two different approaches: linear and non-linear. The linear model is easy and simple but sometimes doesn't produce accurate results. Non-linear modules produce accurate predictions for Y but the inference is a challenging task. Our main aim is to find a statistical learning method for the training data that can estimate the unknown function ƒ or mathematically we can say find a function such that Y= ƒ(X) for any observation of (X, Y). This task is categorized into two categories: parametric and non-parametric

Supervised verses unsupervised learning

Most of the statistical-based learning algorithms are categorized into two categories: supervised learning algorithms and unsupervised learning algorithms. The example which we discussed earlier in this chapter is a suitable example of supervised learning. It is referred to as supervised machine learning because the way an algorithm's learning process is done, it is a training data set or in other words, we can say that the process will be done under the supervision of the supervisor. The algorithm learns the prediction in the training process and the corrections will be done by the teachers. The learning process will continue until it will not achieve its acceptable performance. Supervised learning is further divided into two categories: regression and classification which we will discuss later in this chapter. There are some popular machine learning algorithms are classification, support vector machine and linear regression, and so on.

Unsupervised learning algorithms are those algorithms where you have input values or input data but there are no corresponding output variables. The main aim of the unsupervised learning technique is to understand the structure of the data or the distribution of the data and these algorithms help the learners to understand more about the data. In unsupervised learning, there is no concept of teachers. Algorithms have their own choice to learn and explore more about the data. Suitable examples of unsupervised learning are clustering and association problems. Some of the popular algorithms are k-means clustering, PCA, independent component analysis, and so on.

Regression Verses classification

Classification related problems are used to find the function which can divide the available dataset into different categories according to its parameters. In the classification process, algorithms are trained on the training dataset and it can categorize the data into different classes. On the other hand, regression is the process of finding the correlation between the dependent and independent variables. Regression algorithms are used for the prediction of house prices, stock predictions, weather forecasting, and so on.

Feature selection

With the excess development of big data, we have more access to the high dimensional dataset and it has also improved the performance of most of the machine learning algorithms. But, on the other side, there is lots of data which are collected from different sensors and the methods are more corrupted, noisy, and sometimes useless. These types of data sometimes influence the accuracy and computational speed of the model. Hence, the selection of useful data or features is always a cumbersome task for any machine learning algorithm. Because useful and effective features always improve the accuracy of the model and also increases the model interpretability. Keeping this thing in mind, feature selection algorithms are always playing an important role in the statistical learning-based approach.

Feature selections are also referred to as variable selection or attribute selection or subset selection and it is a process by which the data scientist will automatically select the relevant or useful features to use in the statistical-based learning methods. The feature selection algorithm will select the most suitable features from the pool of features which can help reduce the prediction time and also improve the prediction accuracy.

The feature selection approach can be explained in two steps:

i. Combination of different search techniques that adds new feature subsets.

ii. The evaluation process measures the score of different feature subsets.

This could be computationally expensive because selecting the best subset features from the available features is always challenging. Keeping the same thing in mind, feature selection approaches are categorized into different categories and now we will explore the different methods and techniques which makes the selection process easier, simpler, faster, and reliable.

Here are some feature selection approaches:

- Filter
- Wrapper
- Embedded methods

Filters

In this method, the irrelevant attributes or redundant columns are filtered out. We can choose a single statistical measure that is suitable for the whole dataset. The module can calculate the feature score for each column and then the column can return their value by their feature score. If we choose the right feature then there is a possibility of improving the accuracy and efficiency of the model. The selection of the best feature never depends on any machine learning algorithm. The features can be selected according to their score in numerous statistical tests that suit the particular dataset. The various statistical test methods based on categorical or continuous data are:

Feature/Response	Continuous	Continuous
Continuous	Pearson's Correlation	LDA
Categorical	ANOVA	Chi-Square

Pearson correlation

The Pearson correlation method is used to measure the quantifying linear independence between two continuous variables x and y. the value can vary from -1 to +1. It is shown as:

$$\rho_{x,y} = \frac{cov(x,y)}{\sigma_x \, \sigma_y} \rho_{x,y} = \frac{cov(x,y)}{\sigma_x \, \sigma_y} \qquad (3)$$

It is calculated by taking the covariance of the two variables and then dividing it by the product of the standard deviation. If the scale of the two variables changes, it doesn't affect the coefficient value.

Chi-squared

It is another statistical method that is used to measure how much the expected values are closer to the actual values. In this method, we assume that variables are random variables that are drawn from an adequate sample of independent variables.

Linear discriminant analysis (LDA)

LDA is used to find the linear combination of features that categorizes the feature into different classes. It is suitable for categorical data.

Analysis of variance (ANOVA)

This method is similar to LDA but in this method one dependent feature and one categorical independent feature is used. This method is suitable for finding if several groups are equal or not.

Wrappers

In the wrapper method, the feature selection method is based on machine learning algorithms. This method follows the greedy search approach in which it tries to evaluate the performance of all the possible combinations of features according to the various evaluation criteria. Evaluation criteria is performance metrics that always depends on the type of the problem. For example, for classification, the evaluation parameter is accuracy, precision, and recall. For regression, the evaluation parameter is p-values and r-squared, and so on. Finally, wrappers will select the best combination of the features which is going to give the optimal results. The same procedure is depicted in *Figure 4.4*. The different types of wrapper methods are forward feature selection, backward feature elimination, bidirectional search, exhaustive feature selection, and so on.

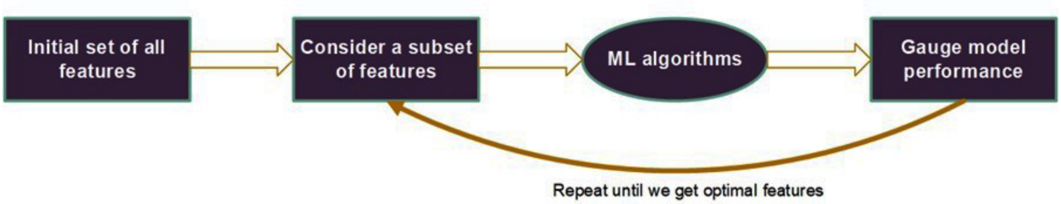

Initial set of all features
Consider a subset of features
ML algorithms
Gauge model performance
Repeat until we get optimal features

***Figure 4.4**: Flow chart of wrappers method*

Embedded methods

Embedded methods complete the feature selection process within the construction of machine learning algorithms. In other words, we can say that the feature selection process will be done during the training of the model and that's the only reason it is called **embedded methods**. The advantage of an embedded method is that the classification and selection of the features are done at the same time because for that reason they are less computationally expensive. The different types of embedded methods are L1 regularization, L2 regularization, and L1/L2 regularization. These methods will be discussed later in this chapter.

Model selection

Model selection is a technique of selecting the best model which is suitable for the particular dataset or application. Generally, the test data set consists of those points which are not seen by the model in the training phase. There are different types of model selection methods which we will discuss now.

Re-sampling methods

Re-sampling is a simple technique of rearranging data samples. Rearranging of data samples ensures that the model will perform well on the dataset which is not seen before. In other words, we can say that the re-sampling will ensure that the model will work perfectly during training and testing. The different re-sampling methods are:

Random split: The random split method is used to split the data randomly into training, testing, and validation sets. It helps prevent the biased sampling of the dataset. The test dataset is used for the model evaluation purpose. The validation set consists of completely unseen data samples that are not seen in tuning and feature selection also. This is used for a final evaluation of the model.

Time-based split: Sometimes random split does not work for some specific type of datasets. For example, if we are training the model for weather forecasting, the random split will not work properly. There is the possibility that it will jumble the dataset into different patterns. For such type of dataset time split is the perfect solution.

k- fold cross-validation: The cross-validation technique is the extended version of the random split method in which the dataset is randomly shuffled and then finally split into k different groups.

Therefore k groups are considered as testing sets and other groups are considered as training sets. The model is tested on the testing groups and this process will continue for all the k groups. At the end of the process k different results will be available for k different groups and the best model will be selected based on the highest score of the model.

Bootstrap: This method is close to the random splitting method because it follows the same concept. In the first step, the size of the sample will be selected and generally, it will be equivalent to the size of the original dataset. In the second step, randomly the sample data points will be selected from the original datasets and then finally added to the bootstrap sample. The same process must be repeated N number of times and here N depends on the sample size. The bootstrap sample contains multiple instances of the same data points.

Probabilistic measures

Probabilistic measures do not only measure the performance of the model but also considers the complexity of the model. The other important point is that the performance of the model is calculated on the training set only and a test set is not required for the performance evaluation. The different probabilistic measures are:

Akaike Information Criterion (AIC)

This is a well-known fact that no machine learning model is a hundred per cent accurate. There is always a possibility of information loss and this information loss can be measured by Kulback-Liebler measure. KL measures the difference in the probability distribution of two variables. Hirotugu Akaike is a statistician and he considered the relationship between KL information and the maximum likelihood. He developed a new concept of **information criterion** (**IC**). Hence **Akaike's**

information criterion (**AIC**) is used to measure the information loss. This method helps choose the best method between two different methods according to the lost information. It can be calculated as:

$$AIC = \frac{(2K - 2 \log (L))}{N} \qquad (4)$$

Here,

L= Maximum likelihood of the model.

N= Number of data points in the training dataset

K= Number of independent variables. Bayesian Information Criterion (BIC)

The concept of BIC was derived from the Bayesian probability concept and it is the best suitable method for models which are trained under the maximum likelihood estimation. The formula for calculating BIC is:

$$BIC = K * \log (N) - 2 \log (L) \qquad (5)$$

Here,

L= Maximum likelihood of the model.

N= Number of data-points in the trainingPro dataset.

K= Number of independent variables.

BIC is suitable when the size of the dataset is not very small.

Minimum Description Length (MDL)

As the name suggests, MDL can find the minimum number of bits that are required to represent such a type of model. The concept of MDL is derived from the concept information theory and it deals with the quantities such as entropy that measures the average number of bits required to represent an event from a probability distribution or a random variable. MDL can be represented with the help of the following formula:

$$MDL = L(h) + L(D \mid h) \qquad (6)$$

Here,

D = Predictions made by the model

L(h) = Number of bits required to represent the model

L(D | h) = Number of bits required to represent the predictions from the model

Model evaluation

Model evaluation is a method for checking the correctness of the model on the test dataset and it plays a big role in academic machine learning in research and industrial settings. The main aim of model evaluation is to estimate the accuracy of the model for unseen datasets or testing datasets. There are different metrics for evaluating the performance of the model but finding the right evaluation metric is always a difficult task and it also depends on the problem that we are trying to solve. A clear understanding of a variety of evaluating parameters can be helpful for the evaluators to understand the problem statement. Now we will discuss all the evaluation metrics in detail.

The evaluation parameters are divided into different categories:

- Classification metrics
- Regression metrics
- Clustering metrics

Classification metrics

To check the accuracy of every classification model, a confusion matrix is constructed which gives a clear idea of how many test cases are classified correctly. To understand this, let us discuss the confusion matrix:

In the confusion matrix, yes represents the positive classes and 0 or No represents the negative classes and here is the confusion matrix.

	Actual(NO)	**Actual (YES)**
Predicted (NO)	True Negative	False Negative
Predicted (YES)	False Positive	True Positive

There are different ways of representing classification metrics which are discussed as follows:

Accuracy

Accuracy is the simplest metric for the representation of classification. It can be defined as several test cases correctly classified and divided by the total number of test cases. The formula for calculating the accuracy is:

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

This is the most suitable method for checking the performance of the model if the datasets are unbalanced.

Precision

Precision is used to find the correctness of the classification and it can be calculated using the following formula:

$$Precision = TP / (TP+FP)$$

This equation represents the ratio between correct positive classifications to the total number of predicted positive classifications. if the fraction value is greater it means higher the precision. A higher precision value represents that model can classify the positive a]classes appropriately.

Recall

Recall tells us how many numbers of positive cases are correctly identified out of the total number of positive cases. It can be calculated using the following formula:

$$Recall = TP / (TP + FN)$$

F1 Score

F1 score is the mean value of recall and precision. Hence it balances the strengths of recall and precision. This is the most suitable evaluation metric where human lives are involved. It can be calculated using the following formula:

$$F1\ Score = 2 * \left(\frac{Precision * Recall}{Precision + Recall}\right)$$

AUC curve

AUC curve is a graph between the true positive rate and the false-positive rate. AUC-ROC stands for Area under the Receiver Operating Characteristics. If it covers more area it means the performance of the model is higher. If the curve is somewhere near the 50% diagonal line, it suggests that the model randomly predicts the output variable.

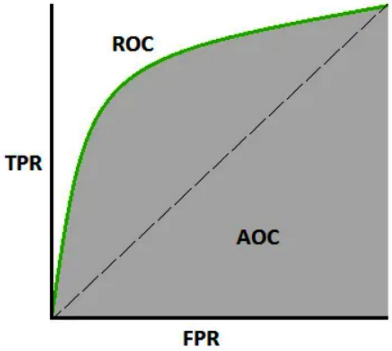


***Figure 4.5**: AUC –ROC Curve.*

Regression metrics

Classification models have discrete output variables whereas regression models provide a continuous output variable. Hence there are different ways of evaluating the performance of the regression model which we will discuss further.

Mean Squared Error or MSE

MSE is used to calculate the difference between the actual value and the predicted value, it squares the error value and finally, it provides the mean of all the errors. It can be represented as:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Root Mean Squared Error or RMSE

RMSE is the root value of MSE and it is helpful to reduce the scale of the error and tries to keep the value closer to the actual value.

Mean Absolute Error or MAE

MAE is the mean of the absolute error values. If we want to ignore the outlier values to a certain degree then MAE can be the right choice as it can reduce the penalty of the outliers significantly with the removal of the square terms.

$$MAE = \frac{1}{n}\sum\nolimits_{i=1} |x_i - x$$

Root Mean Squared Log Error or RMSLE

It is the same as RMSE except for the log function along with the actual and predicted values.

$$RMSLE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\log(x_i + 1)) - (\log(y_i + 1))^2}$$

Here x represents the actual value and y represents the predicted value. These metrics help to scale down the effect of the outliers by downplaying the higher error rates with the log function.

Clustering metrics

Clustering algorithms are used to predict the group of data points and for clustering-based algorithms, distance-based metrics are most useful and effective. Now we will discuss some of the clustering metrics:

Dunn Index

Dunn index metrics are used to identify the clusters which are having low variance. It can be calculated using the following formula:

Here,

δ(Xi, Yj) =Distance between Xi and Xj

Δ(Xk) = Distance within the cluster Xk

If several clusters are more and the dimensions are high then there is the possibility of increasing the computation cost.

Silhouette Coefficient

Silhouette Coefficient tracks how every point in one cluster is close to every point in the other clusters in the range of -1 to +1:

- If silhouette values are closer to +1 then it indicates that sample points of two different clusters are far away.
- If it is zero then it indicates that the sample points are near to the decision boundary.
- If the silhouette values are closer to -1 then it indicates that sample points are not assigned correctly to the clusters.

Elbow method

The elbow method is used to determine the number of clusters in a dataset by plotting the number of clusters on the x-axis against the percentage of variance explained on the y-axis. The point in the x-axis where the curve suddenly bends (the elbow) is considered to suggest the optimal number of clusters.

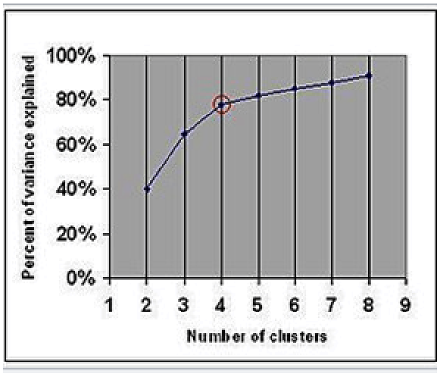


***Figure 4.6**: Representation of Elbow method*

Statistical learning algorithms

Statistical learning is a collection of tools that are used to understand different data. The tools can be categorized as supervised and unsupervised tools. Supervised statistical learning is used to build a statistical model for predicting the output using the given input datasets. Supervised statistical learning is used in various areas such as medical, business-related applications and public policies, and so on. In unsupervised statistical learning, input will be given but there is no supervised output and this type of learning is best suitable to understand the structure and the relationship between data. For a better understanding of statistical learning will discuss various algorithms in detail which is shown in *Figure 4.7*.

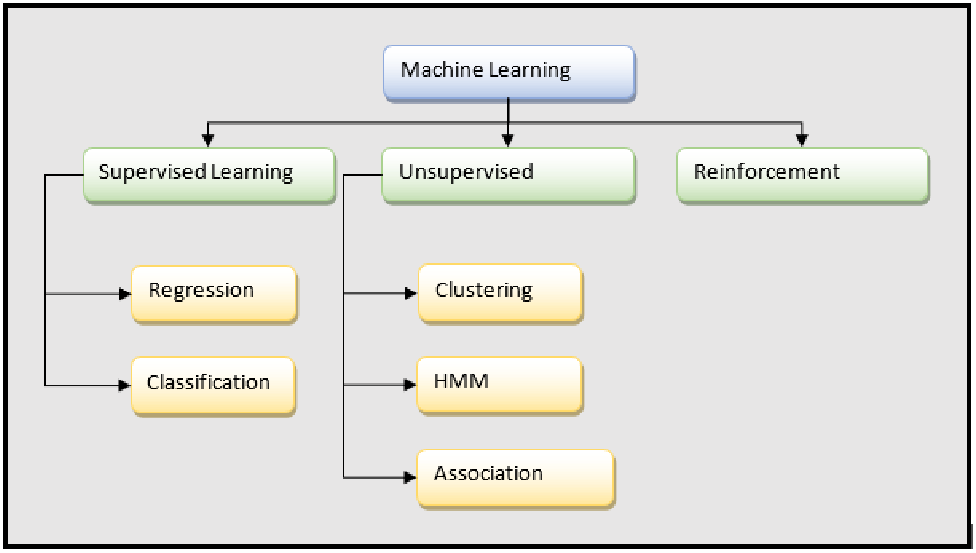


***Figure 4.7**: Various categories of machine learning algorithms*

Supervised learning

Supervised machine learning algorithms are used to predict the particular kind of values. If we are working with the supervised machine learning algorithms then the dataset must be labeled properly. A traditional example of supervised learning is email where the data is already classified as 'spam' or 'not spam'. In this approach, multiple independent variables are used to determine the values of other dependent variables. The performance of the algorithm can be measured by comparing the original and predicted results and this is the main advantage of supervised machine learning algorithms. The first step of these algorithms is to analyze the training dataset and then in the next step find a function that can be helpful to understand the new dataset but dataset must be labeled for predictive analysis. To understand the overall process you can refer to *Figure 4.8*. There are different supervised machine learning algorithms that are available which we will discuss further and the output will be in one of the forms:

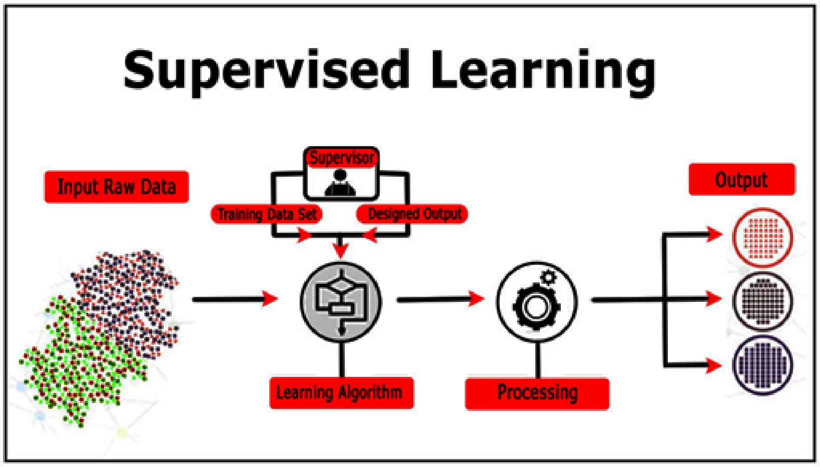


***Figure 4.8**: Overall process of supervised learning*

Regression

Regression is a statistically based approach that establishes a relationship between the target and the predictor variable. In other words, regression allows us to understand that how the value of the dependent variable is changing when the value of the independent variable is fixed. For example, age, salary, house price, temperature, and so on. There are different types of regression techniques that we will discuss in the next section.

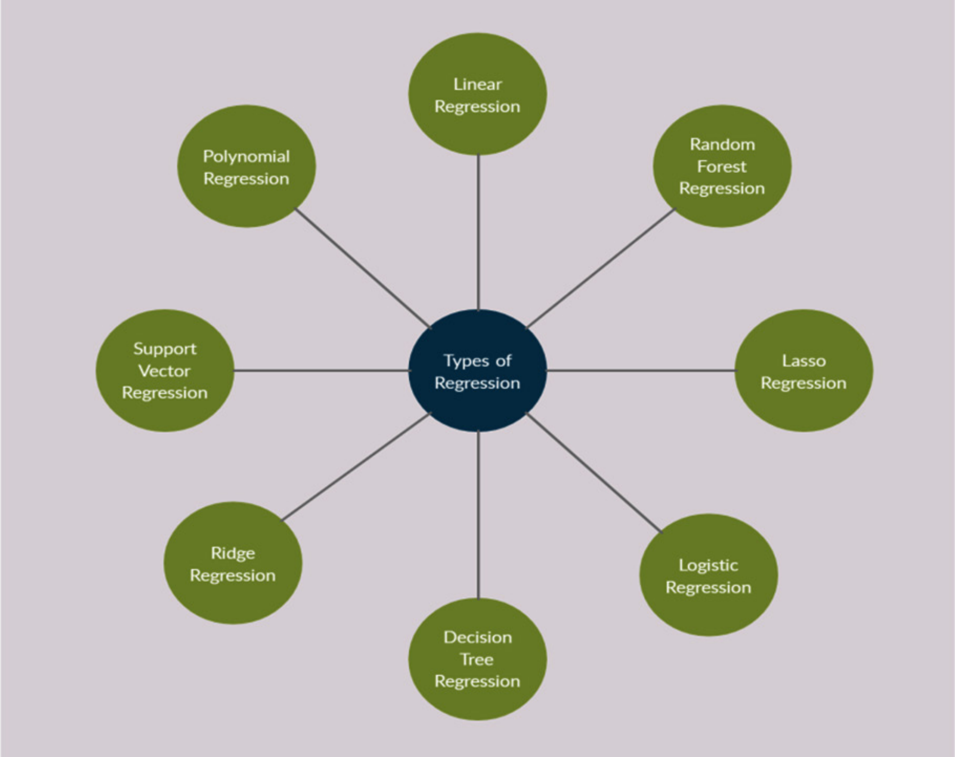


Figure 4.9: *Types of regression*

Linear Regression

Linear regression is a simple regression algorithm that represents the association between continuous variables. Continuous variables are nothing but dependent variables and independent variables. If there is only one independent variable to predict the value of dependent variables then such type of regression is called **simple linear regression**. If a number of input variables are more than one for the prediction of the dependent variable values then it is referred to as multiple linear regressions. Basically, this method is used for predictive analysis.

Logistic Regression

This is another method of solving classification based problems. This method helps in finding the probability that a new instance belongs to a certain class. Because it is related to probability then the output will be either zero or one, true or false, and yes or no. When the logistic classifier is used as a binary classifier then classes are considered as positive or negative. If the probability value is on the higher side (greater than 0.5) then we will consider class as a positive class and if the probability value is low then (less than 0.5) then we will consider class as a negative class.

Consider the example of email filtering. Let's assume that benign ham emails fall under the negative class and malignant spam emails comes under the positive class. In the beginning, many examples of labeled emails are taken and trained the model appropriately so that it can predict the class of new email. Suppose we feed a new example to the model and the model returns the value between 0≤y≤1. The value is 0.7. With this value we can predict that there is 70% probability that the new email is spam.

Polynomial Regression

This is a form of regression analysis. In this model, an association between the independent variable x and the dependent variable y is identified using nth degree of the polynomial in x. this model helps to fit a non-linear model to the data and it is considered as a special case of multiple linear regression. Consider the following scenario: suppose we have a dataset and it consists of data points that are not linear in fashion. To fit such types of data points linear regression is not a good approach. Therefore, here we need polynomial regression. In this approach, the original features of the dataset are transformed into polynomial features of the given degree and finally modeled using the linear model.

Support Vector Regression

SVM algorithm is used for classification and regression. If this algorithm is used to solve the regression problem then it is called **Support Vector Regression**. This algorithm works for continuous variables. There are some keywords that are used for support vector regression that is also shown in *Figure 4.10*:

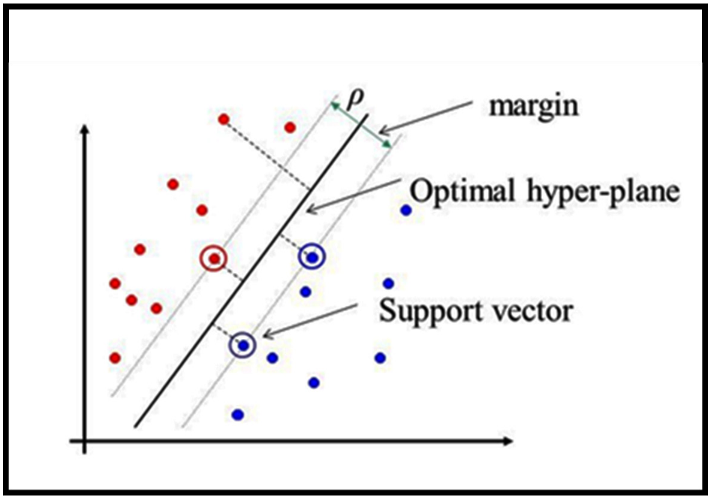


***Figure 4.10**: Support vector machine regression*

- **Kernel:** This function is used to map the low dimensional data into high dimensional data.
- **Hyper Plane:** In SVM, this line is used to separate the two different classes. But in support vector regression, this line is used to predict the continuous variables and it tries to cover all the data points.
- **Boundary Line:** Boundary lines are the two lines that are far from the hyperplane and it is used to create a margin for the data points.
- **Support Vector:** These are the data points that are nearest to the hyperplane.

Decision Tree Regression

This is another algorithm that is used for classification and regression-related problem. This algorithm is able to deal with categorical and numerical data. As name suggests, this algorithm builds a tree structure in which internal node of the tree represents the test attributes and each branch is representing the results of the test and the leaf node represents the final output or decision. When we start constructing a tree then the first node is called **root node** or **parent node** and parent node is split into two parts: **left child node** and **right child node**.

Ridge Regression

A linear or polynomial regression sometimes fails if there is a high correlation between the independent variables. To solve such problems, ridge regression can be used. This is an extended and robust version of linear regression and in this small amount of bias is introduced which is helpful for long term predictions. The amount of bias that is added in the model is known as the **ridge regression penalty**. This penalty can be computed by multiplying the lambda with the squared weight of the features. This method is also referred to as regularization technique and can be used for reducing the complexity of the model and it is also called **L2 regularization**.

Lasso Regression

It is a technique for regularization which is also helpful for reducing the complexity of the model. This method is the same as ridge regression except that the penalty term contains only the absolute weights instead of the square of weights. Since it takes absolute values, hence, it can shrink the slope to 0, whereas ridge regression can only shrink it near to 0. It is also referred to as L1 regularization.

Classification

Classification is a supervised machine learning algorithm that is used to categorize different classes. The common classification related applications are speech recognition, face recognition, document verification, and so on. The classification problem can be related to the binary or multiclass classification problems. Many algorithms are available which are used for classification purposes and it is already shown in *Figure 4.7*. Let's discuss some of the classification algorithms:

Naive Bayes Classifier

This method is based on the Bayes' Theorem and it works on an assumption that the presence of a particular feature in a class is unrelated to the presence of any other feature. This algorithm is easy to build and it is very helpful for the large-sized datasets. This algorithm is also called as **Generative Learning Model**.

Nearest Neighbor algorithm

It is another supervised classification algorithm that uses the concept of proximity or sameness. This algorithm uses a bunch of labeled points for the learning purpose and then further use them for labeling of other points. If the algorithm wants to label other new points then it looks at the labeled points which are closer to the new points. Closeness can be expressed as a dissimilarity function. The algorithm will check the new points with k number of the neighborhood points and finally assign the label according to the points that the neighbors have. The same concept is explained in *Figure 4.11*. As you can see that the test dataset will get the same color that most of its neighbors have.

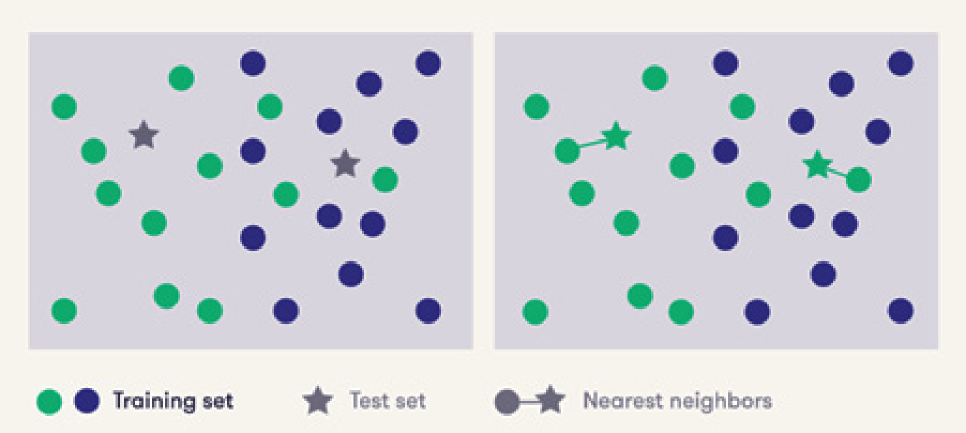


***Figure 4.11**: Example of nearest neighbor algorithm*

But always using the geometric distance for finding the nearest m-neighbor doesn't work because sometimes input data can be available in text form. In that case, finding the geometric distance is not a suitable approach. Hence, we need to choose other parameters as the distance metrics.

Logistic Regression

Logistic regression is sometimes referred to as predictive learning model. This method is suitable when more than one independent variable is used to determine the value of a dependent variable. The main goal of the logistic regression is to find the best fitting model which can describe the characteristics of the independent variable and dependent variable. This method is better than nearest neighbor because it's able to explain the quantities features which lead to classification.

Decision Trees

This is another algorithm that is used for classification and regression-related problems. This algorithm can deal with categorical and numerical data. As the name suggests, this algorithm builds a tree structure in which the internal node of the tree represents the test attributes and each branch represents the results of the test and the leaf node represents the final output or decision. When we start constructing a tree then the first node is called root node or parent node and the parent node is split into two parts: left child node and right child node.

Neural Network

A neural network consists of neurons and these neurons are arranged in layers that can convert the input vectors into some output vectors. Each unit of the neuron works as an input and then applies a non-linear function on the input and finally its passes the output to the layer. Generally, this network is a feed-forward network in which a unit feeds its output to all the units of the next layers but there is no possibility of feeding the output back to the previous layers.

Unsupervised learning

Unsupervised learning algorithms are those algorithms where you have input values or input data but there are no corresponding output variables. The main goal of unsupervised learning is to understand the structure of the data or the distribution of the data and these algorithms help the learners to understand more about the data. In unsupervised learning, there is no concept of teachers. Algorithms have their own choice to learn and explore more about the data. In unsupervised learning, data is neither classified nor labeled and the algorithm will work without any guidance. Here, the task of the machine is to group unsorted information according to similarities, patterns, and differences without any prior training of data as shown in *Figure 4.12*:

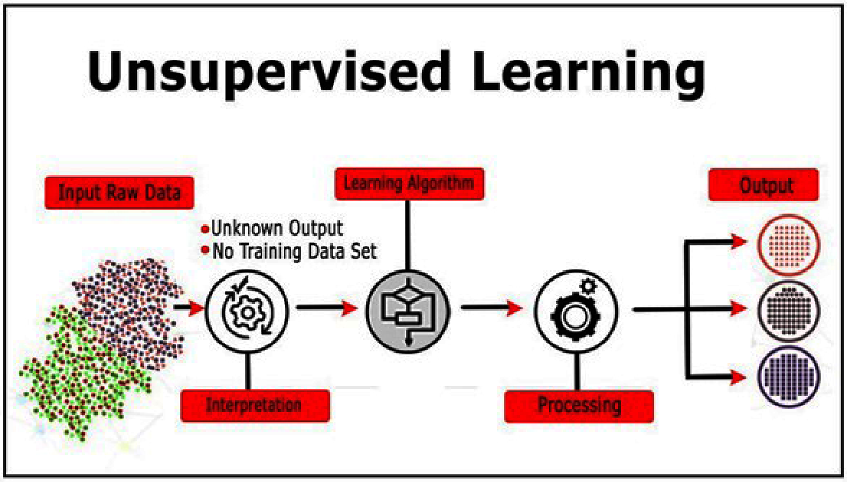


Figure 4.12*: Overall process of unsupervised learning*

The different types of unsupervised machine learning algorithms are available which we will discuss now:

Hierarchical Clustering

These algorithms can build the hierarchy of clusters. In the beginning, the data is assigned to the nearest cluster, and if two clusters are close to each other then they will be merged. This algorithm will end if there is only one cluster left.

K-means Clustering

It is an iterative clustering algorithm that helps us to find the highest value for every iteration. In the beginning, the desired number of the cluster has been selected, and then the data points will be clustered into k groups. A larger the k-value means a smaller size of groups with more granularities. Lower the k-value means the large size of groups with fewer granularities. The output of the algorithm is a group of labeled classes and the new data points will be assigned to one of the classes. Generally in the k- means clustering for every group a centroid will be defined. The centroid is the heart of the cluster which is used to capture the closest points to them and then add them to the cluster.

K- Nearest neighbors

K- the nearest neighbor is the simplest algorithm and it is different from other machine learning algorithms because it does not produce any model. This algorithm is going to store all possible cases and then classify the new data points based on similarity measures. When the size of the training data is large, the algorithm's speed improves.

Principal Components Analysis

PCA is a technique of dimensionality reduction of the dataset. There is a possibility that the dataset consists of many variables and these variables are correlated to each other either less or more. The same concept is repeated and done by transforming and bringing the variables to a whole new set of variables, which are called the **principal components** and are even termed to be orthogonal, ordered in such a way that the retention of variation which is present in the original variables can be decreased as we try to move down in the proper order. So, by following this particular way, the 1st principal component retains the most and maximum variation that was earlier present in the original components. The principal components are known to be the eigenvectors of a covariance matrix, and hence they are even called **orthogonal**. Most importantly, the dataset is based on what the PCA techniques are to be used and must be scaled. The result also turns out to be sensitive based on the relative scaling. As a layman, it can be termed as a method of summarizing the data. Just imagine having some wine bottles on your dining table. Each wine would be described only by its attributes that are color, age, strength, and so on. But eventually, redundancy will arise may be because many of them would be measured based on the related properties.

Summary

The complete chapter has been summarized in the following points:

- Statistical learning is a combination of statistics and machine learning.
- SupervisedStatistical statistical learning is used to build a statistical model for predicting the output using the given input datasets.
- In unsupervisedStatistical statistical learning, input will be given but there is no supervised output and this type of learning is best suitable to understand the structure and the relationship between the data.
- In the classification process, algorithms are trained on the training dataset and it can categorize the data into different classes.
- Regression is the process of finding the correlation between the dependent and independent variables.
- Regression algorithms are used for the prediction of house prices, stock predictions, weather forecasting and so on.

- Feature selection refers to variable selection or attribute selection or subset selection and it is a process by which the data scientist will automatically select the relevant or useful features to use in the statistical-based learning methods.
- The feature selection algorithm will select the most suitable features from the pool of features which can help reduce the prediction time and also improves the prediction accuracy.
- Filter, wrapper an embedded methods are three approaches of to feature selection.
- Model selection is a technique of selecting the best model which is suitable for the particular dataset or application.
- Re-sampling methods, probabilistic measures and Bayesian information criterion are some popular methods of model selection.
- Model evaluation is a method for checking the correctness of the model on the test dataset and it plays a big role in academic machine learning in research and industrial settings.

Exercise (MCQs)

1. **Which of the following is a machine learning technique?**

   a) based on human supervision

   b) supervised learning

   c) semi-reinforcement learning

   d) all of the above

2. **A model-based learning method which is built on various model parameters is called as**

   a) mini-batches

   b) optimized parameters

   c) hyperparameters

   d) super parameters

3. **Which of the following is not supervised learning?**

   a) CPCA

   b) Decision Tree

   c) Linear Regression

   d) Naive Bayesian

4. **Which statement is incorrect about Naïve Bayes?**

   a) All attributes are important equally.

   b) Attributes are statistically dependent on one another given the class value.

   c) Attributes are statistically independent of one another given the class value.

   d) Attributes can be nominal or numeric.

5. **Suppose we want to perform clustering on a geometric location of a house and clusters can be of different shapes and sizes. Which is the appropriate method for this approach.**

   a) Decision trees

   b) Density-based clustering

   c) Model-based clustering

   d) K-means clustering

6. **Data can be visualized using?**

   a) graphs

   b) charts

   c) maps

   d) All of the above

7. **Which is the correct statement for supervised learning and unsupervised clustering?**

   a) output attribute

   b) hidden attribute

   c) input attribute

   d) categorical attribute

8. **Which of the following is a correct categorical outcome?**

   a) RMSE

   b) Accuracy

   c) Squared

   d) All of the mentioned

9. **In terms of variance, which of the following statements about k in k-NN is correct?**

   a) As k is increased, the variance increases

   b) As k is decreased, the variance increases *c)* Both A and B

d) None of these

10. Different learning methods include

a) Introduction

b) Analogy

c) Deduction

d) Memorization

Answers

1. **a**
2. **c**
3. **a**
4. **b**
5. **b**
6. **d**
7. **b**
8. **b**
9. **b**
10. **a**

Short question answer

1. Differentiate supervised and unsupervised machine learning.
2. Explain how a ROC curve works.
3. Define precision and recall.
4. Explain the difference between L1 and L2 regularization.
5. Which is more important to you: model accuracy or model performance?

Long question answer

1. Explain statistical learning in detail.
2. Explain the feature selection methods.
3. Explain the different model evaluation methods.
4. Explain the supervised machine learning algorithms in detail.
5. Explain regression techniques in detail.

CHAPTER 5
Semi-Supervised Learning, Reinforcement Learning

Introduction

Semi-supervised learning is an approach to machine learning that combines a small amount of labeled data with a large amount of unlabeled data during training. Reinforcement learning (RL) is an area of machine learning concerned with how software agents ought to take actions in an environment in order to maximize the notion of cumulative reward. Reinforcement learning is one of three basic machine learning paradigms, alongside supervised learning and unsupervised learning.

This chapter describes the different approaches of the semi supervised learning and reinforcement learning with real-time examples.

Structure

In this chapter, we will discuss the following topics:

- Semi-supervised learning
- Markov Decision Process (MDP)
- Bellman equations
- Monte Carlo methods
- Policy evaluation using Monte Carlo

- Policy iteration and value iteration
- Q-learning
- State-Action-Reward-State-Action (SARSA)
- Model-based reinforcement learning

Objectives

By reading this chapter, you will be able:

- To learn about the concepts of semi-supervised and reinforcement learning.
- To learn about the fundamentals of the Markov decision process and its applications.
- To learn about the *Bellman Equation* for value function.
- To learn about the *Monte Carlo*, policy iteration and value iteration.
- To evaluate the Q-learning reinforcement learning algorithm.
- To identify the significance of the State-Action-Reward-State-Action (SARSA)
- To identify the role of the model-based reinforcement learning technique.

Semi-supervised learning

Semi-supervised learning falls between unsupervised learning and supervised learning. Unlabeled data, when used in conjunction with a small amount of labeled data, can produce considerable improvement in learning accuracy. The acquisition of the labeled data for a learning problem often requires a skilled human agent or a physical experiment. The cost associated with the labeling process thus may render large, fully labeled training sets infeasible, whereas the acquisition of unlabeled data is relatively inexpensive. In such situations, semi-supervised learning can be of great practical value. Semi-supervised learning is also of theoretical interest in machine learning and as a model for human learning.

Semi-supervised learning may refer to either inductive or transductive learning. The goal of inductive learning is to infer the correct mapping from X to Y. The goal of transductive learning is to infer the correct labels from the given unlabeled data only.

Reinforcement learning (**RL**) differs from supervised learning in not needing labeled input/output pairs to be presented, and in not needing sub-optimal actions to be explicitly corrected. Instead, the focus is on finding a balance between the exploration (of uncharted territory) and the exploitation (of current knowledge).

The environment is typically stated in the form of a **Markov decision process** (**MDP**) because many reinforcement learning algorithms for this context utilize dynamic programming techniques. The main difference between the classical dynamic programming methods and reinforcement learning algorithms is that the latter does not assume knowledge of an exact mathematical model of the MDP and they target large MDPs where exact methods become infeasible.

Reinforcement learning, due to its generality, is studied in many other disciplines, such as game theory, control theory, operations research, information theory, simulation-based optimization, multi-agent systems, swarm intelligence, and statistics. In the operations research and control literature, reinforcement learning is called approximate dynamic programming, or neuro-dynamic programming. The problems of interest in reinforcement learning have also been studied in the theory of optimal control, which is concerned mostly with the existence and characterization of optimal solutions, and algorithms for their exact computation, and less with learning or approximation, particularly in the absence of a mathematical model of the environment. In economics and game theory, reinforcement learning may be used to explain how equilibrium may arise under bounded rationality.

Markov Decision Process (MDP)

The most widely recognized issue of retraining **Reinforcement Learning** (**RL**), there is a chief called an **operator** and the connection we experience is known as **nature**. The earth offers rewards and new states dependent on the activities of the operator. In this manner, in reinforcement learning, we do not show an operator how to accomplish something however give them compensations for being acceptable (positive) or terrible (negative) contingent upon their activities.

Markov Chain and Markov Process

The Markov property conveys that the future relies just on the present and not on the past. The Markov chain is a probabilistic model that exclusively relies on the current status and not the past states, that is, what is to come is restrictively freed from a prior time. Moving to begin with one state then onto the following is called **progress** and its probability is known as a **change probability**. We can consider an instance of anything where the following state depends just upon the present status.

A wide range of calculations manages this issue. Fortifying learning is characterized as a particular kind of issue, and the entirety of its answers are recorded as reinforcing learning calculations. For this situation, the specialist ought to decide the best game-plan depending on their present circumstance (state). When this progression is rehashed, the issue is known as the **Markov Decision Process**.

Markov's choice procedure is a model for foreseeing results. Like Markov's chain, the model attempts to anticipate the result given just the data given by the present status. Notwithstanding, Markov's choice procedure fuses components of the activity and inspiration. At each progression simultaneously, the chief may decide to make a move depending on the present status, bringing about the model moving to the subsequent stage and remunerating the leader.

Markov Decision Process is characterized by:

1. **State**: A state S is a collection of states known as the state space.
2. **Model or change work T**: This is a component of the present status, the activity performed, and the state wherein it closes.
3. **Action**: Action A will be a conceivable activity. These are things that an operator can do inside a specific state.
4. **Reward**: This will be a genuine esteemed prize capacity. It reveals to us the convenience of entering the state.
5. **Policy**: It lets us know, regardless, what move to make. The ideal arrangement is the one that boosts the drawn-out remuneration.

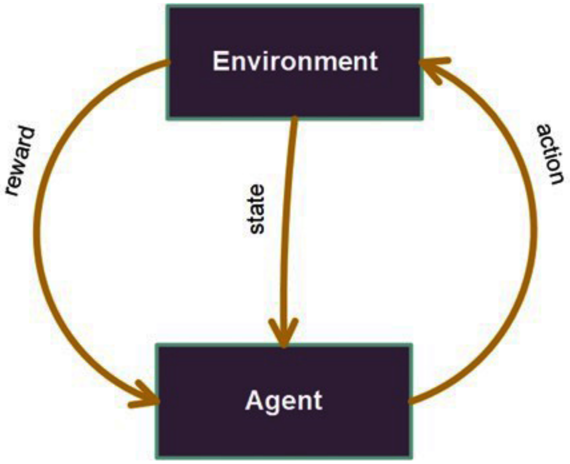


***Figure 5.1**: A simplified diagram of the MDP*

As shown in *Figure 5.1* at each step, the state of the environment is presented to the agent, who then makes a decision. The environment changes and the agents are rewarded with a fair (positive), negative, or neutral value based on the action taken. These observations are given to the agent to make another decision.

In the case of RL, the agent is the decision-maker. It receives recognition of the current state of the natural environment and acts based on that information. The agent is then rewarded according to the selected action. This reward, as well as the new status, is passed on to the agent and another action is taken. The environment includes whatever the agent has no control over. This feedback loop is known as an MDP.

The reward is calculated as a function of the current state of the environment and the action taken. In the full MDP model, all the information needed to calculate the agent policy is available. In addition to locations, actions, and rewards (as well as discounts), the MDP model requires opportunities for state change, opportunities for one country to move to provide the action taken. With this information, relevant policies can be calculated directly by strategies such as policy measurement. With policy iteration, the policy of each state is continually updated until the expected future reward for each state is maximized.

Applications of Markov Decision Process

There are some applications where we can use MDP. The following are the applications of MDP:

1. Hybrid energy storage system
2. Traffic signal control
3. Generating explanations
4. Remote experimental setups for evaluation
5. Speech to the speech interaction system
6. A leak detection system
7. A learning design recommendation system

Bellman Equations

Richard Bellman was an American applied mathematician who derived the Bellman equation which allowed us to start solving MDPs. Bellman equation is the basic block of solving reinforcement learning and is omnipresent in RL. It helps us to solve MDP. Here, to solve means finding the optimal policy and value functions.

Bellman functions allow us to write an equation that will represent our state-value function as a recursive relationship between the value of a state and the value of its successor states. Mathematically we can define Bellman Expectation Equation as:

$$V_{\pi}(s) = \mathrm{E}\,(\,R_{t+1} + \;\gamma\, V_{\pi}\;(s_{t+1}\,)\,|\,S_t = s \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ (1)$$

Bellman Equation for Value Function (State-Value Function)

From the above-mentioned equation (1), we can see that the value of a state can be decomposed into an immediate reward (R[t+1]) plus the value of a successor state (v[S (t+1)]) with a discount factor(γ). This still stands for Bellman Expectation Equation. But now what we are doing is we are finding the value of a particular state subjected to some policy(π). This is the difference between the Bellman Equation and the Bellman Expectation Equation.

The above equation tells us that the value of a particular state is determined by the immediate reward plus the value of successor states when we are following a certain policy(π).

Similarly, we can express our state-action value function (Q-Function) as follows:

$$q_{\pi} (s,a) = E_{\pi} [R_{t+1} + \gamma q_{\pi} (S_{t+1}, A_{t+1}) \mid S_t = s, A_t = A] \ldots\ldots\ldots\ldots..(2)$$

Bellman Expectation Equation for State-Action Value Function (Q-Function)

Let us call this equation (2). From the above equation, we can see that the State-Action Value of a state can be decomposed into the immediate reward we get on performing a certain action in state(s) and moving to another state(s′) plus the discounted value of the state-action value of the state(s′) for some action(a) our agent will take from that state on-wards.

Going Deeper into Bellman Expectation Equation

First, let us understand the Bellman Expectation Equation for State-Value Function with the help of a backup diagram.

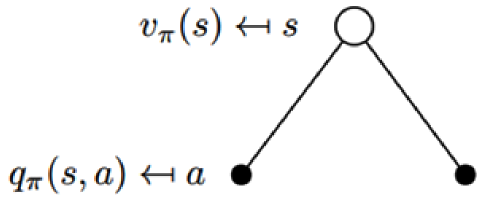


***Figure 5.2**: Backup Diagram for State-Value Function*

This *Figure 5.2* describes the value of being in a particular state. From the state *s*, there is some probability that we take both the actions. There is a Q-value (State-action value function) for each of the actions. We average the Q-values which tells us how good it is to be in a particular state. It defines Vπ(s). [Look at Equation 1]

Mathematically, we can define it as follows:

$$V_\pi(s) = \sum_{a \in A} \pi(a|s)\, q_\pi(s,a) \ \ldots\ldots\ldots\ldots\ldots\ldots(3)$$

Value of being in a state

Equation (3) also tells us the connection between the State-Value function and State-Action Value Function.

Now, let us look at the backup diagram for State-Action Value Function:

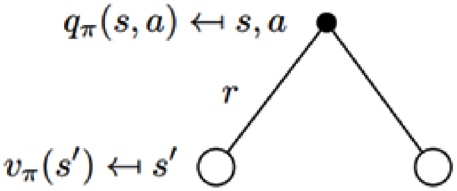

$q_\pi(s,a) \leftarrow\!\shortmid s,a$
r
$v_\pi(s') \leftarrow\!\shortmid s'$

***Figure 5.3**: Backup Diagram for State-action Value Function*

Figure 5.3 shows that suppose we start by taking some action(a). Because of the action(a) the agent might be blown to any of these states by the environment. Therefore, we are asking, how good it is to take an action(a)?

We again average the state-values of both the states, added with an immediate reward which tells us how good it is to take a particular action(a). This defines our q_π(s,a).

Mathematically, we can define this as follows:

$$q_\pi(s,a) = R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a V_\pi(s') \qquad \text{...............(4)}$$

Equation (4) defines how good it is to take a particular action (a) in state(s). Where P is the transition probability. Now let us stitch these backup diagrams to define the State-Value Function, Vπ(s):

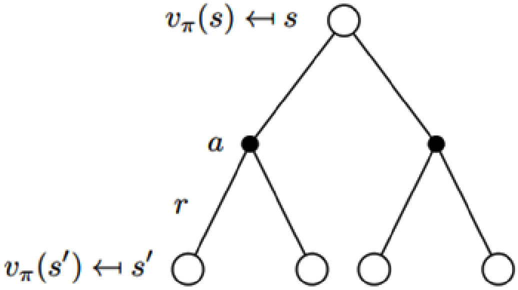

$v_\pi(s) \leftarrow\!\shortmid s$
a
r
$v_\pi(s') \leftarrow\!\shortmid s'$

***Figure 5.4**: Backup Diagram for State-Value Function*

Figure 5.4 shows if our agent is in some state(s) and from that state suppose our agent can take two actions due to which the environment might take our agent to any of the states(s′). Note that the probability of the action our agent might take from state ‘s′ is weighted by our policy and after taking that action the probability that we land in any of the states(s′) is weighted by the environment.

Now the question is, how good it is to be in state(s) after taking some action and landing on another state(s′) and following our policy(π) after that?

It is like what we have done before; we are going to average the value of successor states(s′) with some transition probability (P) weighted with our policy.

Mathematically, we can define the State-Value function for being in state S in Backup Diagram as follows:

$$V_\pi(s) = \sum_{a \in A} \pi(a|s) \left(R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a V_\pi(s') \right) \quad \text{.......................(5)}$$

Now, let us do the same for the State-Action Value Function, :

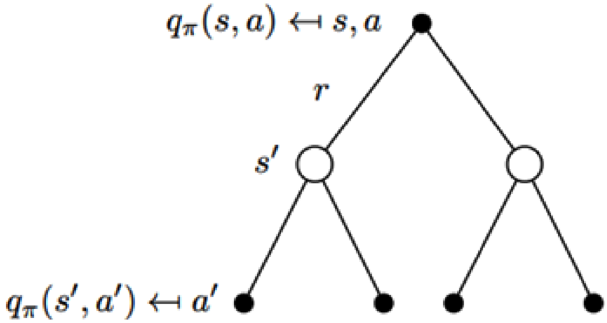


***Figure 5.5**: Backup Diagram for State-Action Value Function*

It is like what we did in the State-Value Function but just it is inverse. *Figure 5.5* tells us that our agent takes some action(a) because of which the environment might land us on any of the states(s), then from that state we can choose to take any actions(a′) weighted with the probability of our policy(π). Again, we average them together and that gives us how good it is to take a particular action following a particular policy(π) all along.

Mathematically, the State-Action Value Function from the backup diagram can be expressed as follows:

$$q_\pi(s,a) = R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a \sum_{a' \in A} \pi(a'|s')\, q_\pi(s',a')| \quad \text{..............(6)}$$

So, this is how we can formulate Bellman Expectation Equation for a given MDP to find its State-Value Function and State-Action Value Function.

Monte Carlo Methods

Any method which takes care of an issue by producing reasonable random numbers and seeing that fraction of numbers complying with some property or properties can be named a Monte Carlo technique. Monte Carlo techniques require experiences like the example arrangements of states, actions, and rewards from on-line or communication with an environment. Gaining from the online experience is striking since it requires no prior information on the environment's elements yet can at present accomplished ideal conduct. Learning from simulated experience is likewise amazing.

Monte Carlo methods are ways of solving the reinforcement learning problem based on averaging sample returns. To guarantee that well-defined returns are available, we characterize Monte Carlo strategies just for episodic tasks. That is, we accept that experience is separated into episodes, and that all episodes eventually terminate no matter what actions are chosen. It is just upon the culmination of an episode that the value estimates and policies are changed. Monte Carlo strategies are in this manner incremental in an episode-by-episode sense, yet not in a step-by-step sense. The term "*Monte Carlo*" is often used more broadly for any estimation method whose activity includes a significant random component.

Monte Carlo Policy Evaluation

We start by considering Monte Carlo techniques for learning the state-value function for a given policy. Review that the estimation of a state is the expected return-expected combined future discounted reward - beginning from that state. A conspicuous method to evaluate it for a fact, at that point, is basically to average the returns observed after visits to that state. As more returns are seen, the average should converge to the expected value. This thought underlines all the Monte Carlo techniques.

Specifically, assume that we wish to estimate V^π (s), the value of a state ‘s’ under policy π, given a set of episodes acquired by following π and passing through ‘s’. Each occurrence of state ‘s’ in an episode is known as a visit to ‘s’. The every-visit MC technique estimates V^π (s) as the average of the returns following all the visits to s in a set of episodes. Inside a given episode, the first time ‘s’ is visited as the primary visit to s. The first-visit MC method averages just the returns following first visits to s. These two Monte Carlo techniques are fundamentally the same as however have marginally extraordinary hypothetical properties.

First Visit Monte Carlo: Average returns just for the first time s is visited in an episode.

Here is a step-by-step perspective on how the algorithm works:

1. Initialize the policy, state-value function.
2. Start by generating an episode according to the current policy.
    a) Keep track of the states experienced through that episode.
3. Select a state in 2.a
    a) Add to a list the return received after the first occurrence of this state.
    b) Average over all returns.
    c) Set the value of the state as that computed average.
4. Repeat step 3.
5. Repeat 2-4 until satisfied.

Every visit to Monte Carlo: Average returns for every time s is visited in an episode.

For this algorithm, we just change step 3.a of **First Visit to Monte Carlo** algorithm to '*Add to a list the return received after every occurrence of this state*'.

Here is a step-by-step view of how the algorithm works:

1. Initialize the policy, state-value function.
2. Start by generating an episode according to the current policy.
    a) Keep track of the states encountered through that episode.
3. Select a state in 2.a
    a) Add to a list the return received after every occurrence of this state.
    b) Average overall returns.
    c) Set the value of the state as that computed average.
4. Repeat step 3.
5. Repeat 2-4 until satisfied.

Let us consider a simple example to understand this concept. Suppose there is an environment where we have 2 states - A and B. Let us say we observed 2 sample episodes:

A + 3 -> A + 2 -> B – 4 -> A + 4 -> B – 3 -> terminate.

B – 2 -> A + 3 -> B – 3 -> terminate.

A+3 => A indicates a transition from state A to state A, with a reward +3. Let us find out the value function using both the methods:

First Visit to Monte Carlo	Every visit to Monte Carlo
V (A) = ½ (2 + 0) = 1 V (B) = ½ (-3 + -2) = -5/2	V (A) = ¼ (2 + -1 + 1 + 0) = ½ V (B) = ¼ (-3 + -3 + -2 + -3) = -11/4

Table 5.1: *Example first and every visit Monte Carlo*

Policy iteration and value iteration

In policy iteration, a policy is chosen at random and a value function corresponding to it is found, then a new (improved) policy is found based on the previous value function, and so on until an optimal policy is found.

In value iteration, a value function is chosen at random, then a new (improved) value function is found in an iterative process until the optimal value function is found, and then optimal policy is derived from that optimal value function.

Policy iteration is based on the principle of "policy evaluation —-> policy improvement."

The value iteration method is based on the principle of "optimal value function —-> optimal policy."

Policy Iteration

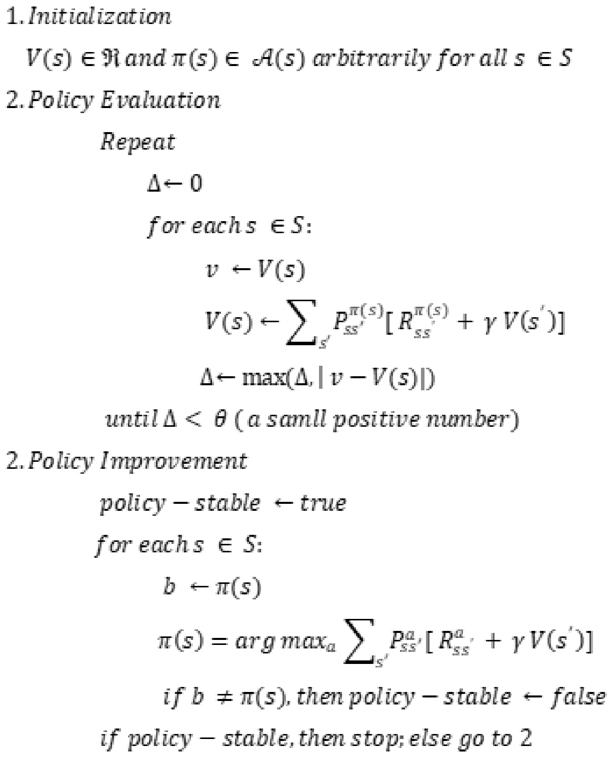
Once a policy, π, has been improved using to yield a better policy, π', we can then compute $V^{\pi'}$ and improve it again to yield an even better π''. We can thus obtain a sequence of monotonically improving policies and value functions as shown in equation (7):

$$\pi_0 \xrightarrow{E} V^{\pi_0} \xrightarrow{I} \pi_1 \xrightarrow{E} V^{\pi_1} \xrightarrow{I} \pi_2 \xrightarrow{E} \ldots\ldots\ldots \xrightarrow{I} \pi^* \xrightarrow{E} V^* \qquad \ldots\ldots\ldots (7)$$

Where → denotes a policy evaluation and → denotes a policy improvement. Each policy is guaranteed to be a strict improvement over the previous one (unless it is already optimal). Because a finite MDP has only a finite number of policies, this process must converge to an optimal policy and optimal value function in a finite number of iterations.

This way of finding an optimal policy is called policy iteration. The complete algorithm is given below in *figure 5.6*. Note that each policy evaluation, itself an iterative computation, is started with the value function for the previous policy. This typically results in a great increase in the speed of convergence of policy evaluation (presumably because the value function changes a little from one policy to the next).

Here’s a step-by-step Policy iteration (using iterative policy evaluation) for V^* :



```
1. Initialization
  $V(s) \in \Re$ and $\pi(s) \in \mathcal{A}(s)$ arbitrarily for all $s \in S$
2. Policy Evaluation
        Repeat
            $\Delta \leftarrow 0$
            for each $s \in S$:
                $v \leftarrow V(s)$
                $V(s) \leftarrow \sum_{s'} P_{ss'}^{\pi(s)}[R_{ss'}^{\pi(s)} + \gamma V(s')]$
                $\Delta \leftarrow \max(\Delta, | v - V(s)|)$
        until $\Delta < \theta$ ( a samll positive number)
2. Policy Improvement
        policy − stable ← true
        for each $s \in S$:
            $b \leftarrow \pi(s)$
            $\pi(s) = arg\,max_a \sum_{s'} P_{ss'}^{a}[R_{ss'}^{a} + \gamma V(s')]$
            if $b \neq \pi(s)$, then policy − stable ← false
        if policy − stable, then stop; else go to 2
```

***Figure 5. 6**: Policy iteration (using iterative policy evaluation*

Policy iteration often converges in surprisingly few iterations. The policy improvement theorem assures us that these policies are better than the original random policy. In this case, however, these policies are not simply better, but optimal, proceeding to the terminal states in the minimum number of steps. In this example, policy iteration would find the optimal policy after one iteration.

Value Iteration

One drawback of the policy iteration is that each of its iterations involves policy evaluation, which may itself be a protracted iterative computation requiring multiple sweeps through the state set. If policy evaluation is done iteratively, then convergence exactly to occurs only in the limit. Must we wait for exact convergence, or can we stop short of that?

The policy evaluation step of policy iteration can be truncated in several ways without losing the convergence guarantees of policy iteration. One important case is when policy evaluation is stopped after just one sweep (one backup of each state).

This algorithm is called value iteration. It can be written as a particularly simple backup operation that combines the policy improvement and truncates policy evaluation steps:

$$V_{k+1}(s) = \max_{a} E\{r_{t+1} + \gamma\, V_k(S_{t+1}) \mid S_t = s, a_t = a\}$$

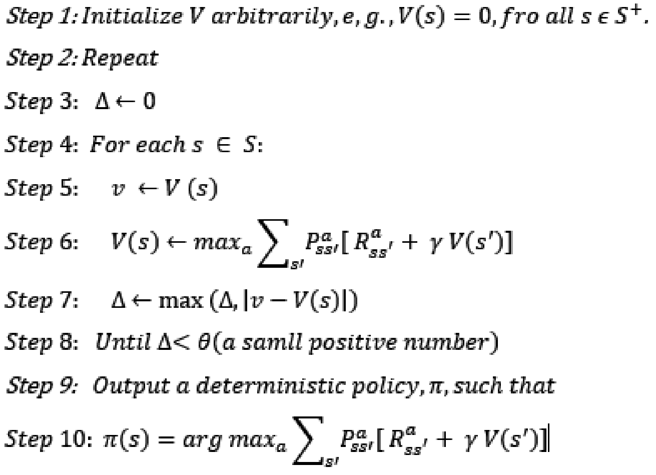
$$\ldots\ldots\ldots\ldots (8)$$

$$V_{k+1}(s) = \max_{a} \sum_{s'} P^{a}_{ss'} \left[R^{a}_{ss'} + \gamma\, V_k(s')\right]$$

For all s∈S. For arbitrary , the sequence {V_k} can be shown to *V** converge to under the same conditions that guarantee the existence of *V**.

Like policy evaluation, value iteration formally requires an infinite number of iterations to converge exactly to *V**. In practice, we stop once the value function changes by only a small amount in a sweep. *Figure 5.7* gives the complete value iteration algorithm with this kind of termination condition.

Value iteration effectively combines, in each of its sweeps, one sweep of policy evaluation and one sweep of policy improvement. Faster convergence is often achieved by interposing multiple policy evaluation sweeps between each policy improvement sweep. In general, the entire class of the truncated policy iteration algorithms can be thought of as sequences of sweeps, some of which use the policy evaluation backups and some of which use the value iteration backups. Since the max operation is the only difference between these backups, this just means that the max operation is added to some sweeps of policy evaluation.



```
Step 1: Initialize V arbitrarily, e, g., $V(s) = 0$, fro all $s \in S^+$.
Step 2: Repeat
Step 3:  $\Delta \leftarrow 0$
Step 4:  For each $s \in S$:
Step 5:     $v \leftarrow V(s)$
Step 6:     $V(s) \leftarrow max_a \sum_{s'} P^a_{ss'}[R^a_{ss'} + \gamma V(s')]$
Step 7:     $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
Step 8:  Until $\Delta < \theta$(a samll positive number)
Step 9:  Output a deterministic policy, $\pi$, such that
Step 10: $\pi(s) = arg\ max_a \sum_{s'} P^a_{ss'}[R^a_{ss'} + \gamma V(s')]$
```

Figure 5.7*: Value iteration*

Q-Learning

Q-learning is an off-policy reinforcement learning algorithm. Trying to find the best action you can take given the current state. It is considered off-policy because the Q-learning function learns from the actions outside the current policy, such as taking random steps, so policy is not required. Q-learning wants to learn a policy that maximizes total rewards.

This is a value-based learning algorithm. Value-based algorithms update/adjust the value function based on an equation. Whereas, policy-based predict the value function with a greedy policy obtained from the last policy improvement.

The 'Q' in Q-learning stands for quality. Quality in this case represents how useful a given action is in gaining some future reward.

Introducing the Q-Table

Q-Table is just a fancy name for a simple lookup table. Q-Table is the data structure used to calculate the maximum expected future rewards for action at each state. This table will guide us to the best action in each state. To learn each value of the Q-table, a Q-Learning algorithm is used.

Q-table or matrix follows the shape of [state, action] and we initialize all values to zero. We then update and store Q-values after an episode. This Q-table becomes a reference table for the agent to select the best action based on the Q-value.

In the Q-Table, the columns are the actions, and the rows are the states.

To learn each value of the Q-table, we use the *Q-Learning algorithm*.

The Q-Learning algorithm

Q-function: The Q-function uses the Bellman equation and takes two inputs: state (s) and action (a).

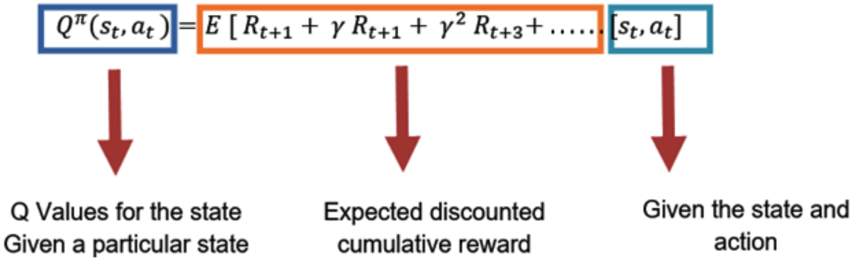


Figure 5.8: Q-function

Using the above Q-function (*Figure 5.8*, we get the values of Q for the cells in the table. In the beginning, all the values in the Q-table are zeros. There is an iterative process of updating the values. As we start to explore the environment, the Q-function gives us better and better approximations by continuously updating the Q-values in the table.

Q-learning algorithm process: Let us understand the Q-learning algorithm with each step in detail as shown in *figure 5.9* as follows:

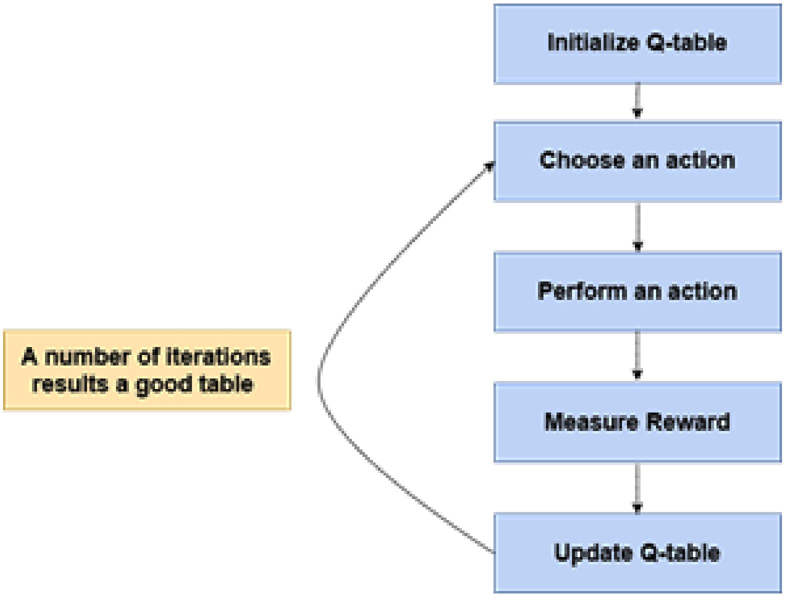


***Figure 5.9**: Q-learning Algorithm*

Step 1: Initialize the Q-Table

Here we first create a Q-table. There are n columns, where n = number of actions. There are m rows, where m = number of states. We will initialize the values at 0.

Steps 2 and 3: Choose and perform an action

The combination of steps 2 and 3 is performed for an undefined amount of time. These steps run until the time training is stopped, or when the training loop is stopped as defined in the code.

First, an action (a) in the state (s) is chosen based on the Q-Table. Note that, as mentioned earlier when the episode initially starts, every Q-value should be 0. Then, update the Q-values for being at the start and moving right using the Bellman equation which has been explained above.

Steps 4: Measure reward

Now we have taken an action and observed an outcome and reward.

Steps 5: Evaluate

We need to update the function Q (s, a). This process is repeated until the learning is stopped. In this way, the Q-Table is updated and the value function Q is maximized. Here, the Q (state, action) returns the *expected future reward* of that action at that state.

$$New\ Q(s,a) = Q(s,a) = [R(s,a) + maxQ'(s',a') - Q(s,a)] \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ (9)$$

Where,

New Q(s,a) : New Q value for that state and the action

Q(s,a) : Current Q values

α : Learning Rate

R(s,a) : Reward for taking thar action at that state

Y : Discount Rate

maxQ'(s',a'): Maximum expected future reward given the new state (s') and all possible actions at that new state.

State-Action-Reward-State-Action (SARSA)

SARSA algorithm is a slight variation of the well-known Q-learning algorithm. For a learning agent in any reinforcement learning algorithm, the policy can be of two types:

1. **On Policy**: In this, the learning agent learns the value function as indicated by the current action derived from the policy currently being used.
2. **Off Policy**: In this, the learning agent learns the value function according to the action derived from another policy.

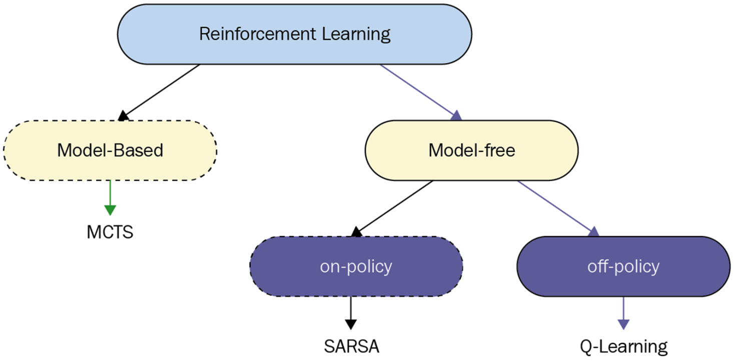


***Figure 5.10**: Types of policy in the Reinforcement Learning*

Q-Learning technique is an **Off-Policy** method and uses the greedy way to learn the Q-value. SARSA technique, on the other hand, is an **On-Policy** method that uses the action performed by the current policy to learn the Q-value as shown in *Figure 5.10*.

SARSA is an on-policy algorithm where, in the current state, S an action, A is taken and the agent gets a reward, R and ends up in next state, S′ and takes action, A′ in S′. Therefore, the tuple (S, A, R, S′, A′) stands for the acronym **SARSA** as shown in *Figure 5.11*.

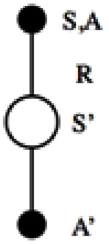


***Figure 5.11**: SARSA*

It is called an on-policy algorithm because it updates the policy based on actions taken.

SARSA Vs Q-learning

The significant distinction between SARSA and Q-learning is that the maximum reward for the following state ′is not utilized for updating the Q-values. Rather, a new action, and in this manner reward, is chosen utilizing the same policy that decided the original action.

In SARSA, the agent begins in state 1, performs action 1, and gets a reward (reward 1). Presently, it is in state 2 and plays out another action (action 2) and gets the reward from this state (reward 2) preceding it returns and updates the estimation of activity 1 acted in state 1. In contrast, in Q-learning, the agent begins in state 1, performs action 1 and gets a reward (reward 1), and then looks and sees what the maximum possible reward for an action is in state 2, and uses that to update the action value of performing action 1 in state 1. So, the difference is in the way the future reward is found. In Q-learning, it is simply the highest possible action that can be taken from state 2, and in SARSA it is the value of the actual action that was taken.

SARSA Algorithm

We will choose the current action At and the next action A(t+1) using the same policy. Thus, in the state S(t+1), its action will be A(t+1) which is selected while updating the action-state value of St.

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\, [R_{t+1} + \gamma\, Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t) \quad \text{............ (10)}$$

The algorithm for SARSA is a little bit different from Q-learning as described as follows:

```
$Step\ 1$:  $Initialize\ Q\ (s, a)\ for\ all\ s\ \epsilon\ S, a\ \in A\ (s), arbitrarily, and\ Q(terminal - state) = 0$
$Step\ 2$:  $Repeat\ (for\ each\ episode)$:
$Step\ 3$:  $Initialize\ S$
$Step\ 4$:  $Choose\ A\ from\ S\ using\ policy\ derived\ from\ Q\ (e.g., \epsilon - greedy)$
$Step\ 5$:  $Repeat\ (for\ each\ step\ of\ episode)$:
$Step\ 6$:  $Take\ action\ A, observe\ R, S'$
$Step\ 7$:  $Choose\ A'\ from\ S'\ using\ policy\ derived\ Q\ (e.g., \epsilon - greedy)$
$Step\ 8$:  $Q(S, A)\ \leftarrow Q\ (S, A) + \alpha\ [R + \gamma\ A(S', A') - Q(S, A)]$
$Step\ 9$:  $S\ \leftarrow S'; A \leftarrow A'$;
$Step\ 10$: $Until\ S\ is\ terminal$
```

The Q-value is updated taking into account the action, A1 performed in the state, S1 in SARSA as opposed to Q-learning where the action with the highest Q-value in the next state, S1 is used to update Q-table.

Model-based Reinforcement Learning

We should consider the example of a chess game: Assume we are playing a chess match, we made a move, by then we get held up till our enemy makes his move, by then we make another move, and our adversary follows with his, by then we find that our first move was not unreasonably adequate taking everything into account! If we are learning regularly, our teacher will let us move back the moves so we can pick up from our misstep. Notwithstanding, that we have lost a long while back.

On the other hand, if we reproduce the moves in our brain (which is what everyone does) and tell our self *"in case I do this move my enemy can counter with this move, by then I do this move, etc."* we would avoid all the previous circumstance. We are truly doing that we are unfurling in our brain a hunting tree dependent on the model we think about chess, and from this tree, we will pick the best move that will possibly provoke winning. Presently supplant ourselves by an AI specialist, and we get a Model-Based Reinforcement Learning.

We can see how this will spare training time. Clearly, it won't be apparent in small conditions with high reactivity (Grid World for example), anyway for complex situations, for instance, any Atari game learning by methods for model-free RL systems is dull, while on the other hand making a decreased game plan of exercises makes a model, by then to use this model to interpret scenes is fundamentally more proficient.

What is a Model?

Hypothetically, a model is our representation of reality or the environment that we are in. In RL the model is having a representation M of the MDP [S, A, P, R]. This implies having a version I (should be as exact as could be expected under the circumstances) of the genuine MDP.

If we expect that the states space S and the transition probabilities A are understood the model Mi will become [S, A, Pi, Ri]. So according to the model Mi going from state S to ′′ in the wake of performing activity A, is dependent upon the likelihood Pi (′′ | S, An), equivalently having a prize ′′ when at state S and performing activity A is reliant upon the association Ri (′′ | S, A).

Difference between Model-Based and Model-Free

Model-free strategies adapt straightforwardly for the experience; this suggests they perform activity either in real (ex: robots) or on PC (ex: games). By then they accumulate the reward from the environment, whether positive or negative, and they update their worth capacities.

This is a key differentiation with a model-based strategy. Model-Free procedures act in the real environment to learn. On the other hand, the model-based technique uses a diminished number of collaborations with the real environment during the learning stage. Its point is to construct a model reliant on these connections, and later use this model to reproduce the further scenes, not in the genuine condition anyway but by applying them to the created demonstration and get the results returned by that model.

As depicted already, this has the upside of speeding the learning, since there is no convincing motivation to believe that the environment will respond nor to reset nature to some state in to keep learning. On the downside, however, if the model is incorrect, we take hazard in something not the same as the truth. Another significant point is that the model-based technique will regardless use model-free systems either to construct the model or in the arranging/recreation.

Learning the Model

Learning the model contains executing activities in the real environment and accumulate feedback. We call this experience. Along these lines, for each state and activity, the environment will give a new state and reward. Because of these collections of experiences, we try to deduce the model. As one can figure, this is not other than supervised learning as shown here:

$S_1, A_1 \rightarrow R_2, S_2$

$S_2, A_2 \rightarrow R_3, S_3$

.

.

.

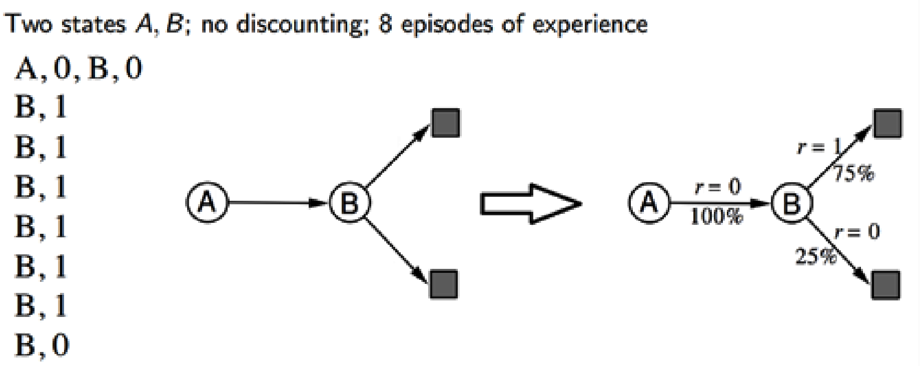
$S_{T-1}, A_{T-1} \rightarrow R_T, S_T$

We take care of this issue by utilizing one of the supervised learning methods that are accessible, it tends to be a regression or neural networks or something different. Depending on the supervised learning technique, the model will be represented by a table lookup, neural network, or others.

Concrete Example

The example in *Figure 5.12* represents two states A and B with transitions from A to B and from B to two possible terminal states. Here we assume there is no discount factor.

To manufacture the model, we run a few episodes in the real environment, we gather the landing states and the outcomes. Afterwards, we deduce the [P, R] of the model as demonstrated as follows:



Two states *A*, *B*; no discounting; 8 episodes of experience

A, 0, B, 0

B, 1

B, 1

B, 1

B, 1

B, 1

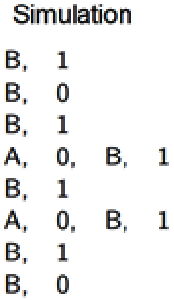
B, 1

B, 0

Figure 5.12*: An example of the Supervised learning*

We can see that as per our experience (the tests we run) going from A to B is 100% with 0 rewards, while from B to the upper terminal state is 75% of the time with reward 1, and to the lower terminal state 25% of the time with reward n 0.

This tells that on the off chance that we make a reproduction utilizing this model (not the real environment) and we run an arbitrary set of tests each time we are at A, the model will disclose to us that we will move to B 100% of the time with r = 0 and each time we are at B we can go up 75% of the time with r = 1 and 25% of the time with r =0.



Simulation

```
B,  1
B,  0
B,  1
A,  0,  B,  1
B,  1
A,  0,  B,  1
B,  1
B,  0
```

Figure 5.13: *A Simulation example of the Supervised learning*

Now, we can compute the value function utilizing a technique, for example, Monte Carlo which leads our example to V(A) = 1 and V(B) = .75. As a remainder of Monte Carlo value computation:

$G(s) = r + \boldsymbol{\gamma}\, G(s')$ *where G is the return at state 's' after every episode.*
$V(s) =$ *average* $G(s)$ *for all episodes*

$V(B) = (6 * G1 + 2G2) / 8 = (6 * 1 + 2 * 0) / 8 = .75$
$V(A) = ([G(A) + G(B)] + [G(A) + G(B)]) / 2 = ((0 + 1) + (0 + 1)) / 2 = 1$

We considered there is no discounting so $\boldsymbol{\gamma} = 1$.

The primary loop of Model-Based RL **(**see *Figure 5.14***)** is as per the following:

- Act in the real environment, gather experience (states and rewards).
- Deduce a model and use it to produce samples (planning).
- Update the value functions and policies from samples.
- Use these value functions and policies to choose actions to act in the real environment.

- Restart the loop.

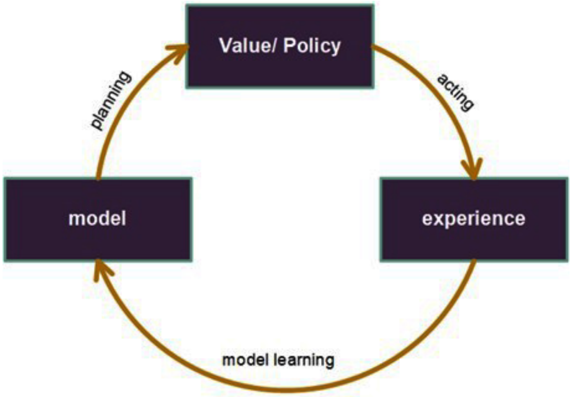


***Figure 5.14**: The primary loop of Model-Based RL*

Dyna Architecture

It is a variation of the Model-Based RL, called **Dyna Architecture** as appeared in *Figure 5.15*. Rather than utilizing the genuine experience to develop a model, it is additionally used to update the value functions.

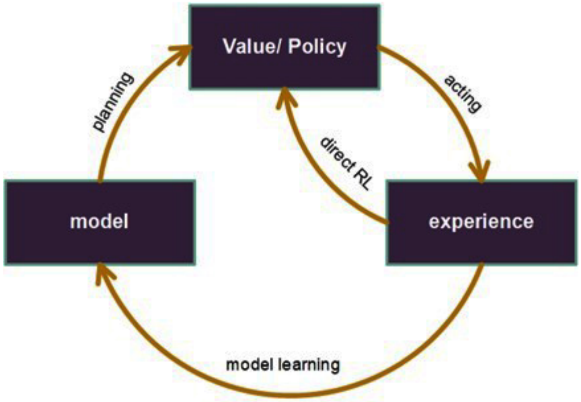


***Figure 5.15**: Dyna Architecture*

Summary

- The Markov chain is a probabilistic model that solely depends upon the current status and not on the previous states.
- Markov's choice procedure is a model for foreseeing results.

- Markov Decision Process (MDP) is characterized by:
  - o State
  - o Model or change work
  - o Action
  - o Reward
  - o Policy
- Applications of Markov Decision Process: hybrid energy storage system, traffic signal control, generating explanations, remote experimental setups for evaluation, speech to the speech interaction system, a leak detection system, a learning design recommendation system, and so on.
- Bellman equation is the basic block of solving reinforcement learning and is omnipresent in RL.
- Mathematically, we can define Bellman Expectation Equation as:

$$v_\pi(s) = \mathbb{E}_\pi \left[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s\right]$$

- Monte Carlo methods are ways of solving the reinforcement learning problem based on averaging sample returns.
- The term "*Monte Carlo*" is often used broadly for any estimation method whose activity includes a significant random component.
- First Visit to Monte Carlo: Average returns just for first the time s is visited in an episode.
- Every visit to Monte Carlo: Average returns for every time s is visited in an episode.
- Policy Iteration:

$$\pi_0 \xrightarrow{\text{E}} V^{\pi_0} \xrightarrow{\text{I}} \pi_1 \xrightarrow{\text{E}} V^{\pi_1} \xrightarrow{\text{I}} \pi_2 \xrightarrow{\text{E}} \cdots \xrightarrow{\text{I}} \pi^* \xrightarrow{\text{E}} V^*$$

where $\xrightarrow{\text{E}}$ denotes a policy evaluation and $\xrightarrow{\text{I}}$ denotes a policy improvement.

- Each policy evaluation, itself an iterative computation, is started with the value function for the previous policy.
- The policy improvement theorem assures us that these policies are better than the original random policy.
- One drawback of the policy iteration is that each of its iterations involves policy evaluation, which may itself be a protracted iterative computation requiring multiple sweeps through the state set.

- Value Iteration:

$$\begin{aligned} V_{k+1}(s) &= \max_a E\left\{r_{t+1} + \gamma V_k(s_{t+1}) \mid s_t = s, a_t = a\right\} \\ &= \max_a \sum_{s'} \mathcal{P}^a_{ss'} \left[\mathcal{R}^a_{ss'} + \gamma V_k(s')\right], \end{aligned}$$

  for all $s \in \mathcal{S}$. For arbitrary V_0, the sequence $\{V_k\}$ can be shown to converge to V^*under the same conditions that guarantee the existence of V^*.

- Q-learning is an off-policy reinforcement learning algorithm.

- Value-based algorithms update the value function based on equations. Whereas, policy-based predict the value function with a greedy policy obtained from the last policy improvement.

- The 'Q' in Q-learning stands for quality. Quality in this case represents how useful a given action is in gaining some future reward.

- Q-Table is the data structure used to calculate the maximum expected future rewards for action at each state.

- The Q-learning algorithm process:
  - **Step 1**: Initialize Q-Table
  - **Step 2**: Choose an action
  - **Step 3**: Perform an action
  - **Step 4**: Measure Reward
  - **Step 5**: Evaluate/Update Q-table

- State-Action-Reward-State-Action

- SARSA algorithm is a slight variation of the well-known Q-Learning algorithm.

- Q-Learning technique is an Off-Policy method and uses the greedy way to learn the Q-value.

- SARSA is an on-policy algorithm where, in the current state, S an action, A is taken and the agent gets a reward, R and ends up in next state, S' and takes action, A' in S'. Therefore, the tuple (S, A, R, S', A') stands for the acronym SARSA.

- Model-free methods learn directly from the experience; this implies they act in reality.

- Model-Based method utilizes fewer interactions with the real environment during the learning stage.
- Learning the model comprises of executing actions in the real environment and gather feedback.
- A variation of the Model-Based RL is called Dyna Architecture.

Exercise (MCQs)

1. **What are the essential elements in a Markov Decision Process?**

   a) State, Model, Probabilities, Station

   b) State, Modeling, Action, Reward

   c) Model, Line, Predict, Reward

   d) State, Action

2. **In Q-Learning-reinforcement learning, Q- represents:**

   a) Quality

   b) Quantity

   c) Quantitative

   d) Qualitative

3. **Semi-supervised learning may refer to either inductive or ______________ learning.**

   a) Deductive Learning

   b) Unseen Learning

   c) Transductive Learning

   d) Reinforcement Learning

4. **SARSA acronym stands for:**

   a) State-Action-Reward-State-Action

   b) State-Action-Renew-State-Action

   c) State-About-Reward-State-Action

   d) State-Action-Reward-State-Actionary

5. **The Q-learning algorithms is having ________ steps.**

   a) Two

   b) Three

   c) Four

   d) Five

6. **The Markov's choice procedure is a model for ____________ results.**
   a) Model
   b) Solve
   c) Action
   d) Foreseeing

7. **For a learning agent in any reinforcement learning algorithm the policy can be of two types:**
   a) On-Policy and Off-Policy
   b) Off-Policy and Off-Policy
   c) On-Policy and On-Policy
   d) None of these

8. **Q-Learning technique is an _______________ method.**
   a) Off-Policy method
   b) On-Policy method
   c) On-Off Policy method
   d) Off-Off Policy method

9. **MDP acronym stands for:**
   a) Markov Decision Procedure
   b) Markov Decision Problem
   c) Markov Decision Process
   d) Markov Decide Process

10. **Reinforcement learning is used for:**
   a) Game theory
   b) Control theory
   c) Information theory
   d) All of the above

Answers

1. **(d)** State, Action
2. **(a)** Quality
3. **(c)** Transductive Learning
4. **(a)** State-Action-Reward-State-Action
5. **(d)** Five

6. **(d)** Foreseeing
7. **(a)** On-Policy and Off-Policy
8. **(a)** Off-Policy method
9. **(c)** Markov Decision Process
10. **(d)** All of the above

Descriptive Questions

1. Define Markov Decision Process with the help of a diagram.
2. Write some applications of the Markov Decision Process.
3. What is the purpose of using the Bellman Equation? Explain the Bellman Equation for Value Function.
4. Why are the Monte Carlo Methods used? Explain with an example.
5. Differentiate between First Visit Monte Carlo Vs Every Visit Monte Carlo.
6. Write the implementation of policy iteration and value iteration agents for taxi game.
7. What is Q-learning? Explain the Q-learning algorithm process.
8. Differentiate between the Q-Learning and SARSA techniques.
9. Write the SARSA algorithm.
10. Differentiate between model-based and model-free methods.

CHAPTER 6

Recommended Systems

> **"I think people need to understand that deep learning is making a lot of things, behind the scenes, much better"**
>
> *– Sir Geoffrey Hinton*

Introduction

Recommended Systems (**RS**) are software tools and techniques that provide recommendations for items that may be useful to a user. This chapter explores the various recommended system techniques and the introduction of deep learning.

Structure

In this chapter, we will discuss the following topics:

- Recommended systems
- Collaborative filtering
- Content-based filtering
- Artificial Neural Network (ANN)
- Perceptron
- Multilayer network

- Back propagation algorithm
- Introduction to deep learning

Objective

By reading this chapter, you will:

- Understand the concept of the recommended systems with real-time application uses.
- Understand the collaborative and content-based filtering.
- Understand the Artificial Neural Network (ANN).
- Understand the backpropagation algorithm.
- Understand the concept of deep learning with real-time examples.

Recommended Systems

Recommended Systems (**RS**) are software tools and techniques that provide recommendations for items that may be useful to a user. The recommendations are for various decision-making processes, such as what things to buy, what music to listen to, or what online news to read. The term "*things*" refers to what the system recommends to users as a whole. An RS typically focuses on a specific type of thing (for example, CDs or news). As a result, its design, graphical user interface, and core recommendation technique to generate recommendations are all tailored to provide valuable and practical recommendations for that specific type of item. RSs are primarily aimed at people who lack the personal experience or competence to evaluate the potentially overwhelming number of alternative items available on a website. A book recommender system, for example, can help users choose a book to read. *Amazon.com*, a popular website, uses an RS to personalize the online store for each customer. Recommendation systems are usually personalized; different users or user groups receive a variety of recommendations.

Personalized recommendations are presented in their most basic form as ranked lists of items. RSs attempt to predict the most appropriate products or services based on the user's preferences and constraints while performing this ranking.

RS emerged from the observation that people rely on recommendations from others while making routine, daily decisions. When choosing a book to read, it is common to rely on what one's peers recommend.

Recommendations are given for items liked by similar users (those with similar tastes). While having options is beneficial, having more options is not always preferable - electronic commerce websites are designed to provide recommendations based on filtering the entire range of available alternatives. Instead of being beneficial, the availability of options began to reduce users' well-being.

Recently, RS has proven to be a valuable tool for dealing with information overload. Finally, an RS addresses this phenomenon by directing a user to new, unexplored items relevant to the user's current task. RS generate recommendations based on various types of knowledge and data about the users, available items, and previous transactions stored in the customized databases in response to a user's request, which can be articulated by the user's context and need, depending on the recommendation approach. The user can then scroll through the recommendations. These user actions and feedbacks can be saved in the recommender database and can be used to generate new recommendations in subsequent user-system interactions. As previously stated, research in recommender systems is relatively new when compared to research in other traditional information system tools and techniques (for example, databases, and search engines).

Importance of recommended systems

As evidenced by the following facts, interest in the recommender systems has gone up significantly:

- Popular websites such as *Amazon.com*, *YouTube*, and *Netflix* make extensive use of RS. Many media organizations are now incorporating RS into their services.
- The ACM recommender systems conference is the premier annual event for researchers and practitioners of recommender technology. Sessions on RS are frequently included in more traditional conferences on databases and information systems.
- Undergraduate and graduate courses are now entirely dedicated to RS at institutions of higher learning.
- Several special issues of academic journals have been published that cover research and development in the field of RS. They are - AI Communications, IEEE Intelligent Systems, International Journal of Electronic Commerce, International Journal of Computer Science and Applications, ACM Transactions on Computer-Human Interaction, and ACM Transactions on Information Systems.

Types of recommended systems

An RS must anticipate that an item is worth recommending to carry out its core function of identifying useful items for the user. For example, a popular song, that is, something that is liked (high utility) by many users, will most likely be liked by a generic user as well. The utility of these popular songs is expected to be fairly high for an average user.

As previously stated, some recommender systems do not fully estimate utility before making a recommendation, but they may use heuristics to hypothesize that an item is useful to a user. It is also worth noting that the user utility of an item has been observed to be dependent on other variables, which we refer to as "*contextual*." For example, the utility of an item for a user can be influenced by the user's domain knowledge (for example, expert vs. beginning digital camera users) or by the time when the recommendation is requested.

The most basic and original implementation of this approach suggests to the active user items that other users with similar tastes have liked in the past. The similarity in taste between two users is calculated based on the users' rating history. This is why collaborative filtering is referred to as "people-to-people correlation". Collaborative filtering is widely regarded as the most popular and widely used technique in RS. Key examples are as follows:

- **Amazon:** Product recommendations
- **Netflix:** Movie and TV show recommendations
- **News websites:** Popular news and article recommendations

There are three types of recommendation approaches as shown in *Figure 6.1*:

- **Content-based methods**: This recommended system uses item attributes. The system learns to recommend items that are similar to those that the user has previously liked. The similarity of items is calculated using the features of the compared items. For example, if a user gives a positive rating to a comedy film, the system can learn to recommend other films in that genre.
- **Collaborative filtering methods**: This recommended system uses user behaviors (interactions) in addition to the item attributes.

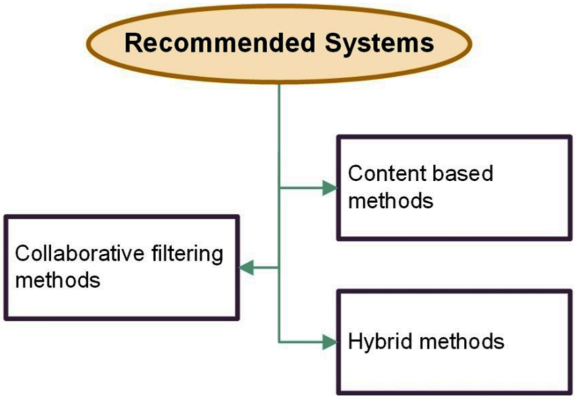


***Figure 6.1**: Different types of recommended systems*

- **Hybrid methods**: Hybrid RS is based on the combination of the content-based and collaborative filtering recommended systems. A hybrid system that combines techniques A and B attempts to use the benefits of A to compensate for the shortcomings of B.

Collaborative filtering methods, for example, suffer from new-item issues, that is, they cannot recommend items with no ratings. This does not limit content-based approaches because new item prediction is based on their description (features), which are typically readily available. In a temporal context, for example, vacation recommendations in winter should be very different from those in summer. Similarly, a restaurant recommendation for a Saturday evening with friends should differ from the one for a workday lunch with co-workers.

Recommended Systems (RS) functions

We defined RS in the previous section as software tools and techniques that provide users with suggestions for items they may want to use. Now we want to refine this definition by illustrating a variety of possible roles for an RS. First and foremost, we must distinguish between the role of the RS on behalf of the service provider and that of the RS user. For example, a travel intermediary (for example, *Expedia.com*) or a destination management organization (for example, *Visitfinland.com*) may introduce a travel recommender system to increase its turnover (Expedia), i.e., sell more hotel rooms, or to increase the number of tourists to the destination. When visiting a destination, the user's primary motivation for accessing the two systems is to find a suitable hotel and interesting events/attractions.

Following are the RS functions:

(a) **Increase the number of items sold**: This is likely the most significant function for a commercial RS, the ability to sell an additional set of items in addition to those normally sold without any kind of recommendation. This objective is met because the recommended items are likely to meet the user's needs and desires.

(b) **Sell more diverse items**: Another important function of an RS is to allow the user to select items that may be difficult to find without a specific recommendation. For example, in a movie rental service like Netflix, the service provider is interested in renting all of the DVDs in the catalog, not just the most popular ones. This may be difficult without an RS because the service provider cannot afford to risk advertising movies that are unlikely to appeal to a specific user's tastes.

(c) **Increase user satisfaction**: A well-designed RS can also strengthen the user's experience with the site or application. With a properly designed human-computer interaction, the user will find the recommendations interesting, relevant, and useful. The combination of effective, that is, accurate, recommendations and a user-friendly interface will improve the system's subjective evaluation by the user.

(d) **Increase user fidelity**: A user should be loyal to a website that recognizes an existing customer and treats him as a valuable visitor when they visit. This is a common feature of an RS because many RSs compute recommendations by leveraging information gleaned from previous interactions with the user, such as item ratings.

(e) **Better understand what the user wants**: Another important function of an RS that can be applied to a variety of different applications is the description of the user's preferences, which can be collected explicitly or predicted by the system. The service provider may then decide to re-use this knowledge for a variety of other purposes, such as improving stock management or production. In the travel domain, for example, destination management organizations can decide to advertise a specific region to new customer sectors or advertise a specific type of promotional message derived from RS data analysis (transactions of the users).

Applications and Challenges of Recommended Systems

The following are some of the major applications and challenges to the adoption of the recommended systems:

Applications

RS research has been conducted with a strong focus on practice and commercial applications, as it is generally directed at enhancing commercial RSs in addition to its theoretical concept. The developer of an RS for a specific application domain should understand the domain's specific facets, requirements, application challenges, and limitations.

We define more broad categories of fields for the most common recommended system applications based on these specific application domains.

- **Entertainment applications**: Movies and music recommendations.
- **Content-based applications**: Personalized newspapers, documents, web pages, e-learning, and e-mail filters recommendations.
- **E-commerce applications**: Products to buy such as books, cameras, PCs, and so on. recommendations.
- **Services applications**: Travel services, experts for consultation, houses to rent, and matchmaking services recommendations.

As recommended systems gain popularity, there is an increased interest in the potential benefits of new applications, such as recommending friends or tweets to follow, as in *www.facebook.com*. As a result, the preceding list cannot cover all of the application domains that are now being addressed by RS techniques; it merely provides an overview of the various types of application domains.

Challenges

The following are some of the major challenges to the adoption of the recommended systems:

- The algorithms' scalability with massive and real-world datasets. As core methodology research progresses and matures, it becomes clear that a fundamental issue for RSs is determining how to embed core recommendation techniques in real operating systems and how to deal with vast and dynamic sets of data generated by user interactions with items (ratings, preferences, reviews, and so on.).

- Proactive recommender systems, or recommenders that provide recommendations even when they are not explicitly requested. The vast majority of recommended systems developed thus far adhere to a "*pull*" model, in which the user initiates the request for a recommendation.

- Systems that protect your privacy are recommended. RSs use user data to make personalized recommendations.

- The variety of items recommended to a target user. When there is a certain degree of diversity among the included items in a recommendation list, the user is more likely to find a suitable item.

- Systems that are distributed and operate in open networks are recommended. The computational model of the vast majority of RSs follows a standard client-server architecture, in which the user-client requests recommendations from the server-recommender, who responds with the recommendations. This is a severe limitation, and it suffers from all of the traditional problems associated with centralized systems.

- A recommender that optimizes a series of recommendations. As previously stated, conversational RSs emerged in an attempt to improve the effectiveness of recommendations provided by systems using a simpler approach: a one-time request/response. Conversational RSs can be strengthened further by incorporating learning capabilities that can optimize not only the items that are recommended but also how the interaction between the actor and the computer must unfold in all possible situations.

- Recommenders built for mobile devices and usage contexts. Mobile computing is quickly gaining traction as the most natural platform for personal computing. Many recommendation requests are likely to be made while the user is on the move, such as while shopping or staying in a hotel in a new city. This necessitates "*mobilizing*" the user interface and designing computational solutions that can efficiently use the mobile devices' still limited resources (computational power and screen size).

Collaborative Filtering

It is based on historical data; collaborative filtering does not require anything other than the users' historical preferences on a set of items. The central assumption here is that users who have previously agreed tend to agree again in the future. In terms of user preference, it is usually expressed in two categories. An explicit rating is a rating given to an item by a user on a sliding scale, such as 5 stars for Titanic. This is the most direct way for users to express how much they like a product. Implicit Rating indicates a user's preference indirectly, such as page views, clicks, purchase records, whether or not to listen to a music track, and so on.

Collaborative filtering filters information by utilizing the system's interactions and data collected from other users. It is based on the assumption that people who agreed on certain items' evaluations are likely to agree again in the future. The idea is simple: when we're looking for a new movie to watch, we often ask our friends for recommendations. We naturally place more trust in recommendations from friends who have similar tastes to our own.

The so-called similarity index-based technique is used by the majority of collaborative filtering systems. Several users are chosen in the neighborhood-based approach based on their similarity to the active user. A weighted average of the ratings of the selected users is used to infer the active user. The relationship between users and items is the focus of collaborative-filtering systems. The similarity of items is determined by the similarity of their ratings by users who rated both items.

All around us, we see the use of recommendation systems. These systems personalize our web experience by telling us what to buy (*Amazon*), what movies to watch (*Netflix*), who to friend (*Facebook*), what songs to listen to (*Spotify*), and so on. These recommendation systems use our shopping/watching/listening habits to predict what we might like in the future based on our past behavior. The most fundamental models for recommendation systems are collaborative filtering models, which are based on the assumption that people like things that are similar to other things they like and things that other people with similar tastes like.

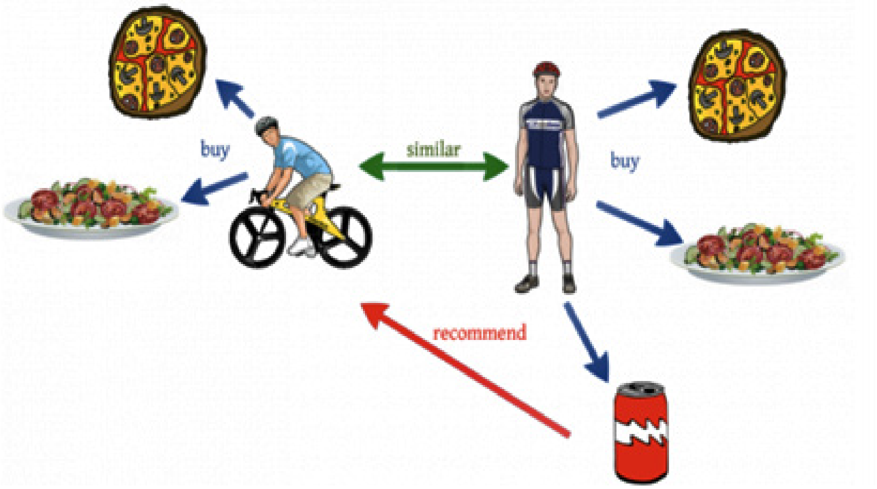


***Figure 6.2**: Example of collaborative filtering*
Source: https://towardsdatascience.com/various-implementations-of-collaborative-filtering-100385c6dfe0

Collaborative Filtering classes

The collaborative Filtering classes is divided into two types as follows:

- **User-based class**: This measures the similarity of target users to other users.
- **Item-based class**: This measures the similarity of items rated or interacted with by target users to other items.

Types of collaborative filtering techniques

The CF techniques are broadly divided into two types as shown in *Figure 6.3*:

- Memory-based approach
- Model-based approach

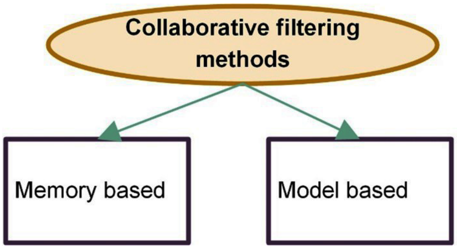


***Figure 6.3**: Types of collaborative filtering approaches*

- **Memory-based approach**: Memory-based CF approaches are classified into two categories: user-item filtering and item-item filtering as shown in *Figure 6.4*:

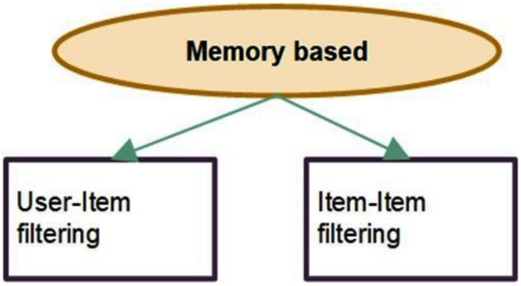


***Figure 6.4**: Memory-based collaborative filtering approaches*

A user-item filtering algorithm takes a specific user, finds users who are similar to that user based on rating similarity, and recommends items that those similar users liked. Item-item filtering, on the other hand, will take an item and find users who liked it, as well as other items that those users or similar users liked. It takes items and generates recommendations for other items.

User-Item Collaborative Filtering: *"Users who are similar to you also liked …"*

Item-Item Collaborative Filtering: *"Users who liked this item also liked …"*

The main difference between the memory-based approach and the model-based techniques is that we are not learning any parameters using gradient descent (or any other optimization algorithm). The closest user or items are determined solely through the use of Cosine similarity or Pearson correlation coefficients, which are based solely on arithmetic operations.

- **Model-based approach**: In this method, CF models are created using machine learning algorithms to predict the ratings of unrated items by users. This approach's algorithms are further classified into three sub-types: Clustering-based, Matrix factorization based and Deep Learning as shown in *Figure 6.5*:

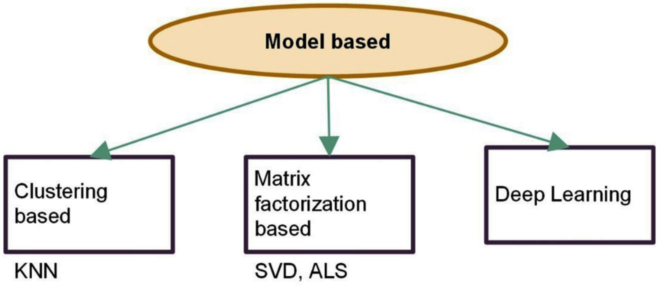


***Figure 6.5**: Model- based collaborative filtering approaches*

Content-based filtering

Systems that use a content-based recommendation approach investigate a set of documents and/or explanations of items previously rated by a user and create a model or profile of that user's interests based on the characteristics of the objects rated by that user. The profile is a structured representation of the user's interests that is used to recommend new and interesting items.

The recommendation process entails matching the attributes of a user profile to the attributes of a content object. As a result, the user's level of interest in that object is represented by a relevant judgment. When a profile accurately reflects user preferences, it greatly improves the efficiency of an information access process. It could, for example, be used to filter search results by determining whether a user is interested in a specific Web page and, if not, preventing it from being displayed.

The architecture of content-based filtering

Content-based Filtering (CF) systems require appropriate techniques for representing items and creating the user profile, as well as some strategies for comparing the user profile to the item representation. *Figure 6.6* depicts the architecture of a content-based recommended system. The recommendation process is divided into three steps, each of which is handled by a different component:

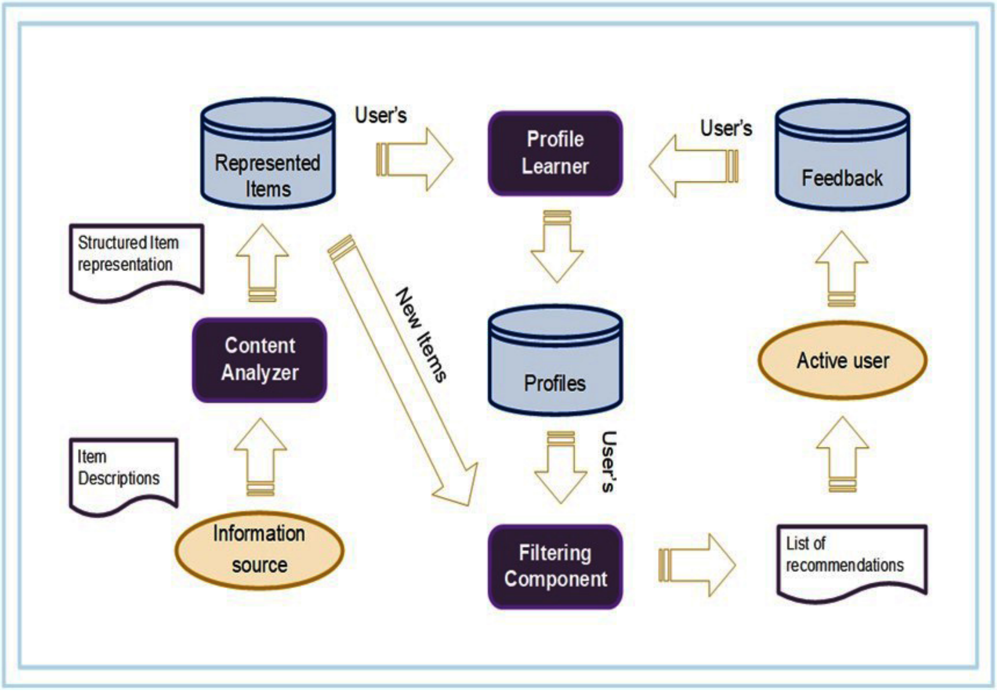


***Figure 6.6**: Architecture of Content-based filtering*

1. **Content Analyzer**: The component's primary responsibility is to represent the content of items (e.g., documents, web pages, news, product descriptions, etc.) derived from information sources in a format suitable for subsequent processing steps. Feature extraction techniques are used to analyze data items. This representation serves as the input for the profile learner and filtering component.
2. **Profile Learner**: To construct the user profile, this module collects data representative of the user's preferences and attempts to generalize this data. Typically, the generalization strategy is implemented using machine learning techniques, which can infer a model of user interests based on items liked or disliked in the past. For example, a web page recommender's profile learner can use a relevance feedback method in which the learning technique combines vectors of positive and negative examples into a prototype vector representing the user profile. Web pages that have received positive or negative feedback from users serve as training examples.
3. **Filtering Component**: This module uses the user profile to suggest relevant items by comparing the profile representation to the representation of the items to be recommended. The result is a binary or continuous relevance judgment, with the latter producing a ranked list of potentially interesting items.

The content analyzer, which typically borrows techniques from information retrieval systems, performs the first step in the recommendation process. Item descriptions from the information source are processed by the content analyzer, which extracts features from unstructured text to create a structured item representation, which is then saved in the repository represented items.

To build and update the profile of the active user (the user for whom recommendations must be provided), her reactions to items are collected in some way and recorded in the repository feedback. These reactions, known as annotations or feedback, along with the related item descriptions, are used to train a model that can predict the actual relevance of newly presented items. Users can also define their areas of interest explicitly as an initial profile without providing feedback.

Typically, there are two types of relevant feedback: positive information (inferring features liked by the user) and negative information (i.e., inferring features the user is not interested in). For recording user feedback, two different techniques can be used. When a system requires the user to explicitly evaluate items, this technique is known as "*explicit feedback*." The other technique, known as "*implicit feedback*," does not require any active user involvement in the sense that feedback is derived from monitoring and analyzing the user's activities.

Advantages of content-based filtering

When compared to the collaborative paradigm, adopting the content-based filtering paradigm has several advantages.

- **User independence**: Content-based recommenders rely solely on ratings provided by the active user.
- **Transparency**: Interpretations on how the recommended system works can be presented by explicitly listing the content features or descriptions that resulted in an item appearing in the list of recommendations. These characteristics are indicators to consider when deciding whether to trust a recommendation.
- **New item**: Content-based recommended systems can recommend items that have not yet been rated by any user. As a result, they do not suffer from the first-rate problem, which affects collaborative recommenders that make recommendations solely based on user preferences.

Drawbacks of content-based filtering

Following are the disadvantages:

- **Limited Content Analysis**: Content-based techniques have a natural limit in terms of the number and type of features that can be associated with the objects they recommend, whether automatically or manually. Domain knowledge is frequently required, for example, for movie recommendations, the system must know the actors and directors, and domain ontologies are sometimes required. No content-based recommendation system can provide appropriate suggestions if the analyzed content lacks sufficient information to distinguish between items the user likes and items the user does not like.
- **Over-specialization**: There is no inherent method for finding something unexpected in content-based recommended systems. The system suggests items whose scores are high when compared to the user profile, so the user will be recommended items similar to those already rated. This disadvantage is also known as the **serendipity problem** because it highlights the tendency of content-based systems to produce recommendations with a limited degree of novelty.
- **New User**: A sufficient number of ratings must be collected before a content-based recommended system can truly understand user preferences and make accurate recommendations. As a result, when there are few ratings available, such as for a new user, the system will be unable to provide reliable recommendations.

Artificial Neural Network (ANN)

Artificial Neural Networks are computational models based on the human brain. Many recent advances in the field of Artificial Intelligence have been made, including voice recognition, image recognition, and robotics using Artificial Neural Networks. **Artificial Neural Networks** (**ANNs**) are biologically inspired computer simulations that perform specific tasks such as clustering, classification, and pattern recognition.

These biological computing methods are regarded as the next major advancement in the technology world. Simply, Artificial Neural Networks are software implementations of human brain neural structures. A neural network (ANN) is a computational system that is influenced by the structure, processing capability, and learning ability of the human brain.

Let us look at the structure of our brain instead of delving into its complex biology. The human brain is made up of billions of neurons that function as organic switches. All of these neurons are linked together to form a massive and complex structure known as a **Neural Network**. A single neuron's output is dependent on inputs from thousands of interconnected neurons.

This brain behavior is critical to Artificial Neural Networks, which simply try to replicate the brain's action. This can be explained as follows:

- Supervised ANN
- Unsupervised ANN

Supervised ANN

A matched input and output sample of data is provided to the network for training in supervised ANN. The goal of this approach is to obtain the desired output for a given input.

The spam filters in our e-mails are an excellent example of a supervised ANN. At the level of training, the Artificial Neural Network engine would be fed a set of words from the body of the e-mail. The output classifies the e-mail as spam or not spam.

Unsupervised ANN

Unsupervised ANNs are more complex than supervised ANNs because they attempt to teach the ANN to understand the data structure provided as input on its own.

The biological neuron's connections are represented as weights. A positive weight indicates an excitatory connection, while a negative weight indicates an inhibitory connection. All inputs are weighted and added together. This activity is known as a **linear combination**. Finally, an activation function regulates the output amplitude.

For example, an acceptable output range is usually between 0 and 1, but it could also be between 1 and 1.

Applications

Artificial Neural Networks can be applied in a variety of fields as follows:

- Nonlinear system identification and control (vehicle control, process control)
- Game-playing and decision making (backgammon, chess, racing)
- Pattern recognition (radar systems, face identification, object recognition)
- Sequence recognition (gesture, speech, handwritten text recognition)
- Medical diagnosis
- Financial applications
- Data mining (or knowledge discovery)
- Function approximation, or regression analysis
- Time series prediction and modeling
- Classification and clustering
- Novelty detection
- Data processing and filtering
- Blind signal separation and compression

Comparison between Artificial Neural Networks (ANN) and Biological Neural Networks (BNN)

S.No.	ANN	BNN
1	Artificial Neural Network	Biological Neural Network
2	Faster in processing information.	Slower in processing information.
3	Information storage is replaceable means new data can be added by deleting an old one.	Highly complex.
4	Processes operate in sequential mode.	The process can operate in massive parallel operations.

5	If any information gets corrupted in the memory it cannot be retrieved.	Information is distributed into the network throughout into sub-nodes, even if it gets corrupted it can be retrieved.
6	The activities are continuously monitored by a control unit.	No specific control mechanism.

***Table 6.1:** Comparison between Artificial Neural Networks (ANN) and Biological Neural Networks (BNN)*

Characteristics of the ANN

Any Artificial Neural Network, regardless of implementation style or logic, has a few basic characteristics. These are listed as follows:

- An Artificial Neural Network is made up of a large number of "*neurons*" or processing elements.
- All of these processing elements are linked by a large number of weighted connections.
- The input signals are routed through connections and connecting weights to the processing elements.
- The links between the elements form a distributed representation of data.
- It can learn, recall, and generalize from given data by assigning and adjusting weights appropriately.
- Neurons' collective actions describe their computational power, and no single neuron carries specific information.

Perceptron

A perceptron is a supervised learning algorithm for binary classifiers. Binary classifiers determine whether an input, which is typically represented by a series of vectors, belongs to a particular class. In a nutshell, a perceptron is a one-layer neural network. They are made up of four major components: input values, weights and bias, net sum, and an activation function.

A single layer neural network is referred to as a perceptron, and a multi-layer perceptron is referred to as a neural network. A perceptron is a type of linear classifier (binary). It is also used in supervised learning. It aids in the classification of the input data. Perceptron is commonly used to divide data into two parts. As a result, it is also referred to as a Linear Binary Classifier.

This algorithm enables neurons to learn and process training set elements one at a time.

There are two types of perceptrons: Single layer and Multilayer.

- Perceptrons with a single layer can only learn linearly separable patterns.
- The processing power of multilayer perceptrons or feedforward neural networks with two or more layers is greater.

To draw a linear decision boundary, the perceptron algorithm learns the weights for the input signals.

Activation Function

The activation function determines whether or not a neuron should be activated by calculating a weighted sum and then adding bias to it. The activation function's purpose is to introduce non-linearity into a neuron's output.

Why do we need an Activation Function?

In a nutshell, activation functions are used to map input between required values such as (0, 1) or (-1, 1).

We understand that neurons in a neural network work by their weight, bias, and activation function. In a neural network, we would update the weights and biases of the neurons based on the output error. This is referred to as backpropagation. Because the gradients are supplied along with the error to update the weights and biases, activation functions enable back-propagation.

Why do we need Non-linear activation functions?

Without an activation function, a neural network is essentially a linear regression model. The activation function applies a non-linear transformation to the input, allowing it to learn and perform more complex tasks.

How does a Perceptron work?

The procedure begins by multiplying all of the input values by their weights. The weighted sum is then calculated by multiplying all of the multiplied values together. The weighted sum is then applied to the activation function, yielding the output of the perceptron. The activation function is crucial in ensuring that the output is mapped between required values such as (0,1) or (-1,1). It is critical to understand that the weight of input indicates the strength of a node. Similarly, the bias value of input allows you to shift the activation function curve up or down.

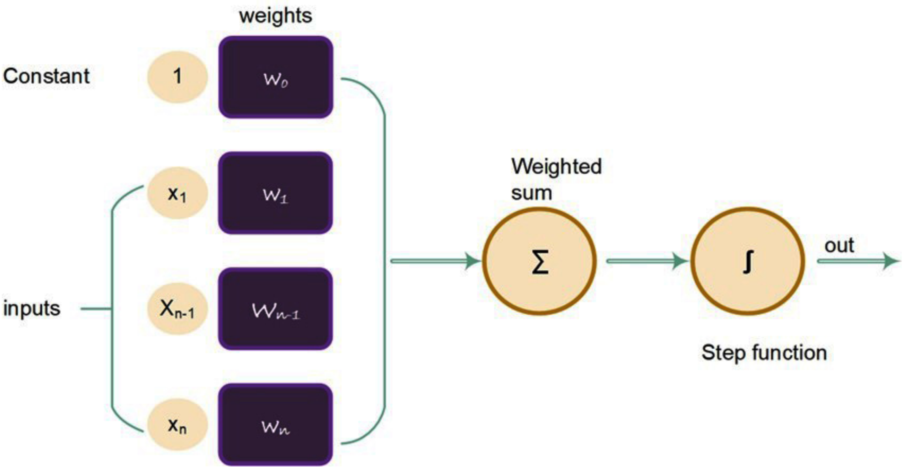


Figure 6.7: Perceptron working

Perceptrons are crucial in binary classification. This means that the perceptron is used to divide data into two parts, thus binary. The perceptron operates on the following simple steps:

Step 1: All the X inputs are multiplied by their weights W.

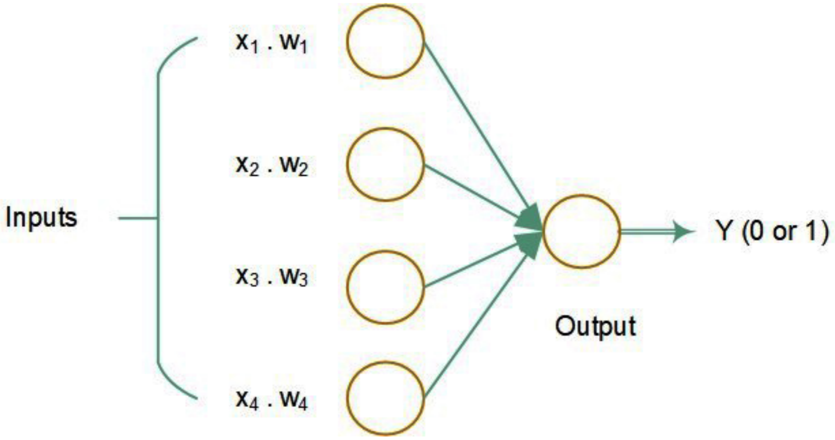


Figure 6.8: Multiplying inputs with weights for 4 inputs

Step 2: Weighted Sum is the sum of all the multiplied values.

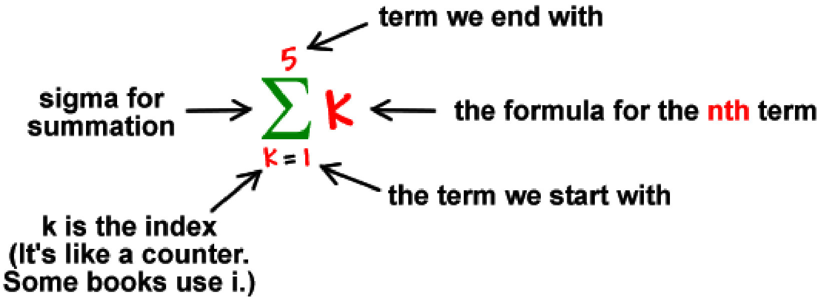

term we end with
5
sigma for summation
$\sum$ K
the formula for the nth term
K = 1
the term we start with
k is the index (It's like a counter. Some books use i.)

Figure 6.9: Calculation of the Weighted Sum

Step 3: Use that weighted sum to find the appropriate activation function.

Multilayer Network

Hidden layers, whose neurons are not directly connected to the output, are used in multilayer networks to solve the classification problem for non-linear sets. The additional hidden layers can be interpreted geometrically as additional hyper-planes that increase the network's separation capacity. *Figure 6.10* depicts common multilayer Neural network architectures.

This new architecture raises an important question: how to train hidden units when the desired output is unknown. This problem can be solved using the Backpropagation algorithm.

I) **Input Layer**: This layer accepts features as input. It brings information from the outside world into the network; no computation is done at this layer; nodes simply pass the information (features) to the hidden layer.

II) **Hidden Layer**: This layer's nodes are not visible to the outside world; they are part of the abstraction provided by any neural network. The hidden layer computes all of the features entered through the input layer and sends the results to the output layer.

III) Output Layer: This layer communicates the network's learned information to the outside world.

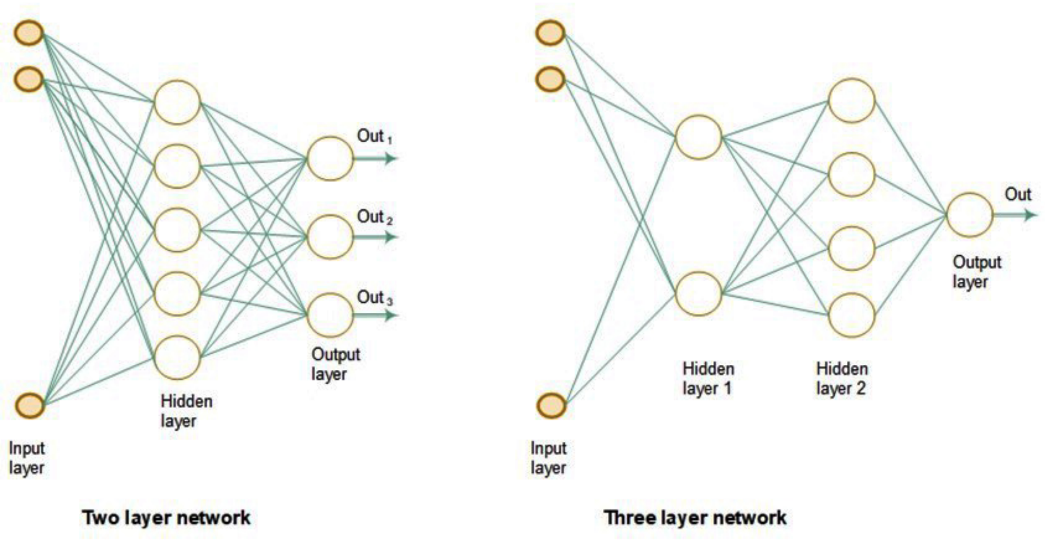


***Figure 6.10**: Multilayer Neural Network Architectures*

Backpropagation algorithm

The backpropagation algorithm is used to effectively train a neural network using a method known as **chain rule**. Backpropagation, in a nutshell, performs a backward pass through a network after each forward pass while adjusting the model's parameters (weights and biases). Backpropagation is an abbreviation for "*backward error propagation*." It is a common technique for training artificial neural networks. This method is useful for calculating the gradient of a loss function for the network's weights.

The essence of neural net training is backpropagation. It is a technique for fine-tuning the weights of a neural network based on the error rate obtained in the previous epoch (i.e., iteration). By fine-tuning the weights, you can reduce error rates and make the model more reliable by increasing its generalization.

Algorithm steps: Consider the following *Figure 6.11* for the backpropagation algorithm working as shown follows:

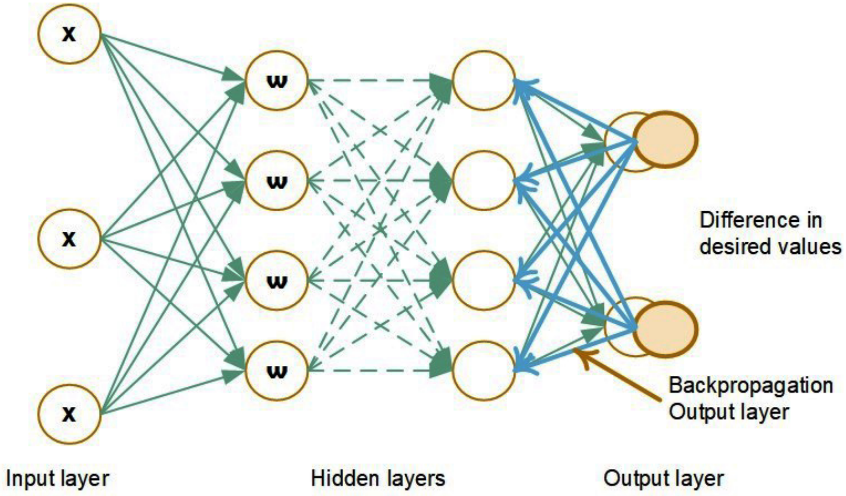


***Figure 6.11**: Backpropagation algorithm working*

Step 1: Inputs X, arrive through the preconnected path.

Step 2: Input is modelled using real weights W. The weights are usually randomly selected.

Step 3: Calculate the output for every neuron from the input layer to the hidden layers, to the output layer.

Step 4: Calculate the error in the outputs

$$ErrorB = Actual\ Output - Desired\ Output$$

Step5: Travel back from the output layer to the hidden layer to adjust the weights such that the error is decreased.

Step 6: Repeat the process (from steps 1 to 5) until the desired output is achieved.

Types of Backpropagation Networks

There are two types of Backpropagation Networks they are as follows:

- **Static backpropagation:** It is a type of backpropagation network that generates a mapping of a static input to a static output. It can be used to solve static classification problems such as optical character recognition.
- **Recurrent Backpropagation:** Recurrent backpropagation is used until a fixed value is reached. The error is then computed and propagated backwards.

The main difference between these two methods is that in static backpropagation, the mapping is fast, whereas, in recurrent backpropagation, the mapping is non-static.

Advantages

The following are the advantages of the Backpropagation Networks:

- It is simple, fast, and easy to program.
- Apart from the numbers of input, there are no parameters to tune.
- It is a standard method that generally works well.
- It is a versatile method because it does not necessitate prior knowledge of the network.

Disadvantages

The following are the disadvantages of the Backpropagation Networks:

- The actual performance is determined by the data input.
- Backpropagation is extremely sensitive to noisy data.
- Backpropagation must be done using a matrix-based approach rather than a mini-batch approach.

Introduction to Deep Learning

Deep learning algorithms employ a technique known as a neural network to discover relationships between a set of inputs and outputs.

Machine learning has a subdomain called **deep learning**. Deep learning algorithms can self-learn hidden patterns within data to make predictions using accelerated computational power and large data sets. In essence, deep learning is a subset of machine learning (see *Figure 6.11)*) that is trained on large amounts of data and involves many computational units working together to make predictions.

The term "*deep learning*" refers to the creation of deep neural networks. This refers to a neural network with many layers. By adding more weights and biases, the neural network improves its capabilities to predict more complex functions.

Deep learning is a subset of machine learning, which is a subset of artificial intelligence. Individual definitions are as follows:

- The broad mandate of artificial intelligence is to create machines that can think intelligently.

- Machine learning employs algorithms to extract insights from data.
- Deep learning employs a specific algorithm known as a Neural Network.

How do deep learning algorithms "learn"?

The following *Figure 6.12* depicts the concept of the deep learning algorithm:

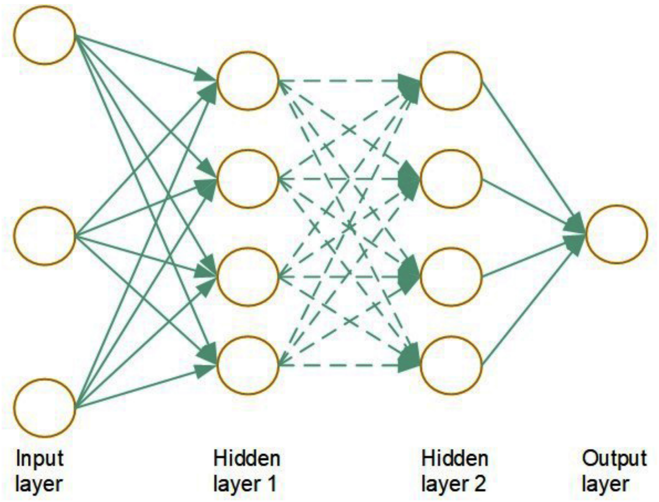


***Figure 6.12**: Deep Learning Algorithms concept*

A neural network is made up of input, hidden, and output layers, all of which are made up of "*nodes*." Input layers receive a numerical representation of data (for example, images with pixel specs), output layers produce predictions, and hidden layers are correlated with the majority of the computation.

Difference between Machine Learning and Deep Learning

To grasp the difference between machine learning and deep learning, we must first understand the relationship between deep learning and machine learning. This relationship is best represented by an example shown in *Figure 6.13*. In machine learning, feature extraction and classification are applied separately to get the output whereas in deep learning feature extraction and classification is applied together to get the output. By using deep learning, the complexity reduces to achieve the results faster.

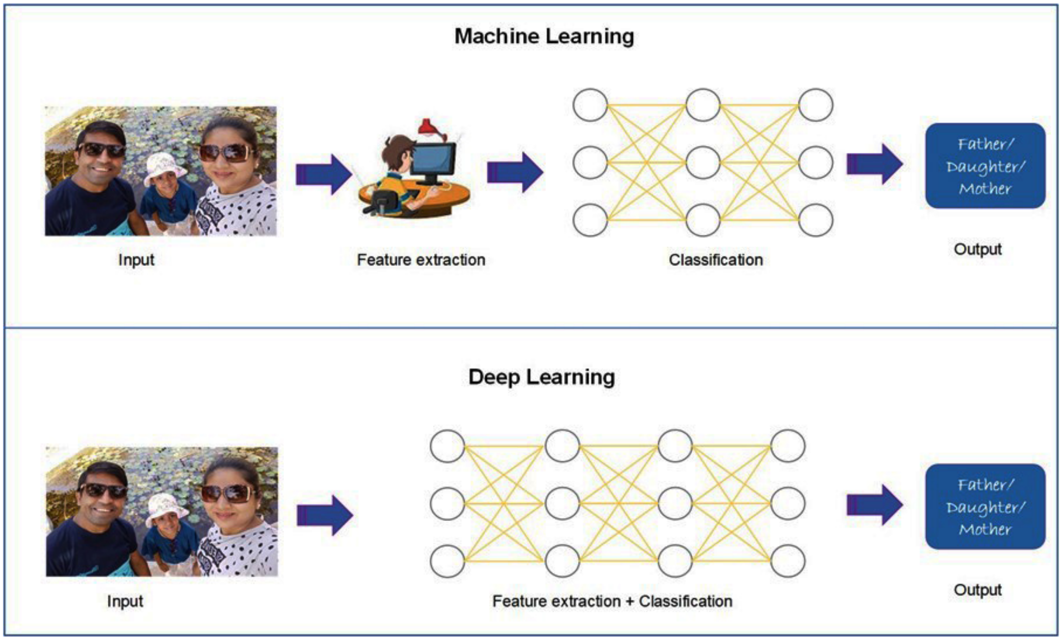


***Figure 6.13**: Machine Learning vs Deep Learning*

The following *table 6.2* shows the difference between Machine Learning and Deep Learning.

Machine Learning	Deep Learning
Acronym is ML	Acronym is DL
Works on a small amount of dataset	Works on a large amount of dataset
Dependent on low-end machine	Dependent on high-end machine
Divides the whole task into sub-tasks and solves them	Solves the problem end to end
Less time to train	Longer time to train
More testing time is required	Less testing time is required

***Table 6.2**: Difference between Machine Learning and Deep Learning*

Importance of Deep Learning

Deep learning is significant because we have achieved meaningful and useful accuracy on tasks that matter. For decades, machine learning has been used for image and text classification, but it has struggled to cross the threshold of accuracy that algorithms require to work in business settings. Deep learning is finally allowing us to cross that line in places where we previously could not as shown in *Figure 6.14*:

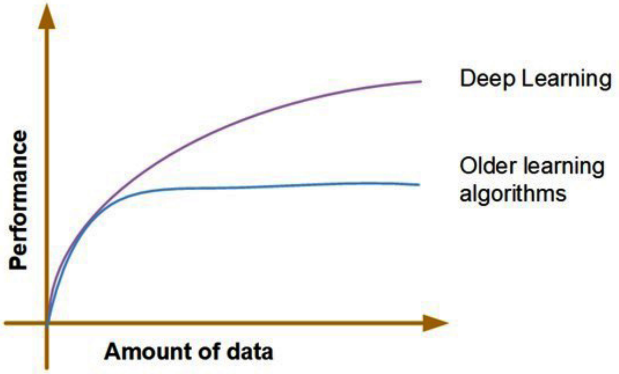


Figure 6.14*: Comparison of deep learning and older learning algorithms*

The following are some of the real-time examples:

Computer vision is an excellent example of a task that deep learning has rendered feasible for business applications. Deep learning is not only better than any other traditional algorithm for classifying and labelling images; it is also beginning to outperform actual humans.

Using deep learning, *Facebook* has already had remarkable success identifying facial expressions in photographs. It's not just a minor improvement; it's a huge success.

Deep learning is already being used by *Google* to manage the energy at the company's data centers. Deep learning has also had an impact on speech recognition.

Deep learning is significant because it brings previously inaccessible workloads into the purview of machine learning. We are just beginning to understand how developers and business leaders can use machine learning to drive business outcomes and having more available tasks at your fingertips as a result of deep learning will transform the economy for decades to come.

Advantages are as follows :

- Best in-class performance
- Reduces the need for feature engineering
- Cuts out unnecessary expenses
- Identifies minor defects

Disadvantages are as follows:

- Massive quantity of data is required

- Training is computationally expensive
- There is no solid theoretical foundation

Applications of Deep Learning

Deep learning has a wide range of applications in almost every industry, including health care, finance, and image recognition. Some of the most popular application areas are as follow:

- **Health care**: Image recognition, cancer detection from MRI imaging and x-rays, drug discovery, clinical trial matching, and genomics.
- **Autonomous vehicles**: Self-driving cars, recognizing a stop sign to seeing a pedestrian on the road.
- **e-commerce**: Product recommendations.
- **Automatic Text Generation**: Word-by-word or character-by-character.
- **Automatic Machine Translation**: Certain words, sentences or phrases in one language is transformed into another language.
- **Personal assistant**: Devices like Alexa or Google Assistant.
- **Predicting Earthquakes**: Teaches a computer to perform viscoelastic computations, which are used in earthquake prediction.

These are only a few of the numerous applications to which deep learning can be applied. Stock market forecasting and weather forecasting are two other popular fields where deep learning has proven useful.

Summary

- **Recommended Systems** (**RS**) are software tools and techniques that provide recommendations for items that may be useful to a user.
- There are three types of recommendation approaches: Content-based recommended systems, Collaborative filtering, and Hybrid recommended systems.
- Collaborative filtering does not require anything other than the users' historical preferences on a set of items.
- The Collaborative filtering techniques are broadly divided into two types: Memory-based and Model-based approach.
- Systems that use a content-based recommendation approach investigate a set of documents and/or explanations of items previously rated by a user and

create a model or profile of that user's interests based on the characteristics of the objects rated by that user.

- **Artificial Neural Networks** (**ANNs**) are biologically inspired computer simulations that perform specific tasks such as clustering, classification, and pattern recognition.
- A perceptron is a supervised learning algorithm for binary classifiers.
- There are two types of perceptrons: Single layer and Multilayer.
- The backpropagation algorithm is used to effectively train a neural network using a method known as chain rule.
- Backpropagation is particularly useful for deep neural networks working on error-prone projects like image or speech recognition.
- Deep learning refers to Deep Neural Networks.
- There are two types of backpropagation networks are: Static Backpropagation and Recurrent Backpropagation.
- Deep learning algorithms employ a technique known as a neural network to discover relationships between a set of inputs and outputs.

Exercise (MCQs)

1. **What is the objective of the backpropagation algorithm?**

   a) to develop a learning algorithm for multilayer feedforward neural network

   b) to develop a learning algorithm for a single-layer feedforward neural network

   c) to develop a learning algorithm for a multilayer feedforward neural network, so that network can be trained to capture the mapping implicitly

   d) none of the above

2. **What is true regarding the backpropagation rule?**

   a) it is also called generalized delta rule

   b) error in output is propagated backwards only to determine weight updates

   c) there is no feedback of signal at any stage

   d) all of the mentioned

3. **Why do we need biological neural networks?**

    a) to solve tasks like machine vision & natural language processing

    b) to apply heuristic search methods to find solutions to a problem

    c) to make smart human interactive & user-friendly system

    d) all of the mentioned

4. **What is plasticity in neural networks?**

    a) input pattern keeps on changing

    b) input pattern has become static

    c) output pattern keeps on changing

    d) output is static

5. **The fundamental unit of a network is**

    a) brain

    b) nucleus

    c) neuron

    d) axon

6. **Which of the following would have a constant input in each epoch of training a deep learning model?**

    a) Weight between input and hidden layer

    b) Weight between hidden and output layer

    c) Biases of all hidden layer neurons

    d) Activation function of the output layer

    e) None of the above

7. **Backpropagations cannot be applied when using pooling layers**

    a) TRUE

    b) FALSE

8. **For a binary classification problem, which of the following architecture would you choose?**

    a) 1

    b) 2

    c) Any one of these

    d) None of these

9. **Suppose there is an issue while training a neural network. The training loss/validation loss remains constant. What could be the possible reason?**
   a) Architecture is not defined correctly
   b) Data given to the model is noisy
   c) Both of these

10. **Which of the following is a subset of machine learning?**
    a) SciPy
    b) *P* NumPy
    c) Deep Learning
    d) All of the above

Answers

1. **c**
2. **d**
3. **d**
4. **a**
5. **c**
6. **a**
7. **b**
8. **c**
9. **c**
10. **c**

Short Answers Questions

1. Define recommended systems.
2. What do you mean by collaborative filtering?
3. Define Artificial Neural Networks (ANNs).
4. Describe the meaning of perceptron.
5. Explain the main characteristics of Artificial Neural Networks?
6. What is an activation function and why use them?
7. Explain the importance of the backpropagation algorithm.
8. Difference between machine learning and deep learning.
9. Write a short note on deep learning.

Long Answers Questions

1. Explain the types of the recommended systems with examples.
2. Explain the collaborative filtering techniques with diagrams.
3. Differentiate between the ANN and BNN.
4. Explain the single-layer and multilayer perceptrons.
5. What is the backpropagation algorithm? Explain with an example.
6. What is the role of weights and bias in a neural network?
7. Explain the importance of deep learning with an application as example.

Bibliography

1. Hormozi, H., Hormozi, E. & Nohooji, H. R. (2012). The Classification of the Applicable Machine Learning Methods in Robot Manipulators. International Journal of Machine Learning and Computing (IJMLC), Vol. 2, No. 5, 2012 doi: 10.7763/IJMLC.2012.V2.189pp. 560 – 563. Available at IJMLC website: **http://www.ijmlc.org/papers/189-C00244-001.pdf**

2. Kotsiantis, S. B. (2007). Supervised Machine Learning: A Review of Classification Techniques. Informatica 31 (2007). Pp. 249 – 268. Retrieved from IJS website: **http://wen.ijs.si/ojs2.4.3/index.php/informatica/article/download/148/140**.

3. Logistic Regression pp. 223 – 237. Available at: **https://www.stat.cmu.edu/~cshalizi/uADA/12/lectures/ch12 .pdf**

4. Rosenblatt, F. (1962), Principles of Neurodynamics. Spartan, New York.

5. Setiono R. and Loew, W. K. (2000), FERNN: An algorithm for fast extraction of rules from neural networks, Applied Intelligence. International Journal of Computer Trends and Technology (IJCTT) – Volume 48 Number 3 June 2017 ISSN: 2231-2803 **http://www.ijcttjournal.org Page 138**

6. Shai Shalev-Shwartz and Shai Ben-David (2014). Understanding Machine Learning From Theory to Algorithms.

7. T. Hastie, R. Tibshirani, J. H. Friedman (2001) — The elements of statistical learning,‖ Data mining, inference, and prediction, 2001, New York: Springer Verlag.

8. Taiwo, O. A. (2010). Types of Machine Learning Algorithms, New Advances in Machine Learning, Yagang Zhang (Ed.), ISBN: 978-953-307-034-6, InTech, University of Portsmouth United Kingdom. Pp 3 – 31. Available at InTech open website: **http://www.intechopen.com/books/new-advances-inmachine-learning/types-of-machine-learning-algorithms**

9. Tapas Kanungo, D. M. (2002). A local search approximation algorithm for k-means clustering. Proceedings of the eighteenth annual symposium on Computational geometry. Barcelona, Spain: ACM Press.

10. Hiran, K., Khazanchi, D., Vyas, A. & Padmanaban, S. (2021). Machine Learning for Sustainable Development. Berlin, Boston: De Gruyter. **https://doi.org/10.1515/9783110702514**

11. Mahrishi, M., Hiran, K. K., Meena, G., & Sharma, P. (Eds.). (2020). Machine Learning and Deep Learning in Real-time Applications. IGI global.

12. Barua, T., Doshi, R. & Hiran, K. (2020). Mobile Applications Development. Berlin, Boston: De Gruyter. **https://doi.org/10.1515/9783110689488**

13. Witten, I. H. & Frank, E. (2005). Data Mining: Practical machine learning tools and techniques (2nd ed.), ISBN: 012-088407-0, Morgan Kaufmann Publishers, San Francisco, CA, U.S.A. © 2005 Elsevier Inc.Retrieved from website: ftp://93.63.40.27/pub/manuela.sbarra/Data Mining Practical Machine Learning Tools and Techniques - WEKA.pdf

14. S. B. Kotsiantis, P. E. Pintelas and I. D. Zaharakis, "Machine learning: a review of classification and combining techniques", Artificial Intelligence Rev, vol. 26, pp. 159-190, 2006.

15. S. B. Kotsiantis, "Supervised Machine Learning: A Review of Classification", Informatica (slovenia), vol. 31, pp. 249-268, 2007.

16. K. Wisaeng, "A Comparison of Different Classification Techniques for Bank Direct Marketing", International Journal of Soft Computing and Engineering (IJSCE), vol. 3, no. 4, pp. 116-119, 2013.

17. M. J. Islam, Q. M. J. Wu, M. Ahmadi and M. A. Sid-Ahmed, "Investigating the Performance of Naïve-Bayes Classifiers and K-Nearest Neighbor Classifiers", Journal of Convergence Information Technology, vol. 5, no. 2, 2010.

18. C.S. Trilok and J. Manoj, "WEKA Approach for Comparative Study of Classification Algorithm", International Journal of Advanced Research in Computer and Communication Engineering, pp. 1925-1931, 2013.

19. R. Caruana and A. Niculescu-Mizil, "An empirical comparison of supervised learning algorithms", 23rd international conference on Machine learning, 2006.

20. R. Entezari-Maleki, A. Rezaei and B. Minaei-Bidgoli, "Comparison of Classification Methods Based on the Type of Attributes and Sample Size", Journal of Convergence Information Technology, vol. 4, no. 3, pp. 94-102, 2009.

21. Lakhwani, K., Bhargava, S., Somwanshi, D., Doshi, R., & Hiran, K. K. (2020, December). An Enhanced Approach to Infer Potential Host of Coronavirus by Analyzing Its Spike Genes Using Multilayer Artificial Neural Network. In 2020 5th IEEE International Conference on Recent Advances and Innovations in Engineering (ICRAIE) (pp. 1-5). IEEE.

22. Vyas, A., Dhiman, H. & Hiran, K. (2021). Modelling of symmetrical quadrature optical ring resonator with four different topologies and performance analysis using machine learning approach. Journal of Optical Communications, (), 000010151520200270. **https://doi.org/10.1515/joc-2020-0270**

23. H. Liu and L. Yu, "Toward integrating feature selection algorithms for classification and clustering", IEEE Transactions on knowledge and Data Engineering, vol. 17, no. 4, pp. 491-502, 2005.

24. L. Rokach and O. Maimon, "Top-down induction of decision trees classifiers-a survey", IEEE Transactions on Systems Man and Cybernetics Part C: Applications and Reviews, vol. 35, no. 4, pp. 476-487, 2005.

25. C. Lorena et al., "Comparing machine learning classifiers in potential distribution modelling", Expert Systems with Applications, vol. 38, pp. 5268-5275, 2011.

26. M. Robnik-Sikonja, "Improving Random Forests", Machine Learning ECML, 2004.

27. M. Aly, "Survey on multiclass classification methods", Neural Network, pp. 1-9, 2005.

28. G. Mountrakis, J. Im and C. Ogole, "Support vector machines in remote sensing: A review", ISPRS Journal of Photogrammetry and Remote Sensing, vol. 66, no. 3, pp. 247-259, 2011.

29. Ian H. Witten, Eibe Frank and Mark A. Hall, Data Mining — Practical Machine Learning Tools and Techniques, Elsevier, 2014.

30. M. Javier ,M. Moguerza, —Support Vector Machines with Applications,‖ Statistical Science , vol. 21, no. 3, pp. 322-336, 2006.

31. J. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," Data Mining and Knowledge Discovery, vol. 2, 1998.

32. Cover, T. , Hart, —Nearest Neighbor Pattern Classification,‖ IEEE Transactions on Information Theory, vol. 13, no. 1, pp. 21-27, 1967.

33. E. Rumelhart, G. E. Hinton and R. I. Williams, —Learning internal representation by error propagation,‖ Parrallel Distrubuted Processing, 1986.

34. Duda R, Hart P, "Pattern Classification and Scene Analysis," John Wiley and Sons, New York, 1973.

35. F. Rosenblatt, —The perceptron: A probabilisticc model for informtaion storage and organization in the brain,‖ Psychological Review, vol. 65, pp. 386-498, 1958.

36. Gao, Jiawei Hen and Jing, —Classification and regression trees,‖ Wadsworth, Belmont, 1984.

37. J. Han and M. Kamber, Data Mining Concepts and Techniques, Elevier, 2011.

38. J. R. Quinlan, —Discovering rules by induction from large collections of examples,‖ Expert Systems in the Microelectronic age, pp. 168-201, 1979.

39. J. R. Quinlen, —Introduction of Decision Trees,‖ Machine Learning, vol. 1, pp. 81-106, 1986.

40. J. R.Quinlan, —C4.5: Programs for machien learning,‖ Morgan Kaufmann, San Fransisco, 1993.

41. Jensen, —An Introduction to Bayesian Networks,‖ Springer, 1996.

42. K. P. Soman, Insight into Data Miining Theory and Practice, New Delhi: PHI, 2006.

43. Klaus_Robert Muller, Sebastian Mika, Gunnar Ratsch, Koji Tsuda, Bernhard Scholkopf, —An Introduction to Kernel Based Learning Algorithms,‖ CRC Press, 2002.

44. Robert Burbodge, Bernard Buxton, —An introduction to Support Vector Machines for Data Mining,‖ Computer Science Dept., UCL, UK.

45. S. B. Kotsiantis, —Supervised Machine Learning: A Riview of Classification Techniques,‖ Informatica, vol. 31, pp. 249-268, 2007.

46. Thair N. Phyu, —Survey of Classification echniques in Data Mining,‖ in International Multiconference of Engineers and Computer Scientists, Hong Kong, 2009.

47. Vapnik, Corinna Cortes and Vladimir, —Support Vector Network,‖ Machine Learning, vol. 20, pp. 273-297, 1995.

48. XindongWu, Vipin Kumar, J. Ross Quinlan, Joydeep Ghosh, Qiang Yang, Hiroshi Motoda, Geoffrey J. McLachlan, Angus Ng, Bing Liu, Philip S. Yu, Zhi-Hua Zhou, Michael Steinbach, David J. Hand, Dan Steinberg, —Top 10 algorithms in data mining,‖ Knowledge Information system, vol. 14, pp. 1-37, 2008.

49. Marco Stang, Daniel Grimm, Moritz Gaiser, Eric Sax. (2020) Evaluation of Deep Reinforcement Learning Algorithms for Autonomous Driving. 2020 IEEE Intelligent Vehicles Symposium (IV), 1576-1582

50. Hiran, K. K., Doshi, R., Fagbola, T., & Mahrishi, M. (2019). Cloud Computing: Master the Concepts, Architecture and Applications with Real-world examples and Case studies. Bpb Publications.

51. Hiran, K. K. (2021). Investigating Factors Influencing the Adoption of IT Cloud Computing Platforms in Higher Education: Case of Sub-Saharan Africa With IT Professionals. International Journal of Human Capital and Information Technology Professionals (IJHCITP), 12(3), 21-36.

52. Hiran, K. K., & Henten, A. (2020). An integrated TOE-DoI framework for cloud computing adoption in higher education: The case of Sub-Saharan Africa, Ethiopia. In Soft Computing: Theories and applications (pp. 1281-1290). Springer, Singapore.

53. Woojin Seol, Youngjun Jeon, Kyungsoo Kim, Soohyun Kim. (2020) Legged balance on moving table by reinforcement learning. 2020 20th International Conference on Control, Automation and Systems (ICCAS), 900-905.

54. Doshi, R. (2018). Adoption of the ICT application Moodle Cloud to enhance teaching-learning in large classes: Case of Sub-Sahara Africa.

55. Sharma, G., Mahrishi, M., Hiran, K. K., & Doshi, R. (2020). Reverse Engineering for potential Malware detection: Android APK Smali to Java. Journal of Information Assurance & Security, 15(1).

56. Yi Yu, Lina Mroueh, Shuo Li, Michel Terre. (2020) Multi-Agent Q-Learning Algorithm for Dynamic Power and Rate Allocation in LoRa Networks. 2020 IEEE 31st Annual International Symposium on Personal, Indoor and Mobile Radio Communications, 1-5.

57. Amandeep Singh Bhatia, Mandeep Kaur Saggi, Amit Sundas, Jatinder Ashta. 2020. Reinforcement Learning. Machine Learning and Big Data, 281-303.

58. Pablo Hernandez-Leal, Bilal Kartal, Matthew E. Taylor. (2019) A survey and critique of multiagent deep reinforcement learning. Autonomous Agents and Multi-Agent Systems 33:6, 750-797.

59. Riccardo Porotti, Dario Tamascelli, Marcello Restelli, Enrico Prati. (2019) Reinforcement Learning Based Control of Coherent Transport by Adiabatic Passage of Spin Qubits. Journal of Physics: Conference Series 1275, 012019.

60. Mahmoud El Chamie, Dylan Janak, Behçet Açıkmeşe. (2019) Markov decision processes with sequential sensor measurements. Automatica 103, 450-460.

61. R. Rocchetta, L. Bellani, M. Compare, E. Zio, E. Patelli. (2019) A reinforcement learning framework for optimal operation and maintenance of power grids. Applied Energy 241, 291-301.

62. Doshi, R., & Mehta, A. Way of Transforming the Higher Education by Adoption of the Cloud Learning Management System (CLMS).

63. O. Chapelle, B. Scholkopf, and A. Zien, "Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]," IEEE Transactions on Neural Networks, vol. 20, no. 3, pp. 542–542, 2009. **https://doi.org/10.1109/TNN.2009.2015974**.

64. P. K. Mallapragada, University, Some contributions to semisupervised learning. Michigan State 2010.

65. K. Nigam, A. K. McCallum, S. Thrun, and T. Mitchell, "Text classification from labeled and unlabeled documents using e," Machine learning, vol. 39, no. 2, pp. 103–134, 2000. **https://doi.org/10.1023/A:1007692713085**.

66. J. Bernardo, M. Bayarri, J. Berger, A. Dawid, D. Heckerman, A. Smith, and M. West, "Generative or discriminative? Getting the best of both worlds," Bayesian Stat, vol. 8, no. 3, pp. 3–24, 2007.

67. Hiran, K. K., & Henten, A. (2019). An integrated TOE–DoI framework for cloud computing adoption in the higher education sector: case study of Sub-Saharan Africa, Ethiopia. International Journal of System Assurance Engineering and Management, 1-9.

68. Mahrishi, M., Hiran, K. K., & Doshi, R. (2019, December). Selection of Cloud Service Provider based on Sampled non-functional Attribute Set. In International Conference on Intelligent Systems Design and Applications (pp. 641-648). Springer, Cham.

69. Hiran, K. K. (2021, April). Impact of Driving Factors on Cloud Computing Adoption in the Higher Education. In IOP Conference Series: Materials Science and Engineering (Vol. 1131, No. 1, p. 012016). IOP Publishing.

70. Hiran, K. K., Doshi, R., & Rathi, R. (2014). Security & privacy issues of cloud & grid computing networks. International Journal on Computational Sciences & Applications, 4(1), 83-91.

71. Yeboah, T., Odabi, I., & Hiran, K. K. (2015, April). An integration of round robin with shortest job first algorithm for cloud computing environment. In International Conference On Management, Communication and Technology (Vol. 3, No. 1, pp. 1-5).

72. Peprah, N. A., Hiran, K. K., & Doshi, R. (2020). Politics in the Cloud: A Review of Cloud Technology Applications in the Domain of Politics. Soft Computing: Theories and Applications, 993-1003.

73. R. K. Ando and T. Zhang, "Two-view feature generation model for semi-supervised learning," in Proceedings of the 24th international conference on Machine learning. ACM, 2007, pp. 25–32. **https://doi.org/10.1145/1273496.1273500**.

74. Hiran, K. K., & Doshi, R. (2014). The Proliferation of Smart Devices on Mobile Cloud Computing. Lambert Academic Publishing.

75. P. Viswanath, K. Rajesh, C. Lavanya, and Y. P. Reddy, "A selective incremental approach for transductive nearest neighbor classification," in Recent Advances in Intelligent Computational Systems (RAICS), 2011 IEEE. IEEE, 2011, pp. 221–226.

76. M. Belkin and P. Niyogi, "Semi-supervised learning on riemannian manifolds," Machine learning, vol. 56, no. 1-3, pp. 209–239, 2004. **https://doi.org/10.1023/B:MACH.0000033120.25363.1e**.

77. J. Erman, A. Mahanti, M. Arlitt, I. Cohen, and C. Williamson, "Offline/realtime traffic classification using semi-supervised learning," Per-formance Evaluation, vol. 64, no. 9, pp. 1194–1213, 2007. **https://doi.org/10.1016/j.peva.2007.06.014**.

78. C. Methani, R. Thota, and A. Kale, "Semi-supervised multiple instance learning based domain adaptation for object detection," in Proceedings of the Eighth Indian Conference on Computer Vision, Graphics and Image Processing. ACM, 2012, p. 13. **https://doi.org/10.1145/2425333.2425346**.

79. L. Yao and Z. Ge, "Deep learning of semisupervised process data with hierarchical extreme learning machine and soft sensor application," IEEE Transactions on Industrial Electronics, vol. 65, no. 2, pp. 1490–1498, 2018. **https://doi.org/10.1109/TIE.2017.2733448**.

80. D. Chamberlain, R. Kodgule, D. Ganelin, V. Miglani, and R. R. Fletcher, “Application of semi-supervised deep learning to lung sound analysis,” in Engineering in Medicine and Biology Society (EMBC), 2016 IEEE 38th Annual International Conference of the. IEEE, 2016, pp. 804–807. **https://doi.org/10.1109/EMBC.2016.7590823**.

81. Tagaram Soni Madhulatha. "Chapter 48 Comparison between K-Means and K-Medoids Clustering Algorithms" , Springer Science and Business Media LLC, 2011.

82. Tyagi, S. K. S., Mukherjee, A., Pokhrel, S. R., & Hiran, K. K. (2020). An Intelligent and Optimal Resource Allocation Approach in Sensor Networks for Smart Agri-IoT. IEEE Sensors Journal.

83. Lakhwani, K., Gianey, H. K., Wireko, J. K., & Hiran, K. K. (2020). Internet of Things (IoT): Principles, Paradigms and Applications of IoT. Bpb Publications.

84. Wireko, J. K., Hiran, K. K., & Doshi, R. (2018). Culturally based User Resistance to New Technologies in the Age of IoT in Developing Countries: Perspectives from Ethiopia. International Journal of Emerging Technology and Advanced Engineering, 8(4), 96-105.

85. Logeswari, T., and N. Valarmathi. "Assesment ofapriori and enhanced apriori algorithms in mining itemsets from the KDD database", 2014 International Conference on Green Computing

86. Sadique Ahmad, Awais Adnan. "Machine learning based cognitive skills calculations for different emotional conditions", 2015 IEEE 14th International Conference on Cognitive Informatics & Cognitive Computing (ICCI*CC), 2015

87. Software testing help. Frequent Pattern (FP) Growth Algorithm In Data Mining, L.U. 18 Feb-2021, **https://www.softwaretestinghelp.com/fp-growth-algorithm-data-mining/ Accessed 22-Feb-2021**.

88. "Handbook on Data Management in Information Systems", Springer Science and Business Media LLC, 2003

89. Yanxi Liu. "Study on Application of Apriori Algorithm in Data Mining", 2010 Second International Conference on Computer Modeling and Simulation, 01/2010

90. Gitanjali, Lakhwani K. "A novel approach of sensitive data classification using convolution neural network and logistic regression." Int J Innov Technol Explor Eng 8.8 (2019): 2883-6.

91. Lakhwani, Kamlesh. "Advancement in Artificial Intelligence and its Major Breakthrough: A Systematic Review." Journal of the Gujarat Research Society 21.10s (2019): 288-293.

92. Javatpoint, ML Polynomial Regression, https://www.javatpoint.com/machine-learning-polynomial-regression/ Accessed on 20-Feb-2021

93. Kai Zhang and James T. Kwok, “Clustered Nystrom Method for Large Scale Manifold Learning and Dimension Reduction”, IEEE TRANSACTIONS ON NEURAL NETWORKS, VOL. 21, NO. 10, OCTOBER 2010

94. G. James, D. Witten, T. Hastie, and R. Tibshirani, “An Introduction to Statistical Learning” Springer Texts in Statistics, 2013

95. Vaibhav Verdhan. "Supervised Learning with Python" , Springer Science and Business Media LLC, 2020

96. Khalid Shaikh, Sabitha Krishnan, Rohit Thanki. "Chapter 3 Artificial Intelligence and Learning Algorithms" , Springer Science and Business Media LLC, 2021

97. "Big Data Computing and Communications", Springer Science and Business Media LLC, 2016

98. Z.X. Li, Felipe Lemos Renault, Abdul Orlando Cárdenas Gómez, M.M. Sarafraz et al. "Nanofluids as secondary fluid in the refrigeration system: Experimental data, regression, ANFIS, and NN modeling", International Journal of Heat and Mass Transfer, 2019

Practicals

Experiment No. 1

AIM: Implement and demonstrate the FIND-S algorithm for finding the most specific hypothesis based on a given set of training data samples. Read the training data from a CSV file.

FIND-S Algorithm

```
Initialize h to the most specific hypothesis in H
For each positive training instance x
For each attribute constraint $a_i$ in h
If the constraint $a_i$ is satisfied by x then do nothing
Else replace $a_i$ in h by the next more general constraint that is satisfied by x
Output hypothesis h
```

Training Dataset: ML1.CSV

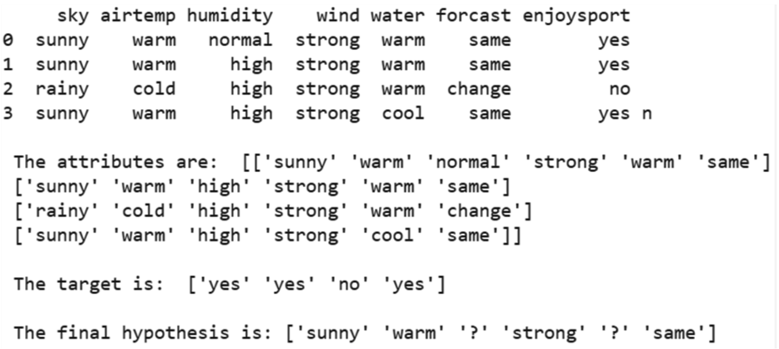
Example	Sky	AirTemp	Humidity	Wind	Water	Forecast	Enjoy Sport
1	sunny	warm	normal	strong	warm	same	yes
2	sunny	warm	high	strong	warm	same	yes
3	rainy	cold	high	strong	warm	change	no
4	sunny	warm	high	strong	cool	same	yes

Program:

```
import pandas as pd

import numpy as np

#to read the data in the csv file

data = pd.read_csv("ML1.csv")

print(data,"n")

#making an array of all the attributes
```

```
d = np.array(data)[:,:-1]

print("\n The attributes are: ",d)

#segragating the target that has positive and negative examples

target = np.array(data)[:,-1]

print("\n The target is: ",target)

#training function to implement find-s algorithm

def train(c,t):

    for i, val in enumerate(t):

        if val == "yes":

            specific_hypothesis = c[i].copy()

            break

    for i, val in enumerate(c):

        if t[i] == "yes":

            for x in range(len(specific_hypothesis)):

                if val[x] != specific_hypothesis[x]:

                    specific_hypothesis[x] = '?'

    return specific_hypothesis

#obtaining the final hypothesis

print("\n The final hypothesis is:",train(d,target))
```

Code Output:



```
     sky airtemp humidity    wind water forcast enjoysport
0  sunny    warm   normal  strong  warm    same        yes
1  sunny    warm     high  strong  warm    same        yes
2  rainy    cold     high  strong  warm  change         no
3  sunny    warm     high  strong  cool    same        yes n

 The attributes are:  [['sunny' 'warm' 'normal' 'strong' 'warm' 'same']
 ['sunny' 'warm' 'high' 'strong' 'warm' 'same']
 ['rainy' 'cold' 'high' 'strong' 'warm' 'change']
 ['sunny' 'warm' 'high' 'strong' 'cool' 'same']]

 The target is:  ['yes' 'yes' 'no' 'yes']

 The final hypothesis is: ['sunny' 'warm' '?' 'strong' '?' 'same']
```

Experiment No. 2

AIM: For a given set of training data examples stored in a .CSV file, implement and demonstrate the Candidate-Elimination algorithm to output a description of the set of all hypotheses consistent with the training examples.

CANDIDATE-ELIMINATION Learning Algorithm

The CANDIDATE-ELIMINATION algorithm computes the version space containing all hypotheses from H that are consistent with an observed sequence of training examples. Initialize G to the set of maximally general hypotheses in H Initialize S to the set of maximally specific hypotheses in H For each training example d, do

- If d is a positive example
    - Remove from G any hypothesis inconsistent with d
    - For each hypothesis s in S that is not consistent with d
        - Remove s from S
        - Add to S all minimal generalizations h of s such that
            - h is consistent with d, and some member of G is more general than h
- Remove from S any hypothesis that is more general than another hypothesis in SIf d is a negative example
    - Remove from S any hypothesis inconsistent with d
    - For each hypothesis g in G that is not consistent with d
        - Remove g from G
        - Add to G all minimal specializations h of g such that
            - h is consistent with d, and some member of S is more specific than h
        - Remove from G any hypothesis that is less general than another hypothesis in G

CANDIDATE- ELIMINATION algorithm using version spaces

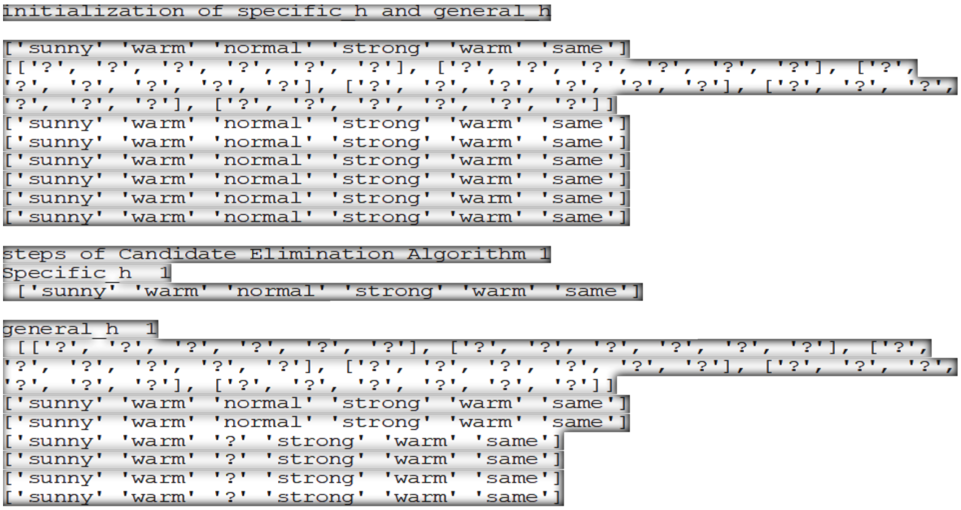
Training Dataset: ML2.CSV

Example	Sky	AirTemp	Humidity	Wind	Water	Forecast	Enjoy Sport
1	sunny	warm	normal	strong	warm	same	yes
2	sunny	warm	high	strong	warm	same	yes
3	rainy	cold	high	strong	warm	change	no
4	sunny	warm	high	strong	cool	change	yes

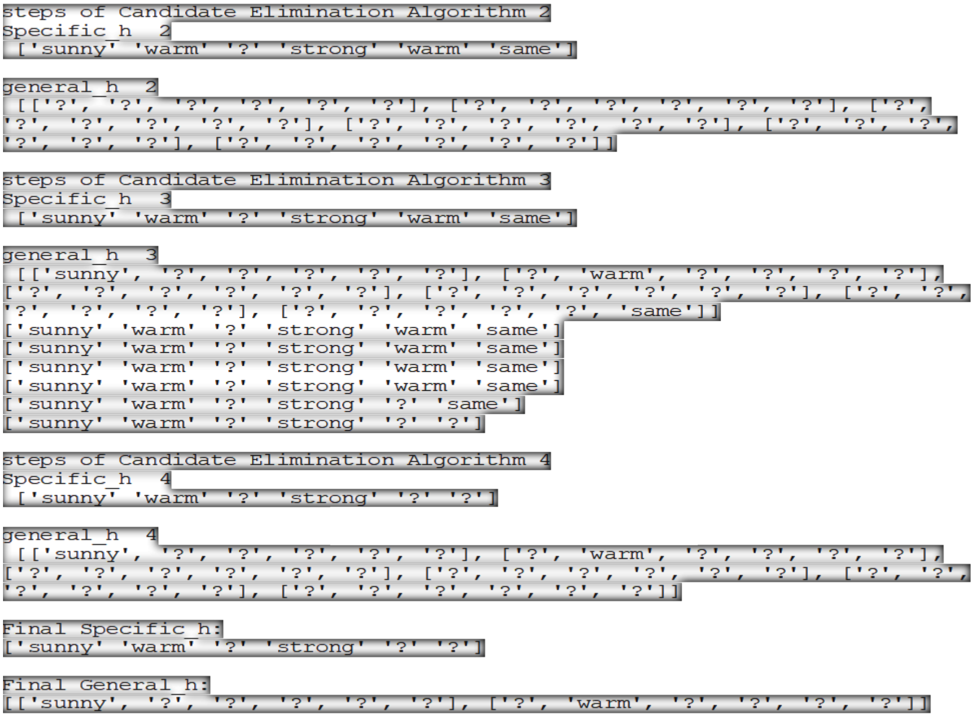
Program:

```
import numpy as np
import pandas as pd
data = pd.DataFrame(data=pd.read_csv('ML2.csv'))
concepts = np.array(data.iloc[:,:-1])
target = np.array(data.iloc[:,-1])
def learn(concepts, target):
    specific_h = concepts[0].copy()
    print("initialization of specific_h and general_h")
    print(specific_h)
    general_h = [["?" for i in range(len(specific_h))] for
i in range(len(specific_h))]
    print(general_h)
    for i, h in enumerate(concepts):
        if target[i] == "Yes":
            for x in range(len(specific_h)):
                if h[x]!= specific_h[x]:
                    specific_h[x] ='?'
                    general_h[x][x] ='?'
                print(specific_h)
        if target[i] == "No":
            for x in range(len(specific_h)):
                if h[x]!= specific_h[x]:
                    general_h[x][x] = specific_h[x]
                else:
                    general_h[x][x] = '?'
        print(" steps of Candidate Elimination Algorithm",i+1)
        print("Specific_h ",i+1,"\n ")
        print(specific_h)
        print("general_h ", i+1, "\n ")
        print(general_h)
    indices = [i for i, val in enumerate(general_h) if val
== ['?', '?', '?', '?', '?', '?']]
    for i in indices:
        general_h.remove(['?', '?', '?', '?', '?', '?'])
    return specific_h, general_h
s_final, g_final = learn(concepts, target)
print("Final Specific_h:", s_final, sep="\n")
print("Final General_h:", g_final, sep="\n")
```

Code Output:



```
initialization of specific_h and general_h

['sunny' 'warm' 'normal' 'strong' 'warm' 'same']
[['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?',
'?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?',
'?', '?', '?'], ['?', '?', '?', '?', '?', '?']]
['sunny' 'warm' 'normal' 'strong' 'warm' 'same']
['sunny' 'warm' 'normal' 'strong' 'warm' 'same']
['sunny' 'warm' 'normal' 'strong' 'warm' 'same']
['sunny' 'warm' 'normal' 'strong' 'warm' 'same']
['sunny' 'warm' 'normal' 'strong' 'warm' 'same']
['sunny' 'warm' 'normal' 'strong' 'warm' 'same']

steps of Candidate Elimination Algorithm 1
Specific_h  1
 ['sunny' 'warm' 'normal' 'strong' 'warm' 'same']

general_h  1
 [['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?',
'?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?',
'?', '?', '?'], ['?', '?', '?', '?', '?', '?']]
['sunny' 'warm' 'normal' 'strong' 'warm' 'same']
['sunny' 'warm' 'normal' 'strong' 'warm' 'same']
['sunny' 'warm' '?' 'strong' 'warm' 'same']
['sunny' 'warm' '?' 'strong' 'warm' 'same']
['sunny' 'warm' '?' 'strong' 'warm' 'same']
['sunny' 'warm' '?' 'strong' 'warm' 'same']
```



```
steps of Candidate Elimination Algorithm 2
Specific_h  2
 ['sunny' 'warm' '?' 'strong' 'warm' 'same']

general_h  2
 [['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?',
'?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?', '?',
'?', '?', '?'], ['?', '?', '?', '?', '?', '?']]

steps of Candidate Elimination Algorithm 3
Specific_h  3
 ['sunny' 'warm' '?' 'strong' 'warm' 'same']

general_h  3
 [['sunny', '?', '?', '?', '?', '?'], ['?', 'warm', '?', '?', '?', '?'],
['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?',
'?', '?', '?', '?'], ['?', '?', '?', '?', '?', 'same']]
['sunny' 'warm' '?' 'strong' 'warm' 'same']
['sunny' 'warm' '?' 'strong' 'warm' 'same']
['sunny' 'warm' '?' 'strong' 'warm' 'same']
['sunny' 'warm' '?' 'strong' 'warm' 'same']
['sunny' 'warm' '?' 'strong' '?' 'same']
['sunny' 'warm' '?' 'strong' '?' '?']

steps of Candidate Elimination Algorithm 4
Specific_h  4
 ['sunny' 'warm' '?' 'strong' '?' '?']

general_h  4
 [['sunny', '?', '?', '?', '?', '?'], ['?', 'warm', '?', '?', '?', '?'],
['?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?'], ['?', '?',
'?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?']]

Final Specific_h:
['sunny' 'warm' '?' 'strong' '?' '?']

Final General_h:
[['sunny', '?', '?', '?', '?', '?'], ['?', 'warm', '?', '?', '?', '?']]
```

Experiment No. 3

AIM: Write a program to demonstrate the working of the decision tree-based ID3 algorithm. Use an appropriate data set for building the decision tree and apply this knowledge to classify a new sample.

ID3 Algorithm

ID3 (Examples, Target attribute, Attributes)

Examples are the training examples. Target attribute is the attribute whose value is to be predicted by the tree. Attributes is a list of other attributes that may be tested by the learned decision tree. Returns a decision tree that correctly classifies the given examples.

```
●  Create a Root node for the tree
●  If all examples are positive, return the single-node tree root, with label = +
●  If all examples are negative, return the single-node tree root, with label = -
●  If attributes are empty, return the single-node tree root, with label = most
   common value of target attribute in examples
●  Otherwise, begin
     o  A ← the attribute from attributes that best* classifies examples
     o  The decision attribute for root ← A
     o  For each possible value, $v_i$, of A,
          ▪  Add a new tree branch below Root, corresponding to the test A = $v_i$
          ▪  Let Examples $_{vi}$, be the subset of examples that have value $v_i$ for A
          ▪  If Examples $_{v}$, is empty
               ◆  Then below this new branch add a leaf node with label =
                  most common value of target attribute in examples
               ◆  Else below this new branch add the subtree ID3(Examples
                  $_{vi}$, Target attribute, Attributes – {A}))
●  End
●  Return Root
```

- The best attribute is the one with the highest information gain

Training Dataset: ML3.CSV

Day	Outlook	Temperature	Humidity	Wind	Answer
D1	sunny	hot	high	weak	no
D2	overcast	hot	high	weak	yes
D3	rain	mild	high	weak	yes
D4	rain	cool	normal	weak	yes
D5	rain	cool	normal	strong	no

Program

```
import pandas as pd

df = pd.read_csv('ML3.csv')

print("\n Input Data Set is:\n", df)

t = df.keys()[-1]

print('Target Attribute is: ', t)

attribute_names = list(df.keys())

attribute_names.remove(t)

print('Predicting Attributes: ', attribute_names)

import math

def entropy(probs):

    return sum( [-prob*math.log(prob, 2) for prob in probs])

def entropy_of_list(ls,value):

    from collections import Counter

    cnt = Counter(x for x in ls)# Counter calculates
the propotion of class

    print('Target attribute class count(Yes/No)=',dict(cnt))

    total_instances = len(ls)

    print("Total no of instances/records associated with {0} is:
{1}".format(value,total_instances ))

    probs = [x / total_instances for x in cnt.values()]  # x means no of YES/NO

    print("Probability of Class {0} is: {1:.4f}".format(min(cnt),min(probs)))
```

```
  print("Probability of Class {0} is: {1:.4f}".format(max(cnt),max(probs)))
    return entropy(probs) # Call Entropy
def information_gain(df, split_attribute, target_attribute,battr):
  print("\n\n-----Information Gain Calculation of ",split_attribute, " ---
-----")
    df_split = df.groupby(split_
attribute) # group the data based on attribute values
    glist=[]
    for gname,group in df_split:
        print('Grouped Attribute Values \n',group)
        glist.append(gname)

    glist.reverse()
    nobs = len(df.index) * 1.0
           df_agg1=df_split.agg({target_attribute:lambda   x:entropy_of_
list(x, glist.pop())})
    df_agg2=df_split.agg({target_attribute :lambda x:len(x)/nobs})

    df_agg1.columns=['Entropy']
    df_agg2.columns=['Proportion']

    # Calculate Information Gain:
    new_entropy = sum( df_agg1['Entropy'] * df_agg2['Proportion'])
    if battr !='S':
      old_entropy = entropy_of_list(df[target_attribute],'S-'+df.iloc[0]
[df.columns.get_loc(battr)])
    else:
      old_entropy = entropy_of_list(df[target_attribute],battr)
    return old_entropy - new_entropy
def id3(df, target_attribute, attribute_names, default_class=None,default_
attr='S'):
    from collections import Counter
    cnt = Counter(x for x in df[target_attribute])# class of YES /NO

    ## First check: Is this split of the dataset homogeneous?
    if len(cnt) == 1:
        return next(iter(cnt))  # next input data set, or raises
StopIteration when EOF is hit.

    ## Second check: Is this split of the dataset empty? if yes,
return a default value
    elif df.empty or (not attribute_names):
        return default_class  # Return None for Empty Data Set

    ## Otherwise: This dataset is ready to be devied up!
    else:
        # Get Default Value for next recursive call of this
function:
        default_class = max(cnt.keys()) #No of YES and NO Class
        # Compute the Information Gain of the attributes:
        gainz=[]
        for attr in attribute_names:
           ig= information_gain(df, attr, target_attribute,default_attr)
            gainz.append(ig)
            print('Information gain of ',attr,' is : ',ig)

        index_of_max = gainz.
index(max(gainz))               # Index of Best Attribute
        best_attr = attribute_names[index_of_
max]            # Choose Best Attribute to split on
        print("\nAttribute with the maximum gain is: ", best_attr)
        # Create an empty tree, to be populated in a moment
     tree = {best_attr:{}} # Initiate the tree with best attribute as a node
```

```
      remaining_attribute_names =[i for i in attribute_names if i != best_
attr]

        # Split dataset-On each split, recursively call this algorithm.
Populate the empty tree with subtrees, which
        # are the result of the recursive call
    for attr_val, data_subset in df.groupby(best_attr):
                 subtree = id3(data_subset,target_attribute, remaining_
attribute_names,default_class,best_attr)
            tree[best_attr][attr_val] = subtree
    return tree
from pprint import pprint
tree = id3(df,t,attribute_names)
print("\nThe Resultant Decision Tree is:")
pprint(tree)
def classify(instance, tree,default=None): # Instance of Play
Tennis with Predicted
    attribute = next(iter(tree)) # Outlook/Humidity/Wind
    if instance[attribute] in tree[attribute].
keys(): # Value of the attributs in  set of Tree keys
        result = tree[attribute][instance[attribute]]
        if isinstance(result, dict): # this is a tree,
delve deeper
            return classify(instance, result)
        else:
            return result # this is a label
    else:
        return default
```

Output

```
Input Data Set is:
     Outlook Temperature Humidity    Wind Answer
0     sunny         hot     high    weak     no
1  overcast         hot     high    weak    yes
2      rain        mild     high    weak    yes
3      rain        cool   normal    weak    yes
4      rain        cool   normal  strong     no
Target Attribute is:  Answer
Predicting Attributes:  ['Outlook', 'Temperature', 'Humidity', 'Wind']

-----Information Gain Calculation of  Outlook  --------
Grouped Attribute Values
     Outlook Temperature Humidity  Wind Answer
1  overcast         hot     high  weak    yes
Grouped Attribute Values
   Outlook Temperature Humidity    Wind Answer
2    rain        mild     high    weak    yes
3    rain        cool   normal    weak    yes
4    rain        cool   normal  strong     no
Grouped Attribute Values
   Outlook Temperature Humidity  Wind Answer
0   sunny         hot     high  weak     no
Target attribute class count(Yes/No)= {'yes': 1}
Total no of instances/records associated with overcast is: 1
Probability of Class yes is: 1.0000
Probability of Class yes is: 1.0000
Target attribute class count(Yes/No)= {'yes': 2, 'no': 1}
Total no of instances/records associated with rain is: 3
Probability of Class no is: 0.3333
Probability of Class yes is: 0.6667
Target attribute class count(Yes/No)= {'no': 1}
Total no of instances/records associated with sunny is: 1
Probability of Class no is: 1.0000
Probability of Class no is: 1.0000
Target attribute class count(Yes/No)= {'no': 2, 'yes': 3}
Total no of instances/records associated with S is: 5
Probability of Class no is: 0.4000
Probability of Class yes is: 0.6000
Information gain of  Outlook  is :  0.4199730940219749

-----Information Gain Calculation of  Temperature  --------
Grouped Attribute Values
   Outlook Temperature Humidity    Wind Answer
3    rain        cool   normal    weak    yes
4    rain        cool   normal  strong     no
Grouped Attribute Values
     Outlook Temperature Humidity  Wind Answer
0     sunny         hot     high  weak     no
1  overcast         hot     high  weak    yes
Grouped Attribute Values
   Outlook Temperature Humidity  Wind Answer
2    rain        mild     high  weak    yes
Target attribute class count(Yes/No)= {'yes': 1, 'no': 1}
Total no of instances/records associated with cool is: 2
Probability of Class no is: 0.5000
Probability of Class yes is: 0.5000
Target attribute class count(Yes/No)= {'no': 1, 'yes': 1}
Total no of instances/records associated with hot is: 2
Probability of Class no is: 0.5000
Probability of Class yes is: 0.5000
Target attribute class count(Yes/No)= {'yes': 1}
Total no of instances/records associated with mild is: 1
Probability of Class yes is: 1.0000
Probability of Class yes is: 1.0000
Target attribute class count(Yes/No)= {'no': 2, 'yes': 3}
Total no of instances/records associated with S is: 5
Probability of Class no is: 0.4000
Probability of Class yes is: 0.6000
Information gain of  Temperature  is :  0.17095059445466854

-----Information Gain Calculation of  Humidity  --------
Grouped Attribute Values
     Outlook Temperature Humidity  Wind Answer
0     sunny         hot     high  weak     no
1  overcast         hot     high  weak    yes
2      rain        mild     high  weak    yes
Grouped Attribute Values
   Outlook Temperature Humidity    Wind Answer
3    rain        cool   normal    weak    yes
4    rain        cool   normal  strong     no
Target attribute class count(Yes/No)= {'no': 1, 'yes': 2}
Total no of instances/records associated with high is: 3
Probability of Class no is: 0.3333
Probability of Class yes is: 0.6667
Target attribute class count(Yes/No)= {'yes': 1, 'no': 1}
Total no of instances/records associated with normal is: 2
Probability of Class no is: 0.5000
Probability of Class yes is: 0.5000
Target attribute class count(Yes/No)= {'no': 2, 'yes': 3}
Total no of instances/records associated with S is: 5
Probability of Class no is: 0.4000
Probability of Class yes is: 0.6000
Information gain of  Humidity  is :  0.01997309402197489

-----Information Gain Calculation of  Wind  --------
Grouped Attribute Values
   Outlook Temperature Humidity    Wind Answer
4    rain        cool   normal  strong     no
Grouped Attribute Values
     Outlook Temperature Humidity  Wind Answer
0     sunny         hot     high  weak     no
1  overcast         hot     high  weak    yes
```

```
2      rain        mild     high  weak    yes
3      rain        cool   normal  weak    yes
Target attribute class count(Yes/No)= {'no': 1}
Total no of instances/records associated with strong is: 1
Probability of Class no is: 1.0000
Probability of Class no is: 1.0000
Target attribute class count(Yes/No)= {'no': 1, 'yes': 3}
Total no of instances/records associated with weak is: 4
Probability of Class no is: 0.2500
Probability of Class yes is: 0.7500
Target attribute class count(Yes/No)= {'no': 2, 'yes': 3}
Total no of instances/records associated with S is: 5
Probability of Class no is: 0.4000
Probability of Class yes is: 0.6000
Information gain of  Wind  is :  0.3219280948873623

Attribute with the maximum gain is:  Outlook
-----Information Gain Calculation of  Temperature  --------
Grouped Attribute Values
   Outlook Temperature Humidity    Wind Answer
3    rain        cool   normal    weak    yes
4    rain        cool   normal  strong     no
Grouped Attribute Values
   Outlook Temperature Humidity  Wind Answer
2    rain        mild     high  weak    yes
Target attribute class count(Yes/No)= {'yes': 1, 'no': 1}
Total no of instances/records associated with cool is: 2
Probability of Class no is: 0.5000
Probability of Class yes is: 0.5000
Target attribute class count(Yes/No)= {'yes': 1}
Total no of instances/records associated with mild is: 1
Probability of Class yes is: 1.0000
Probability of Class yes is: 1.0000
Target attribute class count(Yes/No)= {'yes': 2, 'no': 1}
```

```
Total no of instances/records associated with S-rain is: 3
Probability of Class no is: 0.3333
Probability of Class yes is: 0.6667
Information gain of  Temperature  is :  0.2516291673878229

-----Information Gain Calculation of  Humidity  --------
Grouped Attribute Values
   Outlook Temperature Humidity  Wind Answer
2    rain        mild     high  weak    yes
Grouped Attribute Values
   Outlook Temperature Humidity    Wind Answer
3    rain        cool   normal    weak    yes
4    rain        cool   normal  strong     no
Target attribute class count(Yes/No)= {'yes': 1}
Total no of instances/records associated with high is: 1
Probability of Class yes is: 1.0000
Probability of Class yes is: 1.0000
Target attribute class count(Yes/No)= {'yes': 1, 'no': 1}
Total no of instances/records associated with normal is: 2
Probability of Class no is: 0.5000
Probability of Class yes is: 0.5000
Target attribute class count(Yes/No)= {'yes': 2, 'no': 1}
Total no of instances/records associated with S-rain is: 3
Probability of Class no is: 0.3333
Probability of Class yes is: 0.6667
Information gain of  Humidity  is :  0.2516291673878229

-----Information Gain Calculation of  Wind  --------
Grouped Attribute Values
   Outlook Temperature Humidity    Wind Answer
4    rain        cool   normal  strong     no
Grouped Attribute Values
   Outlook Temperature Humidity  Wind Answer
2    rain        mild     high  weak    yes
3    rain        cool   normal  weak    yes
```

```
Target attribute class count(Yes/No)= {'no': 1}
Total no of instances/records associated with strong is: 1
Probability of Class no is: 1.0000
Probability of Class no is: 1.0000
Target attribute class count(Yes/No)= {'yes': 2}
Total no of instances/records associated with weak is: 2
Probability of Class yes is: 1.0000
Probability of Class yes is: 1.0000
Target attribute class count(Yes/No)= {'yes': 2, 'no': 1}
Total no of instances/records associated with S-rain is: 3
Probability of Class no is: 0.3333
Probability of Class yes is: 0.6667
Information gain of  Wind  is :  0.9182958340544896

Attribute with the maximum gain is:  Wind

The Resultant Decision Tree is:
{'Outlook': {'overcast': 'yes',
             'rain': {'Wind': {'strong': 'no', 'weak': 'yes'}},
             'sunny': 'no'}}
```

Experiment No. 4

AIM: Build an Artificial Neural Network by implementing the Backpropagation algorithm and test the same using appropriate data sets.

BACKPROPAGATION Algorithm

BACKPROPAGATION (*training_example*, η, n_{in}, n_{out}, n_{hidden})

Each training example is a pair of the form $(\vec{\,},t)$, *where* $(\vec{x})$ *is the vector of network input values,* (t) *and is the vector of target network output values.*

η *is the learning rate (e.g., .05).* n_i *is the number of network inputs,* n_{hidden} *the number of units in the hidden layer, and* n_{out} *the number of output units.*

The input from unit i into unit j is denoted x_{ji}, *and the weight from unit i to unit j is denoted* w_{ji}

- Create a feed-forward network with n_i inputs, n_{hidden} hidden units, and n_{out} output units.
- Initialize all network weights to small random numbers
- Until the termination condition is met, Do
    - o For each $(\vec{x},t)$, in training examples, Do

      Propagate the input forward through the network:

      1. Input the instance $\vec{x}$ to the network and compute the output o_u of every unit u in the network

         Propagate the errors backward through the network:

Program

```
import numpy as np

X = np.array(([2, 9], [1, 5], [3, 6]), dtype=float)

y = np.array(([92], [86], [89]), dtype=float)

X = X/np.amax(X,axis=0) # maximum of X array longitudinally
y = y/100

#Sigmoid Function

def sigmoid (x):

    return (1/(1 + np.exp(-x)))

#Derivative of Sigmoid Function
```

```
def derivatives_sigmoid(x):
    return x * (1 - x)
#Variable initialization
epoch=7000 #Setting training iterations
lr=0.1 #Setting learning rate
inputlayer_neurons = 2 #number of features in data set
hiddenlayer_neurons = 3 #number of hidden layers neurons
output_neurons = 1 #number of neurons at output layer
#weight and bias initialization
wh=np.random.uniform(size=(inputlayer_neurons,hiddenlayer_neurons))
bh=np.random.uniform(size=(1,hiddenlayer_neurons))
wout=np.random.uniform(size=(hiddenlayer_neurons,output_neurons))
bout=np.random.uniform(size=(1,output_neurons))
# draws a random range of numbers uniformly of dim x*y
#Forward Propagation
for i in range(epoch):
    hinp1=np.dot(X,wh)
    hinp=hinp1 + bh
    hlayer_act = sigmoid(hinp)
    outinp1=np.dot(hlayer_act,wout)
    outinp= outinp1+ bout
    output = sigmoid(outinp)
    EO = y-output
    outgrad = derivatives_sigmoid(output)
    d_output = EO* outgrad
    EH = d_output.dot(wout.T)
    hiddengrad = derivatives_sigmoid(hlayer_act)
#how much hidden layer wts contributed to error
    d_hiddenlayer = EH * hiddengrad
    wout += hlayer_act.T.dot(d_output) *lr
# dotproduct of nextlayererror and currentlayerop
    bout += np.sum(d_output, axis=0,keepdims=True) *lr
    wh += X.T.dot(d_hiddenlayer) *lr
#bh += np.sum(d_hiddenlayer, axis=0,keepdims=True) *lr
print("Input: \n" + str(X))
print("Actual Output: \n" + str(y))
print("Predicted Output: \n" ,output)
```

Output

```
Input:
[[0.66666667 1.        ]
 [0.33333333 0.55555556]
 [1.         0.66666667]]
Actual Output:
[[92.]
 [86.]
 [89.]]
Predicted Output:
 [[0.99999928]
 [0.99999802]
 [0.9999994 ]]
```

Experiment No. 5

AIM: Write a program to implement the Naïve Bayesian classifier for a sample training data set stored as .CSV file. Compute the accuracy of the classifier, considering few test data sets.

Bayes' Theorem

P(H/E) = P(E/H) P(H)/P(E)

- H- Hypothesis, E-Event / Evidence
- Bayes' Theorem works on conditional probability
- We have been given that if the event has happened or the event is true, then we have to calculate the probability of Hypothesis on this event.
- Means the chances of happening H when the event E happens.
- **P(H) -** It is said **priori (A prior probability),** Probability of H before E happens**.**
- **P(H/E) - Posterior probability,** Probability of E after event E is true.

Training Dataset: Wine Dataset

- The wine dataset contains the results of a chemical analysis of wines grown in a specific area of Italy.
- It contains total 178 samples (data), with 13 chemical analysis (features) recorded for each sample.
- And contains three classes (our target), with no missing values.

Program

```
import numpy as np

import pandas as pd

from sklearn import datasets

wine = datasets.load_wine()

print ("Features: ", wine.feature_names)

print ("Labels: ", wine.target_names)

X=pd.DataFrame(wine['data'])

print(X.head())

print(wine.data.shape)
```

```
y=print (wine.target)
from sklearn.model_selection import train_test_split
X_train,  X_test,  y_train,  y_test  =  train_test_split(wine.data,  wine.
target, test_size=0.30,random_state=109)
from sklearn.naive_bayes import GaussianNB
gnb = GaussianNB()
gnb.fit(X_train, y_train)
y_pred = gnb.predict(X_test)
print(y_pred)
from sklearn import metrics
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
from sklearn.metrics import  confusion_matrix
cm=np.array(confusion_matrix(y_test,y_pred))
print ("confusion Matrix: \n",cm)
```

Output

```
Features:  ['alcohol', 'malic_acid', 'ash', 'alcalinity_of_ash',
'magnesium', 'total_phenols', 'flavanoids', 'nonflavanoid_phenols',
'proanthocyanins', 'color_intensity', 'hue', 'od280/od315_of_diluted_
wines', 'proline']

Labels:  ['class_0' 'class_1' 'class_2']
      0     1     2     3      4     5    ...    7     8     9     10
11      12
0  14.23  1.71  2.43  15.6  127.0  2.80  ...  0.28  2.29  5.64  1.04
3.92  1065.0
1  13.20  1.78  2.14  11.2  100.0  2.65  ...  0.26  1.28  4.38  1.05
3.40  1050.0
2  13.16  2.36  2.67  18.6  101.0  2.80  ...  0.30  2.81  5.68  1.03
3.17  1185.0
3  14.37  1.95  2.50  16.8  113.0  3.85  ...  0.24  2.18  7.80  0.86
3.45  1480.0
4  13.24  2.59  2.87  21.0  118.0  2.80  ...  0.39  1.82  4.32  1.04
2.93   735.0
```

```
[5 rows x 13 columns]
(178, 13)
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2]
[0 0 1 2 0 1 0 0 1 0 2 2 2 2 0 1 1 0 0 1 2 1 0 2 0 0 1 2 0 1 2 1 1 0 1 1
0
 2 2 0 2 1 0 0 0 2 2 0 1 1 2 0 0 2]

Accuracy: 0.9074074074074074

confusion Matrix:
 [[20  1  0]
 [ 2 15  2]
 [ 0  0 14]]
```

Experiment No. 6

AIM: Assuming a set of documents that need to be classified, use the naïve Bayesian Classifier model to perform this task. Built-in Java classes/API can be used to write the program. Calculate the accuracy, precision, and recall for your data set.

Naive Bayes algorithms for learning and classifying text

Examples is a set of text documents along with their target values. V is the set of all possible target values. This function learns the probability terms $P(w_k \mid v_j,)$*, describing the probability that a randomly drawn word from a document in class* v_j *will be the English word* w_k*. It also learns the class prior probabilities* $P(v_j)$*.*

1. collect all words, punctuation, and other tokens that occur in Examples
    - *Vocabulary* ← ***c*** the set of all distinct words and other tokens occurring in any text document from *Examples*

2. calculate the required $P(v_j)$ and $P(w_k \mid v_j)$ probability y terms
    - For each target value v_j in V do
    - $docs_j$ ← the subset of documents from *Examples* for which the target value is *vj*
    - $P(v_j) \leftarrow \mid docs_j \mid \; / \mid \text{Examples} \mid$
    - $Text_j$ ← a single document created by concatenating all members of $docs_j$
    - n ← total number of distinct word positions in $Text_j$
    - for each word w_k in *Vocabulary*
        - n_k ← number of times word $\boldsymbol{w_k}$ occurs in $Text_j$
        - $P(w_k \mid v_j) \leftarrow (n_k + 1) \, / \, (n + \mid Vocabulary \mid)$

Training Dataset: ML6.CSV

	Text Documents	Label
1	I love this sandwich	Pos
2	This is an amazing place	Pos
3	I feel very good about these beers	Pos
4	This is my best work	Pos
5	What an awesome view	Pos
6	I do not like this restaurant	Neg
7	I am tired of this stuff	Neg
8	I can't deal with this	Neg
9	He is my sworn enemy	Neg
10	My boss is horrible	Neg
11	This is an awesome place	Pos
12	I do not like the taste of this juice	Neg
13	I love to dance	Pos
14	I am sick and tired of this place	Neg
15	What a great holiday	Pos
16	That is a bad locality to stay	Neg
17	We will have good fun tomorrow	Pos
18	I went to my enemy's house today	Neg

Program

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn import metrics

msg=pd.read_csv('ML6.csv',names=['message','label'])
print('The dimensions of the dataset',msg.shape)
msg['labelnum']=msg.label.map({'pos':1,'neg':0})
X=msg.message
y=msg.labelnum
#splitting the dataset into train andtestdata
xtrain,xtest,ytrain,ytest=train_test_split(X,y)
```

```
print ('\n The total number of Training Data :',ytrain.shape)
print ('\n The total number of Test Data :',ytest.shape)
#output of count vectoriser is asparsematrix
cv =CountVectorizer()
xtrain_dtm = cv.fit_transform(xtrain)
xtest_dtm=cv.transform(xtest)
print('\n The words or Tokens in the text documents \n')
print(cv.get_feature_names())
df=pd.DataFrame(xtrain_dtm.toarray(),columns=cv.get_feature_names())
print(df)#tabular representation
print(xtrain_dtm) #sparse matrix representation
# Training Naive Bayes (NB) classifier ontrainingdata.
from sklearn.naive_bayes import MultinomialNB
clf = MultinomialNB().fit(xtrain_dtm,ytrain)
predicted = clf.predict(xtest_dtm)
#printing accuracy, Confusion matrix, PrecisionandRecall
from sklearn import metrics
print('\n The Accuracy of classifier is' , metrics.accuracy_
score(ytest, predicted))
print('\n Confusion matrix')
print(metrics.confusion_matrix(ytest, predicted))
print('\nThevalueofPrecision',metrics.precision_score(ytest,predicted))
print('\n The value of Recall' , metrics.recall_score(ytest, predicted))
```

Output

```
The dimensions of the dataset (18, 2)

The total number of Training Data : (13,)

The total number of Test Data : (5,)

The words or Tokens in the text documents

['about', 'am', 'amazing', 'an', 'and', 'awesome', 'bad', 'beers', 'dance',
'do', 'enemy', 'feel', 'fun', 'good', 'great', 'have', 'holiday', 'house',
'is', 'juice', 'like', 'locality', 'love', 'my', 'not', 'of', 'place',
'restaurant', 'sick', 'stay', 'stuff', 'taste', 'that', 'the', 'these',
'this', 'tired', 'to', 'today', 'tomorrow', 'very', 'view', 'we', 'went',
'what', 'will']
    about  am  amazing  an  and  awesome  ...  very  view  we  went  what
will
0       0   0        0   0    0        0  ...     0     0   0     0     0
0
1       0   0        0   0    0        0  ...     0     0   0     0     0
0
2       0   0        0   1    0        1  ...     0     1   0     0     1
0
3       0   1        0   0    0        0  ...     0     0   0     0     0
0
4       0   0        0   0    0        0  ...     0     0   0     1     0
0
5       0   0        0   0    0        0  ...     0     0   0     0     0
0
6       0   0        0   0    0        0  ...     0     0   1     0     0
1
7       0   1        0   0    1        0  ...     0     0   0     0     0
0
8       0   0        0   1    0        1  ...     0     0   0     0     0
0
9       0   0        0   0    0        0  ...     0     0   0     0     1
0
10      0   0        1   1    0        0  ...     0     0   0     0     0
0
11      0   0        0   0    0        0  ...     0     0   0     0     0
0
12      1   0        0   0    0        0  ...     1     0   0     0     0
0
[13 rows x 46 columns]
  (0, 32)    1
  (0, 18)    1
 (0, 6)      1
  (0, 21)    1
  (0, 37)    1
  (0, 29)    1
  (1, 9)     1
  (1, 24)    1
  (1, 20)    1
  (1, 33)    1
  (1, 31)    1
  (1, 25)    1
  (1, 35)    1
  (1, 19)    1
  (2, 44)    1
  (2, 3)     1
  (2, 5)     1
  (2, 41)    1
  (3, 25)    1
  (3, 35)    1
  (3, 1)     1
  (3, 36)    1
  (3, 30)    1
  (4, 37)    1
  (4, 43)    1
  :    :
  (7, 28)    1
  (7, 4)     1
  (7, 26)    1
  (8, 18)    1
  (8, 35)    1
  (8, 3)     1
  (8, 5)     1
  (8, 26)    1
  (9, 44)    1
  (9, 14)    1
  (9, 16)    1
  (10, 18)   1
 (10, 35)    1
```

```
  (10, 3)    1
  (10, 26)   1
  (10, 2)    1
  (11, 37)   1
  (11, 22)   1
  (11, 8)    1
  (12, 13)   1
  (12, 11)   1
  (12, 40)   1
  (12, 0)    1
  (12, 34)   1
  (12, 7)    1

The Accuracy of classifier is 0.8

Confusion matrix
[[3 0]
 [1 1]]

The value of Precision 1.0
The value of Recall 0.5
```

Experiment No. 7

AIM: Write a program to construct a Bayesian network considering medical data. Use this model to demonstrate the diagnosis of heart patients using standard Heart Disease Data Set. You can use Java/Python ML library classes/API.

Training Dataset: heartdisease1.CSV

age	Gender	Family	diet	Lifestyle	cholesterol	heart disease
0	0	1	1	3	0	1
0	1	1	1	3	0	1
1	0	0	0	2	1	1
4	0	1	1	3	2	0
3	1	1	0	0	2	0
2	0	1	1	1	0	1
4	0	1	0	2	0	1
0	0	1	1	3	0	1
3	1	1	0	0	2	0
1	1	0	0	0	2	1
4	1	0	1	2	0	1
4	0	1	1	3	2	0
2	1	0	0	0	0	0
2	0	1	1	1	0	1
3	1	1	0	0	1	0
0	0	1	0	0	2	1
1	1	0	1	2	1	1
3	1	1	1	0	1	0

Program

```python
import pandas as pd
data=pd.read_csv('heartdisease1.csv')
heart_disease=pd.DataFrame(data)
print(heart_disease)

from pgmpy.models import BayesianModel
```

```
model=BayesianModel([
('age','Lifestyle'),
('Gender','Lifestyle'),
('Family','heartdisease'),
('diet','cholestrol'),
('Lifestyle','diet'),
('cholestrol','heartdisease'),
('diet','cholestrol')
])

from pgmpy.estimators import MaximumLikelihoodEstimator
model.fit(heart_disease, estimator=MaximumLikelihoodEstimator)
from pgmpy.inference import VariableElimination
HeartDisease_infer = VariableElimination(model)

print('For age Enter { SuperSeniorCitizen:0, SeniorCitizen:1,
MiddleAged:2, Youth:3, Teen:4 }')
print('For Gender Enter { Male:0, Female:1 }')
print('For Family History Enter { yes:1, No:0 }')
print('For diet Enter { High:0, Medium:1 }')
print('For lifeStyle Enter { Athlete:0, Active:1, Moderate:2,
Sedentary:3 }')
print('For cholesterol Enter { High:0, BorderLine:1, Normal:2 }')

q = HeartDisease_infer.query(variables=['heartdisease'], evidence={
    'age':int(input('Enter age :')),
    'Gender':int(input('Enter Gender :')),
    'Family':int(input('Enter Family history :')),
    'diet':int(input('Enter diet :')),
    'Lifestyle':int(input('Enter Lifestyle :')),
    'cholesterol':int(input('Enter cholesterol :'))
    })

print(q)
```

Output:

```
    age  Gender  Family  diet  Lifestyle  cholesterol  heartdisease
0     0       0       1     1          3            0             1
1     0       1       1     1          3            0             1
2     1       0       0     0          2            1             1
3     4       0       1     1          3            2             0
4     3       1       1     0          0            2             0
5     2       0       1     1          1            0             1
6     4       0       1     0          2            0             1
7     0       0       1     1          3            0             1
8     3       1       1     0          0            2             0
9     1       1       0     0          0            2             1
10    4       1       0     1          2            0             1
11    4       0       1     1          3            2             0
12    2       1       0     0          0            0             0
13    2       0       1     1          1            0             1
14    3       1       1     0          0            1             0
15    0       0       1     0          0            2             1
16    1       1       0     1          2            1             1
17    3       1       1     1          0            1             0

For  age  Enter  {  SuperSeniorCitizen:0,  SeniorCitizen:1,  MiddleAged:2,
Youth:3, Teen:4 }


For Gender Enter { Male:0, Female:1 }


For Family History Enter { yes:1, No:0 }


For diet Enter { High:0, Medium:1 }


For lifeStyle Enter { Athlete:0, Active:1, Moderate:2, Sedentary:3 }
For cholesterol Enter { High:0, BorderLine:1, Normal:2 }


Enter age :1
Enter Gender :0
Enter Family history :0
Enter diet :1
Enter Lifestyle :2
Enter cholesterol :2

Finding Elimination Order: : : 0it [00:00, ?it/s]
0it [00:00, ?it/s]
+-----------------+---------------------+
| heartdisease    |   phi(heartdisease) |
+=================+=====================+
| heartdisease(0) |              0.0000 |
+-----------------+---------------------+
| heartdisease(1) |              1.0000 |
+-----------------+---------------------+
```

Experiment No. 8

AIM: Apply EM algorithm to cluster a set of data stored in a .CSV file. Use the same data set for clustering using the *k*-means algorithm. Compare the results of these two algorithms and comment on the quality of clustering. You can add Java/Python ML library classes/API in the program**.**

K-Means Algorithm

1. Load data set.
2. Clusters the data into *k* groups where *k* is predefined.
3. Select *k* points at random as cluster centers.
4. Assign objects to their closest cluster center according to the *Euclidean distance* function.
5. Calculate the centroid or mean of all objects in each cluster.
6. Repeat steps 3, 4 and 5 until the same points are assigned to each cluster in consecutive rounds.

EM algorithm

These are the two basic steps of the EM algorithm, namely **E Step or Expectation Step or Estimation Step** and **M Step or Maximization Step**

Estimation step:

- Initialize μ_k, $\sum$ k and $\prod_k$ by some random values, or by the K means clustering results or by hierarchical clustering results
- Then for those given parameter values, estimate the value of the latent variables(i.e., γ_k)

Maximization Step:

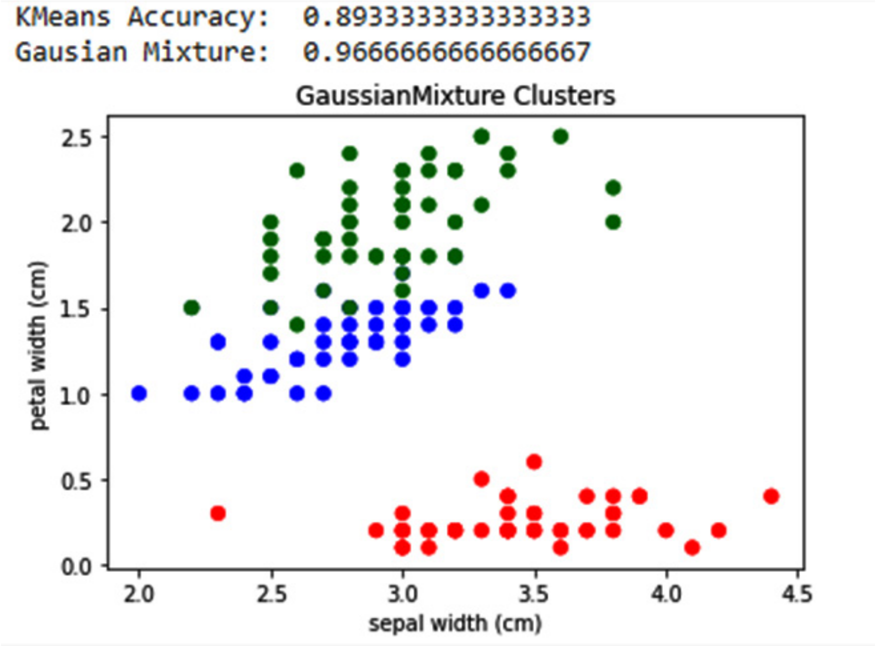
- Update the value of the parameters (i.e., μ_k, $\sum$k and $\prod_k$) calculated using the ML method
  1. Load data set.
  2. Initialize the mean μ_k, the covariance matrix $\sum$k and the mixing coefficients.

     $\prod_k$ by some random values (or other values).
  3. Compute the γ_k values for all k.
  4. Again estimate all the parameters using the current γ_k values.
  5. Compute log-likelihood function.
  6. Put some convergence criterion.

Program

```
import matplotlib.pyplot as plt
from sklearn import preprocessing
from sklearn.datasets import load_iris
import pandas as pd
import numpy as np
iris=load_iris()
x=pd.DataFrame(iris.data, columns=iris.feature_names)
y=pd.DataFrame(iris.target, columns=['target'])
x.head()
colormap=np.array(['red','blue','green'])
plt.title('Actual Clusters')
plt.scatter(x['sepal width (cm)'], x['petal width (cm)'], c=colormap[y.
            target])
plt.xlabel('sepal width (cm)')
plt.ylabel('petal width (cm)')
plt.title('KMeans Clusters')
from sklearn.mixture import GaussianMixture
gm = GaussianMixture(n_components=3).fit(x).predict(x)
plt.scatter(x['sepal width (cm)'], x['petal width (cm)'], c=colormap[gm])
plt.xlabel('sepal width (cm)')
plt.ylabel('petal width (cm)')
plt.title('GaussianMixture Clusters')
from sklearn.cluster import KMeans
km = KMeans(n_clusters=3)
km.fit(x)
from sklearn import metrics as m
print("KMeans Accuracy: ", m.accuracy_score(y, km.labels_))
print("Gausian Mixture: ", m.accuracy_score(y, gm))
```

Output:



```
KMeans Accuracy:  0.8933333333333333
Gausian Mixture:  0.9666666666666667
```

Experiment No. 9

AIM: Write a program to implement the *k*-nearest neighbor algorithm to classify the iris data set. Print both correct and wrong predictions. Java/Python ML library classes can be used for this problem.

K-Nearest Neighbor Algorithm

Training algorithm:

- For each training example (x, f (x)), add the example to the list of training examples

  Classification algorithm:

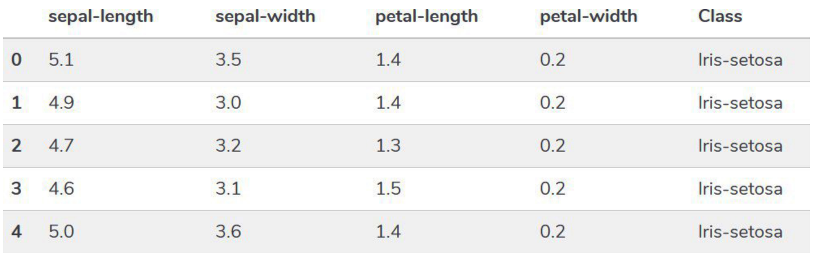
- Given a query instance x_q to be classified,
  - o Let x_1 . . .x_k denote the k instances from training examples that are nearest tox_q
  - o Return

$$\hat{f}(x_q) \leftarrow \frac{\sum_{i=1}^{k} f(x_i)}{k}$$

  - o Where $f(x_i)$ function to calculate the mean value of the k nearest training examples.

Training Dataset: IRIS DATASET

Iris Plants Dataset: Dataset contains 150 instances (50 in each of three classes) Number of Attributes: 4 numeric, predictive attributes and the Class



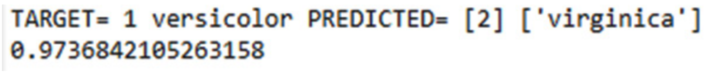
	sepal-length	sepal-width	petal-length	petal-width	Class
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

Program

```
from sklearn.datasets import load_iris

from sklearn.neighbors import KNeighborsClassifier

from sklearn.model_selection import train_test_split
```

```
import numpy as np
dataset=load_iris()
#print(dataset)
X_train,X_test,y_train,y_test=train_test_
            split(dataset["data"],dataset["target"],random_state=0)
kn=KNeighborsClassifier(n_neighbors=1)
kn.fit(X_train,y_train)
for i in range(len(X_test)):
    x=X_test[i]
    x_new=np.array([x])
prediction=kn.predict(x_new)
print("TARGET=",y_test[i],dataset["target_names"][y_
            test[i]],"PREDICTED=",prediction,dataset["target_names"]
            [prediction])
print(kn.score(X_test,y_test))
```

Output



```
TARGET= 1 versicolor PREDICTED= [2] ['virginica']
0.9736842105263158
```

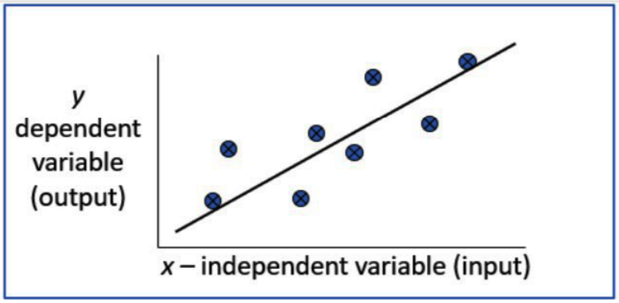
Experiment No. 10

AIM: Implement the non-parametric locally weighted regression algorithm to fit data points. Select the appropriate data set for your experiment and draw graphs.

Locally Weighted Regression Algorithm

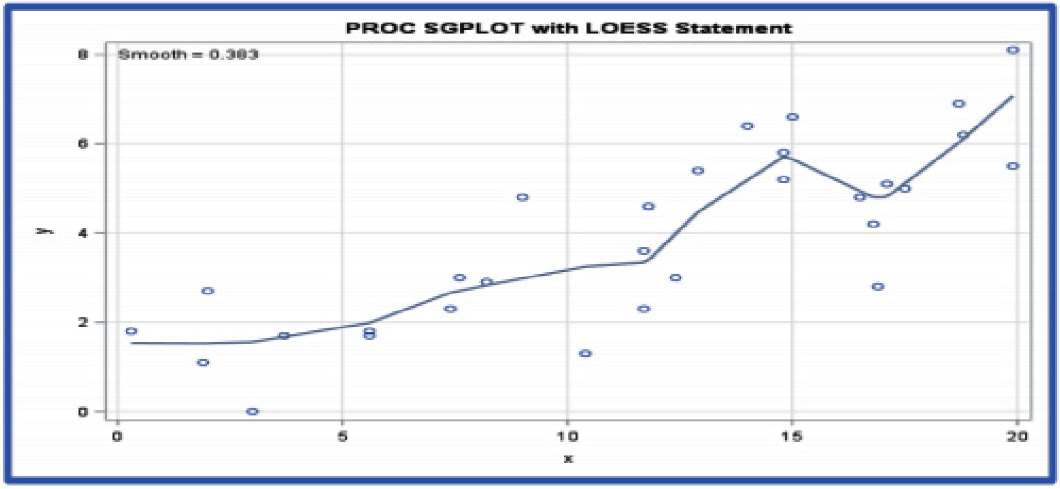
Regression:

- Regression is a technique from statistics that are used to predict values of the desired target quantity when the target quantity is continuous.
- In regression, we seek to identify (or estimate) a continuous variable y associated with a given input vector x.
    - o y is called the dependent variable.
    - o x is called the independent variable.



Loess/Lowess Regression:

Loess regression is a non-parametric technique that uses locally weighted regression to fit a smooth curve through points in a scatter plot.



Lowess Algorithm:

- Locally weighted regression is a very powerful non-parametric model used in statistical learning.
- Given a dataset X, y, we attempt to find a model parameter β(x) that minimizes the residual sum of weighted squared errors.
- The weights are given by a kernel function (k or w) which can be chosen arbitrarily.

Algorithm

1. Read the given data sample to X and the curve (linear or non-linear) to Y.
2. Set the value for smoothening parameter or the free parameter say τ.
3. Set the bias / Point of interest set x0 which is a subset of X.
4. Determine the weight matrix using:

$$w(x, x_o) = e^{-\frac{(x-x_o)^2}{2\tau^2}}$$

5. Determine the value of model term parameter β using:

$$\hat{\beta}(x_o) = (X^T W X)^{-1} X^T W y$$

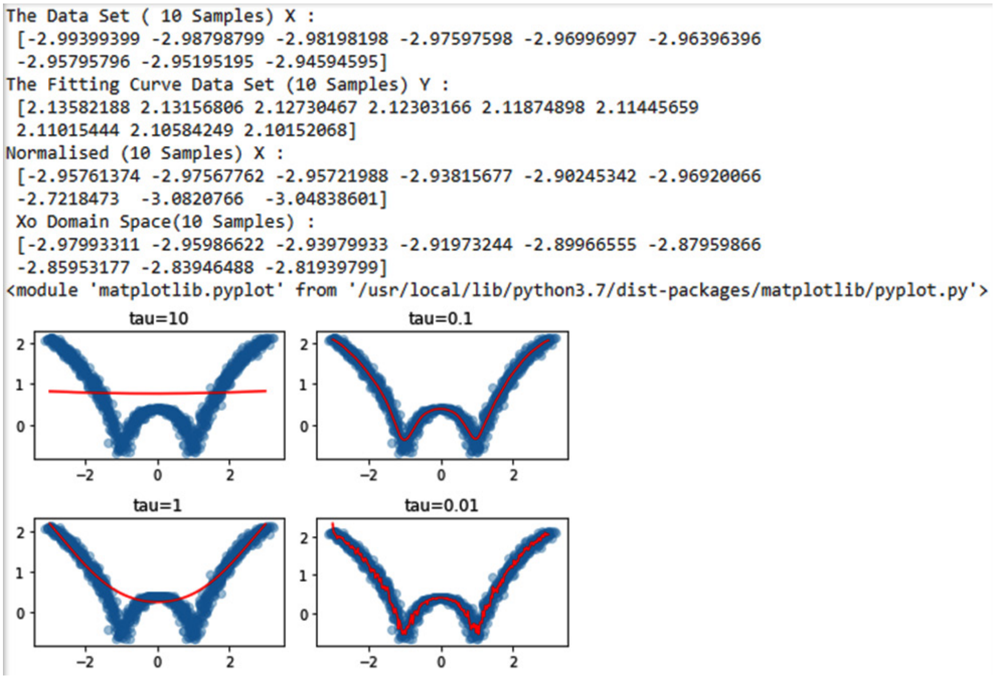
6. Prediction =x0*β:

Program

```
import numpy as np
import matplotlib.pyplot as plt

def local_regression(x0, X, Y, tau):# add bias term
 x0 = np.r_[1, x0] # Add one to avoid the loss in information
 X = np.c_[np.ones(len(X)), X]

 # fit model: normal equations with kernel
 xw = X.T * radial_kernel(x0, X, tau) # XTranspose * W

 beta = np.linalg.pinv(xw @ X) @ xw @ Y #@ Matrix
           Multiplication or Dot Product

 # predict value
 return x0 @ beta # @ Matrix Multiplication or Dot Product for
```

```
            prediction
def radial_kernel(x0, X, tau):
 return np.exp(np.sum((X - x0) ** 2, axis=1) / (-2 * tau * tau))
# Weight or Radial Kernal Bias Function

n = 1000
# generate dataset
X = np.linspace(-3, 3, num=n)
print("The Data Set ( 10 Samples) X :\n",X[1:10])
Y = np.log(np.abs(X ** 2 - 1) + .5)
print("The Fitting Curve Data Set (10 Samples) Y :\n",Y[1:10])
# jitter X
X += np.random.normal(scale=.1, size=n)
print("Normalised (10 Samples) X :\n",X[1:10])

domain = np.linspace(-3, 3, num=300)
print(" Xo Domain Space(10 Samples) :\n",domain[1:10])

def plot_lwr():
 # prediction through regression for tau =10.0
 tau=10.0
 prediction = [local_regression(x0, X, Y, tau) for x0 in domain]
 fig, axs = plt.subplots(2, 2)
 axs[0, 0].plot(domain,prediction,color="red")
 axs[0, 0].set_title('tau=%g' % tau)
 axs[0, 0].scatter(X, Y, alpha=.3)

# prediction through regression for tau =1.0
 tau=1.0
 prediction = [local_regression(x0, X, Y, tau) for x0 in domain]
 axs[1, 0].plot(domain,prediction,color="red")
 axs[1, 0].set_title('tau=%g' % tau)
 axs[1, 0].scatter(X, Y, alpha=.3)

# prediction through regression for tau =0.1
 tau=0.1
 prediction = [local_regression(x0, X, Y, tau) for x0 in domain]
 axs[0, 1].plot(domain,prediction,color="red")
 axs[0, 1].set_title('tau=%g' % tau)
 axs[0, 1].scatter(X, Y, alpha=.3)

# prediction through regression for tau =0.01
 tau=0.01
 prediction = [local_regression(x0, X, Y, tau) for x0 in domain]
 axs[1, 1].plot(domain,prediction,color="red")
 axs[1, 1].set_title('tau=%g' % tau)
 axs[1, 1].scatter(X, Y, alpha=.3)

 fig.tight_layout()


 return plt

plot_lwr()
```

Output



```
The Data Set ( 10 Samples) X :
 [-2.99399399 -2.98798799 -2.98198198 -2.97597598 -2.96996997 -2.96396396
 -2.95795796 -2.95195195 -2.94594595]
The Fitting Curve Data Set (10 Samples) Y :
 [2.13582188 2.13156806 2.12730467 2.12303166 2.11874898 2.11445659
 2.11015444 2.10584249 2.10152068]
Normalised (10 Samples) X :
 [-2.95761374 -2.97567762 -2.95721988 -2.93815677 -2.90245342 -2.96920066
 -2.7218473  -3.0820766  -3.04838601]
 Xo Domain Space(10 Samples) :
 [-2.97993311 -2.95986622 -2.93979933 -2.91973244 -2.89966555 -2.87959866
 -2.85953177 -2.83946488 -2.81939799]
<module 'matplotlib.pyplot' from '/usr/local/lib/python3.7/dist-packages/matplotlib/pyplot.py'>
```

Model Test Paper – 1

Multiple Choice Questions

1. **Machine Learning is a field of AI consisting of learning algorithms that**
   a. Improve their performance
   b. At executing some task
   c. Over time with experience
   d. All of the above

2. **Different learning methods do not include**
   a. Memorization
   b. Analogy
   c. Deduction
   d. Introduction

3. **What is perceptron?**
   a. a single layer feed-forward neural network with pre-processing
   b. an auto-associative neural network
   c. a double layer auto-associative neural network
   d. a neural network that contains feedback

4. **The problem of finding hidden structures in unlabeled data is called**
   a. Supervised learning
   b. Unsupervised learning
   c. Reinforcement learning
   d. None of the above

5. **An e-commerce company wants to segment their customers into distinct groups to send appropriate offers, this is an example of**
   a) Unsupervised learning
   b) Supervised learning
   c) Reinforcement learning
   d) None of the above

6. **Which of the following is/are the application/s of unsupervised learning?**
   a. Clustering
   b. Anomaly detection
   c. Neural networks
   d. All of the above

7. **In the k-mean clustering algorithm k is a:**
   a. Positive integer
   b. Negative integer
   c. Random number
   d. Complex number

8. **The shortest distance between the two closest points of two different clusters is considered as:**
   a. Single linkage
   b. Complete linkage
   c. Average linkage
   d. Centroid linkage

9. **What is machine learning?**
   a. The autonomous acquisition of knowledge through the use of computer programs
   b. The autonomous acquisition of knowledge through the use of manual programs
   c. The selective acquisition of knowledge through the use of computer programs
   d. The selective acquisition of knowledge through the use of manual programs

10. **Which of the following is true for neural networks?**
    (i) The training time depends on the size of the network.
    (ii) Neural networks can be simulated on a conventional computer.
    (iii) Artificial neurons are identical in operation to biological ones.
    a. All of the mentioned
    b. (ii) is true
    c. and (ii) are true
    d. None of the mentioned

Fill in the blanks:

1. Predicting the price of a house based on floor area, number of rooms, etc. is the example of ____________________ machine learning.
2. Branch of an engineering student is a ____________________ feature.
3. In the case of machine learning, ____________________ come in as input and the ____________________ come out as output.
4. ____________________ processes the uncategorized data and divides them into different clusters.
5. Q-learning technique is a ____________________ method.

Short Answer Questions:

1. What is machine learning?
2. Explain the applications of machine learning.
3. Explain the classification errors.
4. Find the difference between PCA and ICA.
5. Explain the support vector machines (SVM).

Descriptive Questions:

1. Explain the different steps used to perform machine learning.
2. Differentiate between supervised, unsupervised, and reinforcement learning.
3. What is the difference between regression and classification? Explain with example.
4. Define clustering. Explain the difference between clustering and classification.
5. Explain the difference between the single linkage and complete linkage method.
6. What are the various advantages and disadvantages of the Apriori Algorithm?
7. Explain the step-by-step working of agglomerative hierarchical clustering.
8. Define GMM also explain the working of the Gaussian mixture model in detail.

9. Write short notes on (any two)

    (a) Linear quadratic regulation (b) Q-learning

    (b) MDPS (d) Value function approximation

10. Consider the following transactional dataset and find the frequent patterns and generate association rules using the FP-growth algorithm.

Sr. No.	Transections	Item list
1.	Tr1	It1, It2, It3
2.	Tr2	It2, It3, It4
3.	Tr3	It4, It5
4.	Tr4	It1, It2, It4
5.	Tr5	It1, It2, It3, It5
6.	Tr6	It1, It2, It3, It4

Consider: Minimum support=50%, Minimum confidence=60%.

Model Test Paper – 2

Multiple Choice Questions

1. **What is true about machine learning?**
   a. Machine Learning (ML) is the field of computer science.
   b. ML is a type of artificial intelligence that extracts patterns out of raw data by using an algorithm or method.
   c. The main focus of ML is to allow computer systems to learn from experience without being explicitly programmed or human intervention.
   d. All of the above

2. **In which of the following learning the teacher returns reward and punishment to the learner?**
   a. Active learning
   b. Reinforcement learning
   c. Supervised learning
   d. Unsupervised learning

3. **Technique used in unsupervised learning accepts__________________ , discovers new patterns of information that were previously unknown or undetected.**
   a. Unlabeled data items
   b. Labeled data items
   c. Classified data items
   d. Clustered data items

4. **A 4-input neuron has weights 1, 2, 3 and 4. The transfer function is linear with the constant of proportionality being equal to 2. The inputs are 4, 10, 5 and 20 respectively. What will be the output?**
   a. 238
   b. 76

c. 119
d. 123

5. **Hierarchical clustering is an example of:**
    a. Exclusive partitioning
    b. Agglomerative clustering
    c. Overlapping clustering
    d. Probabilistic clustering

6. **In language understanding, the levels of knowledge does not include**
    a. Phonological
    b. Syntactic
    c. Empirical
    d. Logical

7. **What takes input as an object described by a set of attributes?**
    a. Tree
    b. Graph
    c. Decision graph
    d. Decision tree

8. **Different learning methods do not include**
    a. Memorization
    b. Analogy
    c. Deduction
    d. Introduction

9. **How many types are available in machine learning?**
    a. 1
    b. 2
    c. 3
    d. 4

10. **Which is used for utility functions in the game-playing algorithm?**
    a. Linear polynomial
    b. Weighted polynomial
    c. Polynomial
    d. Linear weighted polynomial

Fill in the blanks:

1. Semi-supervised learning may refer to either inductive or ____________________ learning.
2. The Markov's choice procedure is a model for ____________________ results.
3. The Q-learning algorithms has____________________ steps.
4. The SARSA acronym stands for____________________ .
5. A model-based learning method which is built on various model parameters is called ____________________.

Short Answer Questions:

1. What is the learning system?
2. Explain the goals of machine learning.
3. What is the K- nearest neighbor?
4. Explain overfitting.
5. What is the difference between supervised and unsupervised learning?

Descriptive Questions:

1. What is the difference between linear and non-linear discriminative classification?
2. Explain in brief the Bellman Equation.
3. What is bagging and boosting? How is KNN different from k-means clustering?
4. Explain the aspect of developing a learning system.
5. Explain the generative probabilistic classification.
6. What is Bayes theorem? How is it useful in a machine learning context?
7. What is the difference between generative and discriminative models?
8. Explain the AdaBoost boost algorithm. What is support vector machine? Explain briefly.
9. Explain perceptron.

   (a) Explain the EM algorithm. (b). What is PCA?
10. Explain the Hidden Markov Model (HMMs). What is spectral clustering? Explain briefly the segmenting indexing.

Model Test Paper – 3

Multiple Choice Questions

1. **ML is a field of AI consisting of learning algorithms that?**

   a. Improve their performance
   b. At executing some task
   c. Over time with experience
   d. All of the above

2. **When performing regression or classification, which of the following is the correct way to pre-process the data?**

   a. Normalize the data → PCA → training
   b. PCA → normalize PCA output → training
   c. Normalize the data → PCA → normalize PCA output → training
   d. None of the above

3. **Which of the following is a widely used and effective machine learning algorithm based on the idea of bagging?**

   a. Decision tree
   b. Regression
   c. Classification
   d. Random Forest

4. **Which of the following is a disadvantage of decision trees?**

   a. Factor analysis
   b. Decision trees are robust to outliers
   c. Decision trees are prone to be overfit
   d. None of the above

5. **To find the minimum or the maximum of a function, we set the gradient to zero because:**

   a. The value of the gradient at the extrema of a function is always zero

   b. B. Depends on the type of problem

   c. C. Both A and B

   d. D. None of the above

6. **Which of the following is a reasonable way to select the number of principal components "k"?**

   a. Choose k to be the smallest value so that at least 99% of the variance is retained. – answer

   b. Choose k to be 99% of m (k = 0.99*m, rounded to the nearest integer).

   c. Choose k to be the largest value so that 99% of the variance is retained.

   d. Use the elbow method

7. **What is the purpose of performing cross-validation?**

   a. To assess the predictive performance of the models

   b. To judge how the trained model performs outside the sample on test data

   c. Both A and B

   d. None of the above

8. **The most widely used metrics and tools to assess a classification model are:**

   a. Confusion matrix

   b. Cost-sensitive accuracy

   c. The area under the ROC curve

   d. All of the above

9. **In which of the following cases will the k-means clustering fail to give good results? 1) Data points with outliers 2) Data points with different densities 3) Data points with nonconvex shapes**

   a. 1 and 2

   b. 2 and 3

   c. 1, 2, and 3

   d. 1 and 3

10. **In the model-based learning methods, an iterative process takes place on the ML models that are built based on various model parameters, they are called?**
    a. mini-batches
    b. optimized parameters
    c. hyperparameters
    d. super parameters

Fill in the blanks:

1. The technique used in unsupervised learning accepts__________________, discovers new patterns of information that were previously unknown or undetected.
2. In __________________ learning the teacher returns reward and punishment to the learner.
3. __________________ is used for utility functions in game playing algorithm.
4. Hierarchical clustering is a/an __________________ machine learning technique of data clustering that constructs the hierarchy of clusters.
5. Association rule learning is a technique of __________________ learning.

Short Answer Questions:

1. Differentiate between training data and testing data.
2. What is confusion matrix? Explain with help of suitable example.
3. How to calculate the root mean square error? Explain with the help of a suitable example.
4. What is an information gain?
5. What is the independent component analysis?

Descriptive Questions:

1. What is a recommender system? How is machine learning useful in recommender systems?
2. Explain the various stages involved in designing a learning system.
3. In which cases Naive Bayes is useful in classification? Why?

4. What is the Gibbs Algorithm? What is its suitability in machine learning?
5. What type of problems are best suited for decision tree learning? Explain with the help of a suitable example.
6. What is entropy? How do we employ mutual information for classification between a positive and negative class?
7. Explain the various issues involved in decision tree learning.
8. What is an inductive bias? Is there any effect on classification due to bias?
9. The values of independent variable x and dependent value y are given below:

X	Y
0	2
1	3
2	5
3	4
4	6

Find the least square regression line y=ax+b. Estimate the value of y when x is 10.

10. Explain the k-nearest neighbour techniques with an example.

Model Test Paper – 4

Multiple Choice Questions

1. **The Apriori algorithm is used for__________________.**
    a. Market basket analysis
    b. Backpropagation
    c. Neural network
    d. Supervised training

2. **The FP-growth algorithm is an important algorithm of__________________.**
    a. Association rule learning
    b. Supervised learning
    c. Reinforcement learning
    d. Neural network

3. **Probabilistic clustering is also known as**
    a. Soft clustering
    b. Hard clustering
    c. Alpha clustering
    d. Beta clustering

4. **The farthest distance between the two points of two different clusters is considered as:**
    a. Single linkage
    b. Complete linkage
    c. Average linkage
    d. Centroid linkage

5. **Suppose we want to perform clustering on the geometric location of a house and clusters can be of different shapes and sizes. Which is the appropriate method for this approach?**
    a. Decision trees

b. Density-based clustering
c. Model-based clustering
d. K-means clustering

6. **Which of the following is not supervised learning?**
a. PCA
b. Decision tree
c. Linear regression
d. Naive Bayesian

7. **Which of these are categorical features?**
a. Height of a person
b. Price of petroleum
c. Mother tongue of a person
d. Amount of rainfall in a day

8. **Which of the following is a supervised learning problem?**
I. Predicting the outcome of a cricket match as a win or loss based on historical data
II. Recommending a movie to an existing user on a website like IMDb based on the search history (including other users)
III. Predicting the gender of a person from his/her image. You are given the data of 1 million images along with the gender
IV. Given the class labels of old news articles, predicting the class of a new news article from its content. Class of a news article can be such as sports, politics, technology, etc.
a. I, II, III
b. I, III, IV
c. II, III, IV
d. I, II, III, IV

9. **Q-learning technique is an ___________________ method.**
a. Off-Policy method
b. (b) On-Policy method
c. (c) On-Off Policy method
d. (d) Off-Off Policy method

10. **A learning agent in any reinforcement learning algorithm it's policy can be of two types:**
    a. On-Policy and Off-Policy
    b. Off-Policy and Off-Policy
    c. On-Policy and On-Policy
    d. None of these

Fill in the blanks:

1. Q-Learning technique is an ____________________ method and uses the greedy way to learn the Q-value.

2. SARSA technique is an ____________________ and uses the action performed by the current policy to learn the Q-value.

3. ____________________ is a real-time machine learning application that determines the emotion or opinion of the speaker or the writer.

4. Exclusive, agglomerative, overlapping, and probabilistic are four methods of __________________.

5. Labeled data is used to train the model in__________________, whereas in __________________ unlabeled data is used to train the model.

Short Answer Questions:

1. Discuss the k-nearest neighbor algorithm.
2. Discuss the locally weighted regression.
3. Discuss the binomial distribution.
4. Write a short note on information gain.
5. Write a short note on the confusion matrix.

Descriptive Questions:

1. Write a candidate elimination algorithm. Apply the algorithm to obtain the final version space for the following training sample.

Sr. No.	Sky	Air temp	Humidity	Wind	Water	Forecast	Enjoy Sport
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	Yes
3	Rainy	Cold	High	Strong	Warm	Change	No
4	Sunny	Warm	High	Strong	Cool	Change	Yes

2. Write an algorithm for the backpropagation algorithm which uses the stochastic gradient descent method. Comment on the effect of adding momentum to the network.

3. Discuss the learning tasks and Q-learning in the context of reinforcement learning.

4. Explain the Naïve Bayes classifier with an example.

5. Explain the EM algorithm in detail.

6. Write Bayes theorem. What is the relationship between Bayes theorem and the problem of concept learning?

7. Write a note on Occam's razor and minimum description principle.

8. Define the following terms: a) Sample error; b) True error; c) Expected value.

9. Give decision tree representations for the following Boolean functions:
   - $A \vee (B \wedge C)$
   - $(A \wedge B) \vee (C \wedge D)$

10. What is entropy? How do we employ mutual information for classification between a positive and negative class?

Index

A

accuracy 119

activation function

  about 179

  need for 179

agglomerative clustering method 66

agglomerative hierarchical clustering (AHC)

  about 74, 75

  working 74

Akaike Information Criterion (AIC) 117, 118

algorithm 65

Apriori algorithm

  about 83

  cons 87

  example 84-87

  frequent itemsets 83

  pros 87

  steps 84

  versus FP-growth algorithm 93

artificial intelligence (AI) 2

Artificial Neural Network (ANN)

  about 176

  applications 177

  characteristics 178

  supervised ANN 176

  unsupervised ANN 176

  versus Biological Neural Network (BNN) 177

association rule 15

association rule learning

  about 81

  applications 93

types 81
working 82
association rule learning, metrics
confidence 82
lift 83
support 82
automatic translation 10
average linkage 76

B

backpropagation algorithm 182, 183
backpropagation network
advantages 184
disadvantages 184
backpropagation network, types
about 183
recurrent backpropagation 183
static backpropagation 183
Bayes 38
Bellman Expectation Equation
about 139, 140
defining 140-142
Bernoulli Naive Model 40
binary classification 13, 37
binary logistic regression 48
Biological Neural Network (BNN)
versus Artificial Neural Network (ANN) 177
breadth-first search (BFS) 83

C

centroid linkage 76
chain rule 182
change probability 137
Chi-squared 115
classification algorithm
about 13, 37, 127
decision tree 129
logistic regression 129
Naive Bayes Classifier 128
Nearest Neighbor algorithm 128
neural network 129
classification metrics
about 119
accuracy 119
AUC curve 120
F1 score 120
precision 120
recall 120
clustering 14, 65, 66
clustering metrics
about 122
Dunn index 122
elbow method 122
Silhouette Coefficient 122
collaborative filtering
about 170
classes 171
technique types 171
complete linkage 76
Construct FP tree 90
content-based filtering (CF)
about 173
advantages 175
architecture 173, 174
disadvantages 175
filtering component 174
profile learner 174

content-based filtering (CF), components
  content analyzer 174

D

decision tree
  about 41, 129
  advantages 43
  disadvantages 43
  example 42
  working 41
decision tree regression 127
deep learning
  about 184, 185
  advantages 187
  applications 188
  disadvantages 187
  importance 186, 187
  versus machine learning 185, 186
dendrogram 73
dependent variable 31
Dyna Architecture 156

E

elbow method 122
embedded method 116
Exclusive partitioning 66
Expectation-Maximization (EM)
  about 97
  E-step 98
  in GMM 98
  M-step 98
  process 97

F

feature selection
  about 114
  embedded method 116
  filters 114
  wrappers 115, 116
filters, feature selection
  analysis of variance (ANOVA) 115
  Chi-squared 115
  linear discriminant analysis (LDA) 115
  Pearson correlation 115
FP-growth algorithm
  about 87
  example 88, 89
  frequent pattern (FP) tree 87
  steps 88
  versus Apriori algorithm 93
frequent pattern (FP) tree
  about 87
  advantages 92
  disadvantages 92
  mining 91
function f
  about 111
  inference 112
  prediction 111, 112

G

Gaussian distribution 95
Gaussian Mixture Model (GMM)
  about 95-97
  example 99, 100, 101
  Expectation-Maximization 98
Gaussian Naïve Model 39

generalization 33

Generative Learning Model 128

Google Neural Machine Translation (GNMT) 10

H

Hash Tree (HT) 83

Hierarchical Cluster Analysis (HCA) 73

hierarchical clustering

- about 73, 130
- approach 73
- dendrogram 77
- dendrogram, creating 78
- distance clusters, measuring 75, 76
- example 78-80
- usage 73

High Variance Model 34

hyperplane 51

I

information criterion (IC) 117

input variable 31

K

k-means clustering

- about 67, 130
- algorithm 67
- example 68-72
- flowchart 68

K-Nearest Neighbors (K-NN) algorithm

- about 43, 130
- usage 43-45

L

L2 regularization 127

lasso regression 127

left child node 127

Linear Binary Classifier 178

linear combination 176

linearly separable data 53

linear regression

- about 47, 125
- types 34

linkage methods 75

logistic regression

- about 46, 47, 129
- examples 49
- ordinal logistic regression 49
- versus linear regression 47

logistic regression, types

- about 48
- binary logistic regression 48
- multinomial logistic regression 49

M

machine learning

- versus deep learning 185, 186

machine learning (ML)

- about 2-4
- advantages 16, 17
- disadvantages 17
- software tools 18-20
- versus traditional programming 4, 5

machine learning (ML) applications

- about 8
- automatic language translation 10
- email malware filtering 9
- email spam filtering 9
- image recognition 9
- medical diagnosis 10

online fraud detection 10
product recommendations 8
self-driving cars 9
sentiment analysis 9
social media features 8
speech recognition 9
stock market trading 10
machine learning (ML) process
about 5
data collection 6
data gathering 6
data, preparing 6
evaluation 7
hypermeter tuning 8
hyperparameter tuning 7
model type, selecting 6
prediction 8
training 7
machine learning (ML) types
about 11
supervised learning 11, 12
unsupervised learning 13, 14
margin 52
Markov chain 137
Markov Decision Process (MDP)
about 137-139
applications 139
Markov process 137
maximum marginal hyperplane (MMH) 52
Mean Absolute Error (MAE) 121
memory-based approach 172
model
about 153
learning 153, 154
model-based approach 172
model-based reinforcement learning 152
model-based strategy
versus model-free procedure 153
model evaluation
about 119
classification metrics 119
clustering metrics 122
regression metrics 121
statistical learning algorithm 123
model-free procedure
versus model-based strategy 153
model selection
about 116
probabilistic measures 117
re-sampling method 116
Monte Carlo
methods 143
policy evaluation 143, 144
multiclass classification 13, 37
multicollinearity 32
multilayer network 181
multinomial logistic regression 49
Multinomial Naive Model 40
multiple linear regression 36

N

Naïve 38
Naïve Bayes
advantages 40
disadvantages 40

Naïve Bayes Algorithms
  applications 40
Naïve Bayes classifier algorithm
  about 37, 38, 128
  example 39
  principle 38
  working 39
Naive Bayes Model, types
  about 39
  Bernoulli Naive Model 40
  Gaussian Naive Model 39
  Multinomial Naive Model 40
Nearest Neighbor algorithm 128
neural network 129, 176
non-linear activation function
  need for 179
non-linearly separated data 53

O

operator 137
ordinal logistic regression 49
outliers 31
output variable 31
overfitting
  about 33
  terminology 33
overlapping clustering 66

P

parent node 127
Pearson correlation 115
perceptron
  about 178
  working 179-181
perceptron, types
  multilayer 179
  single layer 179
policy iteration 145, 146
polynomial regression 126
predictor 31
principal components 131
principal components analysis (PCA) 131
probabilistic clustering 66, 93, 94
probabilistic measures
  about 117
  akaike information criterion (AIC) 117, 118
  minimum description length (MDL) 118

Q

Q-learning 148
  versus State-Action-Reward-State -Action (SARSA) 151
Q-learning algorithm
  about 148
  process 149, 150
Q-Table 148

R

Random Forest algorithm
  about 54
  advantages 56
  applications 57
  disadvantages 56
  working 55, 56
Recommended Systems (RS)
  about 164, 165
  applications 168

challenges 169, 170
functions 167, 168
importance 165
types 165, 166
Recommended Systems (RS), approach
collaborative filtering method 166
content-based method 166
hybrid method 167
recurrent backpropagation 183
regression algorithm
about 12, 13, 30, 124
versus classification algorithm 113
regression metrics
about 121
Mean Absolute Error (MAE) 121
Mean Squared Error (MSE) 121
Root Mean Squared Error (RMSE) 121
Root Mean Squared Log Error (RMSLE) 121
regression, terminologies
about 31
dependent variable 31
independent variable 31
multicollinearity 32
outliers 31
regression, types
decision tree regression 127
lasso regression 127
linear regression 125
logistic regression 125
polynomial regression 126
ridge regression 127
support vector regression 126
reinforcement learning (RL) 15, 136, 137
re-sampling method
about 116
bootstrap 117
k-fold cross-validation 117
random split 117
time-based split 117
response variable 31
ridge regression 127
ridge regression penalty 127
right child node 127
Root Mean Squared Error (RMSE) 121
Root Mean Squared Log Error (RMSLE) 121
root node 127

S

semi-supervised learning 136
Silhouette Coefficient 122
similarity index-based technique 170
simple linear regression
about 34-36
goal 35
single linkage 76
soft clustering 93
State-Action-Reward-State-Action (SARSA)
about 150, 151
algorithm 151, 152
versus Q-learning 151
static backpropagation 183
statistical learning
about 108-111
algorithm 123

supervised learning
  about 11, 12, 28, 29, 124
  classification algorithm 13, 127
  example 154, 155
  regression algorithm 12, 13, 124
  versus unsupervised learning 113
supervised learning, types
  about 29
  regression 30
Support Vector Machine (SVM) algorithm
  about 49-51
  advantages 53
  applications 53
  disadvantages 53
  hyperplane 51
  margin 52
  support vectors 52
  working 52
Support Vector Machine (SVM) types
  about 52
  linear SVM 52
  non-linear SVM 53
support vector regression
  about 126
  boundary line 127
  hyper plane 127
  kernel 127
  support vector 127
support vectors 52

T

target variable 31
techniques, collaborative filtering
  memory-based approach 172
  model-based approach 172
traditional programming
  versus machine learning (ML) 4, 5

U

underfitting
  about 33
  terminology 33
unsupervised learning
  about 13, 14, 64, 129
  association rule 15
  clustering 14
  hierarchical clustering 130
  k-means clustering 130
  k- Nearest neighbors 130
  need for 65
  principal components analysis (PCA) 131
  reinforcement learning 15
  versus supervised learning 113
  working 64

V

value iteration 145-147