Data Mining and Predictive Modelling – Module 4

Dr. Dinesh Kumar
Associate Professor
Bennett University (Times of India Group)
Plot Nos 8, 11, TechZone 2, Greater Noida, Uttar Pradesh 201310

Course Brief

CSET228 - Data Mining and Predictive Modelling

Course Type - Specialized Core – II L-T-P Format 3-0-2 Credits - 4

COURSE SUMMARY

This course exposes multiple techniques of understanding and analysing the data from a mathematical point of view. In addition, they will also use multiple predictive models to analyse the future trend. This will be done in a purely statistical manner.

COURSE-SPECIFIC LEARNING OUTCOMES (CO)

CO1: To articulate data preparation for data mining and analyzing based on pre-processing techniques.

CO2: To examine predictive analysis in various use cases.

CO3: To make use of exploratory data analysis to gain insights and prepare data for predictive modelling.

Components	Percentage
Mid-Term	20
End-Term	40
Course Certification	10
Lab Continuous Evaluation	10
End Term Lab Examination	20

Syllabus

Module 1 (8 hours)

Purpose of Data mining, Procedures of Data Mining, Functionality of Data Mining, Knowledge data discovery process, Data, and attribute type, Properties of data, Discrete and continuous attributes, Dataset types, Data quality measurement, Noise Analysis and its importance, Techniques of Data pre-processing, Aggregation, Sampling, Curse of dimensionality, Dimensionality reduction, Feature selection and generation, Discretization and vectorization, Binarization, Attribute transformation correlation, Association rule mining, Apriori algorithm, Rule generation, Pattern Mining in: Multilevel, Multidimensional Space Pattern Mining.

Module 2 (7 hours)

Rule-based reasoning, Memory-based reasoning, measuring data similarity, Similarity Metrics: Distance-based measure, Information based measures, Set similarity measure, Jaccard Index, Sorenson Dice Coefficient, Model Selection Problem, Error Analysis, Case study, Startups in DataAnalysis.

Module 3 (14 hours)

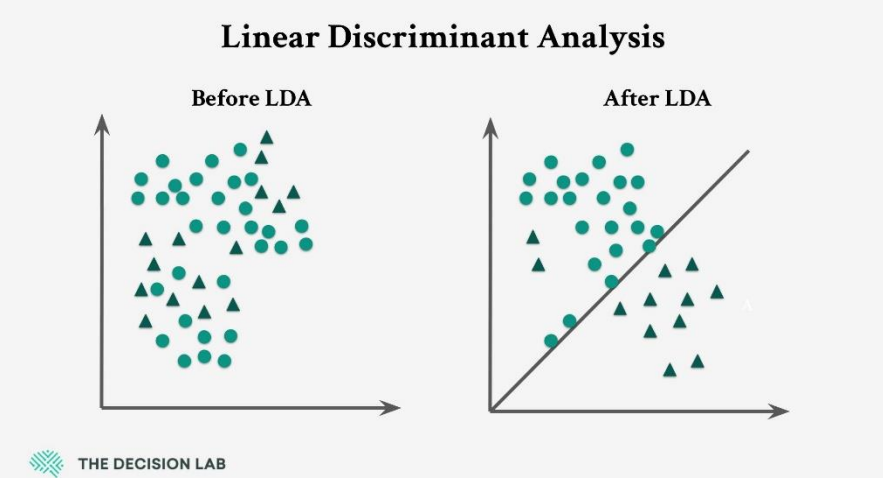
Supervised Learning; Classification: K-NN, Performance Matrix, Linear Regression, Logistic Regression, Decision Tree, SVM, ANN **Unsupervised Learning;** Clustering: K-means, Outlier analysis in classification and clustering, **Exploratory data analysis**, Data summarization and visualization, Dataset exploration, Data Exploration Tools,Interactive Data Exploration, Predictive models, Design Principles, Parametric Models, Non-Parametric Models, one way and two way ANOVA, Regression Analysis

Module 4 (8 hours)

Linear discriminant analysis, Fisher discriminant analysis, Time series Model: ARMA, ARIMA, ARFIMA, Factor Analysis, Recommendation System and Collaborative Filtering, Multidimensional Scaling, Mining Textual Data, Temporal mining, Spatial mining, Visual and audio data mining, Social Impacts of data mining.

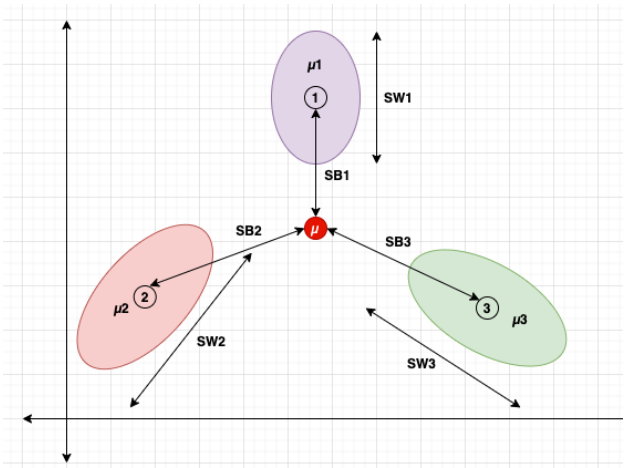
Linear Discriminant Analysis

- *Linear discriminant analysis (LDA),* also known as normal discriminant analysis (NDA) or discriminant function analysis (DFA), **is a powerful dimensionality reduction technique widely used in machine learning and statistics**.
- LDA enhances classification accuracy by identifying the optimal linear combinations of features that separate different classes within a dataset.
- By reducing complexity while preserving critical class distinctions, LDA improves model performance in applications such as pattern recognition, face recognition, and text classification.



How LDA Works?

- LDA utilizes Fisher linear discriminant method to separate the classes.
- Fisher's linear discriminant is a classification method that projects high-dimensional data onto one — dimensional space and performs classification in this one — dimensional space.
- The projection maximizes the distance between the means of the classes while minimizing the variance within each class.



Classes: 1, 2 and 3

Mean of Classes: μ1, μ2 and μ3

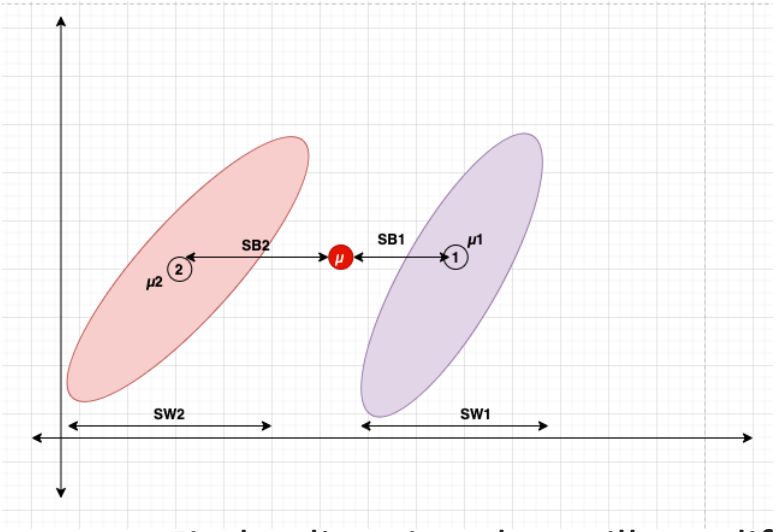
Scatter Between Classes: SB1, SB2 and SB3

Scatter Within Classes: SW1, SW2 and SW3

Dataset Mean: μ

The idea is to Maximize the SBs, Scatter Between Classes and Minimize the SWs, Scatter Within Classes.

Formula



$$J(W) = \frac{(\mu_1 - \mu_2)^2}{S_1^2 + S_2^2}$$

$Where,$

$\mu_1 \rightarrow Mean\ of\ Class\ 1$

$\mu_2 \rightarrow Mean\ of\ Class\ 2$

$S_1 \rightarrow Scatter\ Matrix\ of\ Class\ 1$

$S_2 \rightarrow Scatter\ Matrix\ of\ Class\ 2$

Motivation

- Find a direction that will amplify the inter-class difference.
- **Maximize** (squared) difference between the projected means $(\mu_1 - \mu_2)^2$
- **Minimize** the projected scatter within each class $S_1^2 + S_2^2$

Formula: Scatter

Scatter

$$S^2 = \sum (W^T X - \mu)^2$$

$$=> S_i^2 = \sum (W^T X - W^T x_i)^2$$

$$=> S_i^2 = \sum (W^T X - W^T x_i)(W^T X - W^T x_i)^T$$

$$=> S_i^2 = \sum W^T (X - x_i)(X - x_i)^T W$$

$$=> S_i^2 = W^T (\sum (X - x_i)(X - x_i)^T) W$$

$$=> S_i^2 = W^T (\sum (X - x_i)(X - x_i)^T) W$$

$$=> S_i^2 = W^T * \sigma_i * W$$

$Where,$

$S \rightarrow Scatter$

$W \rightarrow Direction\ of\ Projection$

$X \rightarrow Sample\ Dataset$

$\mu_i \rightarrow W^T x_i \rightarrow Mean\ along\ projection$

$\mu \rightarrow Mean\ of\ a\ Class$

$\sigma \rightarrow Covariance\ of\ a\ Class$

Mean Difference

$$(\mu_1 - \mu_2)^2 = (W^T x_1 - W^T x_2)^2$$

$$=> (\mu_1 - \mu_2)^2 = W^T (x_1 - x_2)(x_1 - x_2)^T W$$

$$=> (\mu_1 - \mu_2)^2 = W^T S_b W$$

$Where,$

$S_b \rightarrow (x_1 - x_2)(x_1 - x_2)^T \rightarrow Scatter\ Matrix\ Between\ Classes$

Formula: Scatter

Mean Difference

$$S_1^2 + S_2^2 = W^T \sigma_1 W + W^T \sigma_2 W$$

$$S_1^2 + S_2^2 = W^T (\sigma_1 + \sigma_2) W$$

$$S_1^2 + S_2^2 = W^T S_w W$$

$$Where,$$

$$S_w \rightarrow \sigma_1 + \sigma_2 \rightarrow Scatter\ Matrix\ Within\ Classes$$

Fischer Index

$$J(W) = \frac{(\mu_1 - \mu_2)^2}{S_1^2 + S_2^2}$$

$$=> J(W) = \frac{W^T S_b W}{W^T S_w W}$$

$$=> \frac{\delta J(W)}{\delta W} = \frac{W^T S_w W * S_b W - W^T S_b W * S_w W}{(W^T S_w W)^2}$$

$$=> W^T S_w W * S_b W - W^T S_b W * S_w W = 0$$

$$=> S_b W = \frac{W^T S_b W}{W^T S_w W} * S_w W$$

$$=> S_b W = \lambda S_w W$$

$$\approx> S_b V = \lambda S_w V$$

$$Where,$$

$$V \rightarrow Eigenvector$$

$$W = [V_1, V_2, ..., V_{C-1}]$$

$$\therefore Cannot\ seek\ C-1\ eigenvalues.\ C\ is\ no.\ of\ Classes.$$

Example

- **Step 1:** Compute Within Class Scatter matrix (SW)

$$S_W = S_{c1} + S_{c2}$$

$Where,$

$S_W \rightarrow Within\ Class\ Scatter\ Matrix$

$S_{c1} \rightarrow Covariance\ Matrix\ of\ Class\ 1$

$S_{c2} \rightarrow Covariance\ Matrix\ of\ Class\ 2$

X1	X2	Class
4	1	1
2	4	1
2	3	1
3	6	1
4	4	1
9	10	2
6	8	2
9	5	2
8	7	2
10	8	2

Find covariance matrix for each class

Class 1 :

$$\mu = [\mu_{x1} \quad \mu_{x2}]$$

$Where,$

$\mu \rightarrow Mean\ Matrix$

$\mu_{x1} \rightarrow Mean\ Feature\ 1$

$\mu_{x2} \rightarrow Mean\ Feature\ 2$

Mean Matrix

$$\mu_{c1} = [3 \quad 3.6]$$

	X1	X2	Class
	4	1	1
	2	4	1
	2	3	1
	3	6	1
	4	4	1
Mean	**3**	**3.6**	

Example

Covariance

$$S_k = \sum_{k=1}^{n} (x_{1k} - \mu_{x1})(x_{2k} - \mu_{x2})^T$$

$Where,$

$k \rightarrow Row$

$\mu_{x1} \rightarrow Mean\ of\ Feature\ 1\ (X_1)\ for\ Class\ 1$

$\mu_{x2} \rightarrow Mean\ of\ Feature\ 2\ (X_2)\ for\ Class\ 1$

	X1	X1 - μ1	X2	X2 - μ2	Covariance
	4	1	1	-2.6	S1=[1,-2.6]
	2	-1	4	0.4	S2=[-1,0.4]
	2	-1	3	-0.6	S3=[-1,-0.6]
	3	0	6	2.4	S4=[0,2.4]
	4	1	4	0.4	S5=[1,0.4]
Mean	**3**		**3.6**		

$$S_1 = \begin{bmatrix} 1 \\ -2.6 \end{bmatrix} * \begin{bmatrix} 1 & -2.6 \end{bmatrix} = \begin{bmatrix} 1 & -2.6 \\ -2.6 & 6.76 \end{bmatrix}$$

.
.
.
.

$$S_5 = \begin{bmatrix} 1 \\ 0.4 \end{bmatrix} * \begin{bmatrix} 1 & 0.4 \end{bmatrix} = \begin{bmatrix} 1 & 0.4 \\ 0.4 & 0.16 \end{bmatrix}$$

Adding **S1** to **S5** we will get **Sc1**

$$S_{c1} = S_1 + S_2 + S_3 + S4 + S_5 = \begin{bmatrix} 0.8 & -0.4 \\ -0.4 & 2.6 \end{bmatrix}$$

Example

Class 2:

	X1	X2
	9	10
	6	8
	9	5
	8	7
	10	8
Mean	**8.4**	**7.6**

Mean matrix $\mu_{c2} = [8.4 \quad 7.6]$

Similarly like **Sc1**, adding **S6 to S10**, we will get covariance **Sc2**

$$S_{c2} = S_6 + S_7 + S_8 + S_9 + S_{10} = \begin{bmatrix} 1.84 & -0.04 \\ -0.04 & 2.64 \end{bmatrix}$$

	X1	X1 - μ1	X2	X2 - μ2	Covariance
	9	0.6	10	2.4	S6=[0.6,-2.4]
	6	-2.4	8	0.4	S7=[-2.4,0.4]
	9	0.6	5	-2.6	S8=[0.6,-2.6]
	8	-0.4	7	-0.6	S9=[-0.4,-0.6]
	10	1.6	8	0.4	S10=[1.6,0.4]
Mean	**8.4**		**7.6**		

Adding **Sc1** and **Sc2** will give us **Sw**, **Within-Class Scatter Matrix**

$$S_W = \begin{bmatrix} 0.8 & -0.4 \\ -0.4 & 2.6 \end{bmatrix} + \begin{bmatrix} 1.84 & -0.04 \\ -0.04 & 2.64 \end{bmatrix} = \begin{bmatrix} 2.64 & -0.44 \\ -0.44 & 5.28 \end{bmatrix}$$

Example

- **Step 2**: Compute Between Class Scatter matrix (SB)

$$S_B = (\mu_{c1} - \mu_{c2}) * (\mu_{c1} - \mu_{c2})^T$$

$$Where,$$

$$S_B \rightarrow Between\ Class\ Scatter\ Matrix$$

We already have mean for each features for **Class 1** and **Class 2**

$$\mu_{c1} = [3 \quad 3.6] \qquad \mu_{c2} = [8.4 \quad 7.6] \qquad \mu_{c1} - \mu_{c2} = [3 \quad 3.6] - [8.4 \quad 7.6] = [-5.4 \quad -4]$$

$$S_B = [-5.4 \quad -4] * \begin{bmatrix} -5.4 \\ -4 \end{bmatrix} = \begin{bmatrix} 29.16 & 21.6 \\ 21.6 & 16 \end{bmatrix}$$

Example

- **Step 3**: Find the best LDA projection vector

Similar to PCA we find this using ***Eigenvector*** with largest ***Eigenvalue*** to find the best projection vector. The ***Eigenvector*** can be represented in below form.

$$A - \lambda I = 0$$

$$Here,$$

$$A \rightarrow S_W^{-1} * S_B$$

$$I \rightarrow Identity\ Matrix$$

$$So,$$

$$S_W^{-1} * S_B - \lambda I = 0$$

Example

We already have **SW** and **SB**

$$=> \begin{bmatrix} 2.64 & -0.44 \\ -0.44 & 5.28 \end{bmatrix}^{-1} * \begin{bmatrix} 29.16 & 21.6 \\ 21.6 & 16 \end{bmatrix} - \lambda * \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = 0$$

$$=> \begin{bmatrix} 11.89 & 8.81 \\ 5.08 & 3.76 \end{bmatrix} - \lambda * \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = 0$$

$$=> \begin{bmatrix} 11.89 - \lambda & 8.81 \\ 5.08 & 3.76 - \lambda \end{bmatrix} = 0$$

Solving for **lambda** we get, **lambda = 15.65**, **highest value**. Now, solve the correspondent vector for each values of lambda

Example

$B * X = 0$

$Where,$

$B \rightarrow Matrix$

$X \rightarrow Vector$

$$For,\ \lambda = 15.65,$$

$$B = A - \lambda I = \begin{bmatrix} 11.89 - 15.65 & 8.81 \\ 5.08 & 3.76 - 15.65 \end{bmatrix} = \begin{bmatrix} -3.76 & 8.81 \\ 5.08 & -11.89 \end{bmatrix}$$

$$=> So,\ For,\ B * X = 0$$

$$=> \begin{bmatrix} -3.76 & 8.81 \\ 5.08 & -11.89 \end{bmatrix} * \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = 0$$

$$=> -3.76v_1 + 8.81v_2 = 0\ and\ 5.08v_1 - 11.89v_2 = 0$$

$$=> \begin{cases} v_1 = 2.34v_2 \\ v_2 = v_2 \end{cases}$$

$$=> \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = v_2 * \begin{bmatrix} 2.34 \\ 1 \end{bmatrix}$$

$$=> For, v_2 = 1$$

$$=> \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} 2.34 \\ 1 \end{bmatrix}$$

Example

- **Step 4**: Transforming the samples onto the new subspace.

$$Y = X * W$$

$Where,$

$Y \rightarrow New\ Dataset$

$X \rightarrow Original\ Dataset$

$W \rightarrow Selected\ Eigenvector$

$$\begin{bmatrix} 4 & 1 \\ 2 & 4 \\ 2 & 3 \\ 3 & 6 \\ 4 & 4 \\ 9 & 10 \\ 6 & 8 \\ 9 & 5 \\ 8 & 7 \\ 10 & 8 \end{bmatrix} * \begin{bmatrix} 2.34 \\ 1 \end{bmatrix} = \begin{bmatrix} 10.36 \\ 8.68 \\ 7.68 \\ 13.02 \\ 13.36 \\ 31.06 \\ 22.04 \\ 26.06 \\ 25.72 \\ 31.4 \end{bmatrix}$$

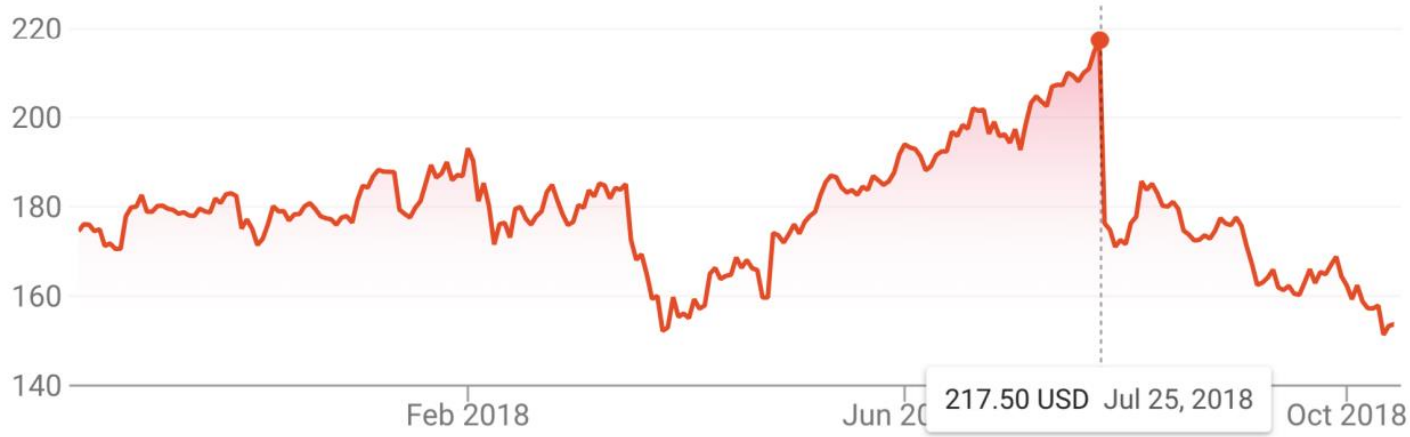
So using **LDA** we have transformed

X1	X2
4	1
2	4
2	3
3	6
4	4
9	10
6	8
9	5
8	7
10	8

X
10.36
8.68
7.68
13.02
13.36
31.36
22.04
26.06
25.72
31.4

Time Series

- A time series is a sequential set of data points, measured typically over successive times.
- Time series analysis comprises methods for analyzing time series data in order to extract meaningful statistics and other characteristics of the data.



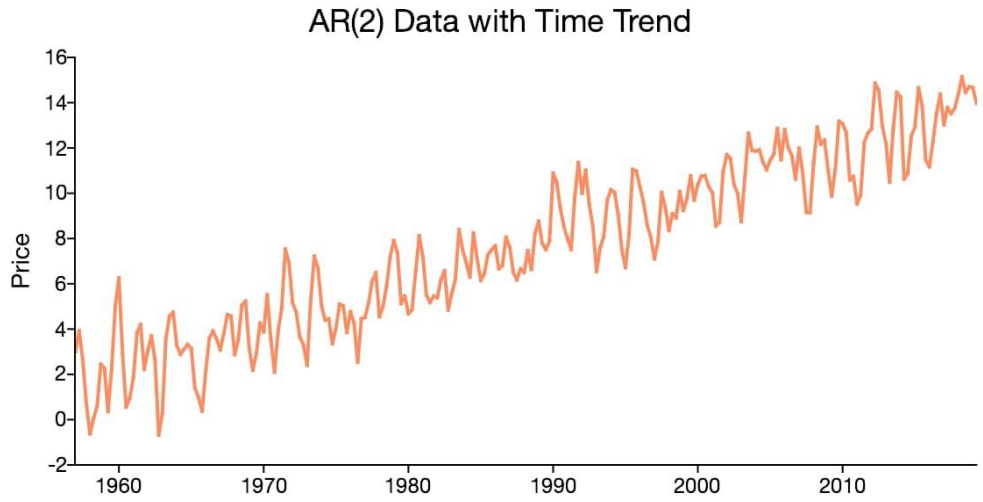
Time Series

- **univariate vs. multivariate** A time series containing records of a single variable is termed as univariate, but if records of more than one variable are considered then it is termed as multivariate.
- **linear vs. non-linear** A time series model is said to be linear or non-linear depending on whether the current value of the series is a linear or non-linear function of past observations.
- **discrete vs. continuous** In a continuous time series observations are measured at every instance of time, whereas a discrete time series contains observations measured at discrete points in time.

Component of a Time Series

In general, a time series is affected by four components, i.e**. trend, seasonal, cyclical and irregular** components.

Trend: The general tendency of a time series to increase, decrease or stagnate over a long period of time.

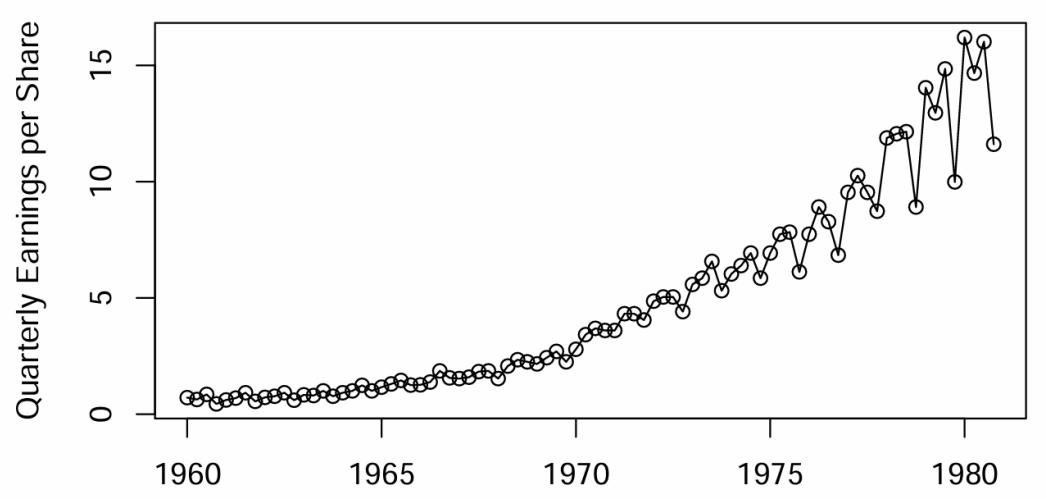


In addition to containing a non-zero mean, time series data may also have a deterministic component that is proportionate to the time period. When this occurs, the time series data is said to have a time trend.

Component of a Time Series

In general, a time series is affected by four components, i.e**. trend, seasonal, cyclical and irregular** components.

Seasonal : Seasonal variation This component explains flactuations within a year during the season, usually caused by climate and weather conditions, customs, traditional habits, etc.

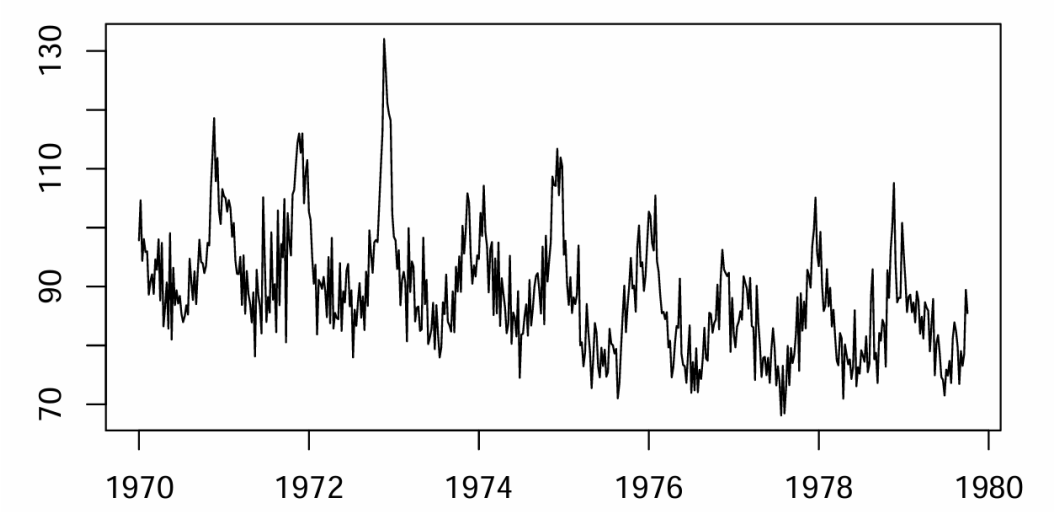


Johnson & Johnson quarterly earnings per share, 84 quarters, 1960-I to 1980-IV.

Component of a Time Series

In general, a time series is affected by four components, i.e**. trend, seasonal, cyclical and irregular** components.

Cyclical variation: This component describes the medium-term changes caused by circumstances, which repeat in cycles. The duration of a cycle extends over longer period of time.

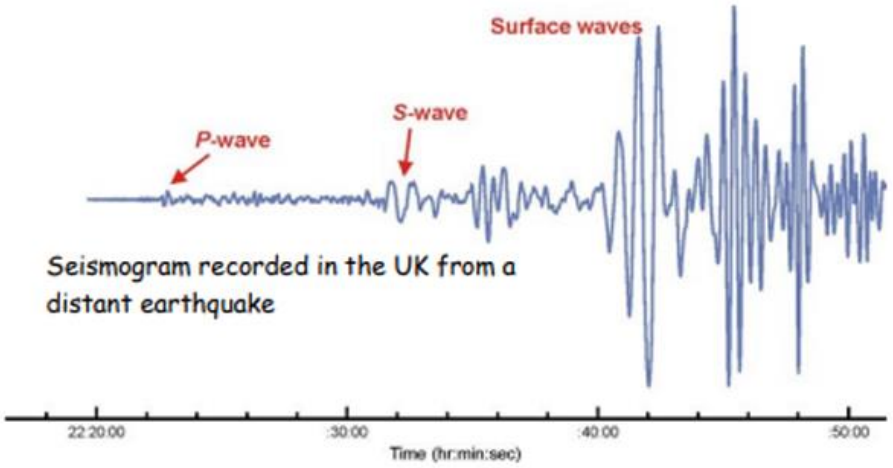


Average weekly cardiovascular mortality in Los Angeles County. There are 508 six-day smoothed averages obtained by ltering daily values over the 10 year period 1970-1979.

Component of a Time Series

In general, a time series is affected by four components, i.e**. trend, seasonal, cyclical and irregular** components.

Irregular variation: Irregular or random variations in a time series are caused by unpredictable influences, which are not regular and also do not repeat in a particular pattern. These variations are caused by incidences such as war, strike, earthquake, flood, revolution, etc. There is no defined statistical technique for measuring random fluctuations in a time series.



Earthquakes generate different types of seismic waves and these travel at different speeds through the Earth. P-waves are fastest and are the first signal to arrive on a seismogram, followed by the slower S-wave, then the surface waves. The arrival times of the P- and S-waves at different seismometers are used to determine the location of the earthquake. Assuming that we know the relative speed of P- and S-waves, the time difference between the arrivals of the P- and S-waves determines the distance the earthquake is from the seismometer.

Time Series

- **univariate vs. multivariate** A time series containing records of a single variable is termed as univariate, but if records of more than one variable are considered then it is termed as multivariate.
- **linear vs. non-linear** A time series model is said to be linear or non-linear depending on whether the current value of the series is a linear or non-linear function of past observations.
- **discrete vs. continuous** In a continuous time series observations are measured at every instance of time, whereas a discrete time series contains observations measured at discrete points in time.

Combination of Four Components

- Considering the effects of these four components, two different types of models are generally used for a time series.

  – Additive Model

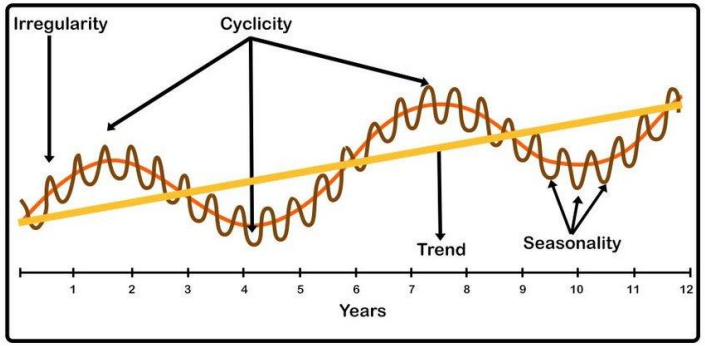
  $$Y(t) = T(t) + S(t) + C(t) + I(t)$$

  Assumption: These four components are independent of each other.

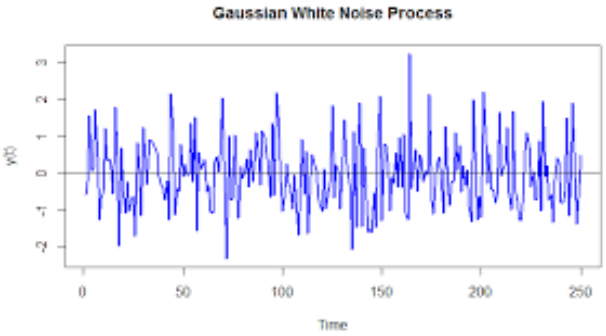
  – Multiplicative Model

  $$Y(t) = T(t) \times S(t) \times C(t) \times I(t)$$

  Assumption: These four components of a time series are not necessarily independent and they can affect one another.



Time Series Example: White Noise

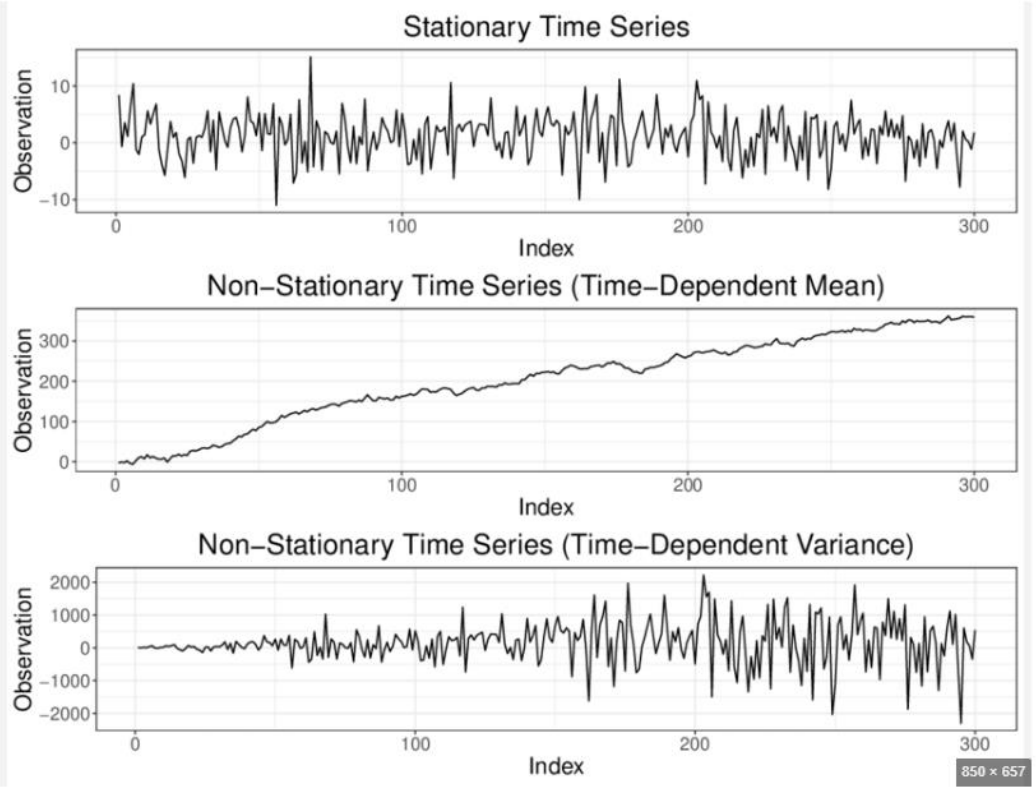


- **White Noise**
  - A simple time series could be a collection of uncorrelated random variables, $\{w_t\}$, with zero mean $\mu = 0$ and finite variance σ_w^2, denoted as $w_t \sim wn(0, \sigma_w^2)$.
- **Gaussian White Noise**
  - A particular useful white noise is Gaussian white noise, whereir. the w_t are independent normal random variables (with mean 0 and variance σ_w^2), denoted as $w_t \sim$ iid $\mathcal{N}(0, \sigma_w^2)$.
- White noise time series is of great interest because if the stochastic behavior of all time series could be explained in terms of the white noise model, then classical statistical methods would suffice.

Stationarity of Stochastic Process

- Forecasting is difficult as time series is non-deterministic in nature, i.e. we cannot predict with certainty what will occur in the future.
- But the problem could be a little bit easier if the time series is stationary: you simply predict its statistical properties will be the same in the future as they have been in the past!
  - A stationary time series is one whose statistical properties such as mean, variance, autocorrelation, etc. are all constant over time.
- Most statistical forecasting methods are based on the assumption that the time series can be rendered approximately stationary after mathematical transformations.

Stationarity of Stochastic Process



ARIMA Models

- ARIMA is an acronym that stands for **A**uto-**R**egressive **I**ntegrated **M**oving **A**verage. Specifically,
  - **AR** *Autoregression.* A model that uses the dependent relationship between an observation and some number of lagged observations.
  - **I** *Integrated.* The use of differencing of raw observations in order to make the time series stationary.
  - **MA** *Moving Average.* A model that uses the dependency between an observation and a residual error from a moving average model applied to lagged observations.
- Each of these components are explicitly specified in the model as a parameter.
- Note that **AR** and **MA** are two widely used linear models that work on stationary time series, and **I** is a preprocessing procedure to "stationarize" time series if needed.

ARIMA Models

- A standard notation is used of $ARIMA(p, d, q)$ where the parameters are substituted with integer values to quickly indicate the specific ARIMA model being used.
  - **p** The number of lag observations included in the model, also called the lag order.
  - **d** The number of times that the raw observations are differenced, also called the degree of differencing.
  - **q** The size of the moving average window, also called the order of moving average.
- A value of 0 can be used for a parameter, which indicates to not use that element of the model.
- In other words, ARIMA model can be configured to perform the function of an ARMA model, and even a simple AR, I, or MA model.

Parameter Estimation

- Note that p is like a hyperparameter for the $AR(p)$ process, thus fitting an $AR(p)$ model presumes p is known and only focusing on estimating coefficients, i.e. $\phi_1, \phi_2, \ldots, \phi_p$.
- There are many feasible approaches:
  - Method of moments estimator (e.g. Yule-Walker estimator)
  - Maximum Likelihood Estimation (MLE) estimator
  - Ordinary Least Squares (OLS) estimator
- If the observed series is short or the process is far from stationary, then substantial differences in the parameter estimations from various approaches are expected.

AutoRegressive (AR) model

In simple terms, an **AR model** predicts future values in a time series using **past values of the same series**.

- The general form of an AR model of order *p* (called AR(p)) is:

$$Y_t = c + \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \cdots + \phi_p Y_{t-p} + \varepsilon_t$$

Where:

- Y_t: Value at time t
- c: Constant (intercept)
- $\phi_1, \phi_2, \ldots, \phi_p$: Coefficients for previous time points
- ε_t: Random error (white noise)

Example: AutoRegressive (AR) model

Day	Temp (°C)
1	20
2	21
3	20.5
4	21.2
5	22
6	22.4
7	22.8

Imagine we are tracking the **daily temperature** in a town, and you notice that today's temperature is **usually close to yesterday's**.

- Let's say we have the following temperature data (in °C) for 7 days:
- The general form of an AR model of order *p* (called AR(p)) is:

We want to **predict Day 8's temperature** using an **AR(1)** model (only using the previous day's value).

Let's assume we fit the AR(1) model and get the following equation (the coefficients are just for illustration):

$$Y_t = 1.2 + 0.95 \cdot Y_{t-1}$$

This means:

If yesterday was 22.8°C (Day 7),

Then predicted temp for Day 8 = $1.2 + 0.95 \times 22.8 = 1.2 + 21.66 = 22.86°C$

Moving Average (MA) Model

- In time series modeling, an **MA(q)** model predicts the value of a series based on **past forecast errors** rather than past values.

General form of an MA(q) model:

$$Y_t = \mu + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q}$$

Where:

- Y_t: value at time t
- μ: mean of the series
- ε_t: white noise/error at time t
- $\theta_1, \theta_2, \ldots, \theta_q$: coefficients for past errors

Example: Moving Average (MA) Model

- Let's say You are tracking the **daily demand for ice cream**, and you suspect that **random shocks (like a sudden heatwave or a rainy day)** affect demand, but that effect doesn't last too long—just for a day or two.

we have 6 days of demand data:

Day	Demand (cones)
1	100
2	105
3	98
4	110
5	108
6	112

Let's assume you fit an **MA(1)** model (uses 1 lag of error), and you get:

Let's suppose:

- $\mu = 105$
- $\theta_1 = 0.5$

Now, if on Day 5, your model predicted 106 but the actual demand was 108, the error was:

$$\varepsilon_5 = 108 - 106 = 2$$

So to forecast Day 6:

$$Y_6 = 105 + \theta_1 \cdot \varepsilon_5 + \varepsilon_6$$

$$Y_6 = 105 + 0.5 \cdot 2 + \varepsilon_6 = 106 + \varepsilon_6$$

You can't know ε_6 in advance (it's a random error), but the idea is: **you use past errors** to make your forecast.

MA models are **good when your time series shows short-term shocks** (sudden changes) that fade quickly.

ARMA (AutoRegressive Moving Average) model

An ARMA(p, q) model combines two parts:

- AR (AutoRegressive): Uses past values of the series
- MA (Moving Average): Uses past forecast errors (shocks)

General form:

$$Y_t = c + \phi_1 Y_{t-1} + \cdots + \phi_p Y_{t-p} + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q}$$

Where:

- Y_t: value at time t
- c: constant
- ϕ_i: coefficients for past values (AR part)
- θ_j: coefficients for past errors (MA part)
- ε_t: white noise/error at time t

Example: ARMA

- Let's say you have data on daily sales of a popular snack over 10 days:

Day	Sales
1	200
2	205
3	210
4	220
5	215
6	218
7	225
8	230
9	232
10	228

Suppose:

- There's a **trend** (people are buying more over time), which is captured by the **AR part**.

- But some days are affected by **random shocks** (like a discount or bad weather), captured by the **MA part**.

Fit an ARMA(1,1) Model

Imagine we fit the following model:

$$Y_t = 2 + 0.8Y_{t-1} + \varepsilon_t + 0.4\varepsilon_{t-1}$$

Assume:

- On **Day 9**, the actual sales were 232.
- Your model predicted 230 → so error: $\varepsilon_9 = 232 - 230 = 2$
- Day 8's error was $\varepsilon_8 = 230 - 228 = 2$
- Day 9's actual sales $Y_9 = 232$

Example: ARMA

- Let's say you have data on daily sales of a popular snack over 10 days:

Day	Sales
1	200
2	205
3	210
4	220
5	215
6	218
7	225
8	230
9	232
10	228

Predicting Day 10:

$$Y_{10} = 2 + 0.8 \cdot 232 + 0.4 \cdot 2 + \varepsilon_{10}$$

$$Y_{10} = 2 + 185.6 + 0.8 + \varepsilon_{10} = 188.4 + \varepsilon_{10}$$

So your model predicts Day 10 sales around **188.4 + noise**, based on:

- How sales are trending (AR part)
- How recent errors behaved (MA part)

ARIMA Model

ARIMA stands for:

- **AR** = AutoRegressive → uses past values
- **I** = Integrated → uses differencing to make data stationary
- **MA** = Moving Average → uses past forecast errors

It's written as **ARIMA(p, d, q)** where:

- **p** = number of AR terms (lags of the series)
- **d** = number of times the series is differenced
- **q** = number of MA terms (lags of forecast errors)

Example: ARIMA

- Let’s say you're analyzing the **monthly number of airline passengers** for a small airline over a year (simplified):

Month	Passengers
Jan	112
Feb	118
Mar	130
Apr	140
May	145
Jun	150
Jul	170
Aug	165
Sep	160
Oct	155
Nov	150
Dec	148

You notice:

• The data is **increasing over time** (non-stationary).

• But once you take the **difference between months** (subtract current from previous), the trend flattens and becomes more stable.

Step1: Differencing (d = 1)

We compute:

$$Y_t' = Y_t - Y_{t-1}$$

Month	Passengers	Difference
Feb	118	6
Mar	130	12
Apr	140	10
…	…	…

Example: ARIMA

- Let’s say you're analyzing the **monthly number of airline passengers** for a small airline over a year (simplified):

Month	Passengers
Jan	112
Feb	118
Mar	130
Apr	140
May	145
Jun	150
Jul	170
Aug	165
Sep	160
Oct	155
Nov	150
Dec	148

Step2: Fit AR and MA parts

Let’s say we analyze the differenced series and find:

• A good fit is **AR(1)** → current value depends on previous one

• And **MA(1)** → current value depends on previous error

So our model is **ARIMA(1, 1, 1)**.

After differencing, the model is:

$$Y_t' = c + \phi_1 Y_{t-1}' + \theta_1 \varepsilon_{t-1} + \varepsilon_t$$

Then, to get the actual value:

$$Y_t = Y_{t-1} + Y_t'$$

Example: ARIMA

- Let's say you're analyzing the **monthly number of airline passengers** for a small airline over a year (simplified):

Month	Passengers
Jan	112
Feb	118
Mar	130
Apr	140
May	145
Jun	150
Jul	170
Aug	165
Sep	160
Oct	155
Nov	150
Dec	148

Let's say:

- The last observed value is 148 (in December),
- Last differenced value Y_t' is -2,
- Model coefficients are:
  - $c = 0.5, \phi_1 = 0.7, \theta_1 = 0.3$
- Last error = -1

Then the forecast for January (next month):

$$Y'_{t+1} = 0.5 + 0.7 \cdot (-2) + 0.3 \cdot (-1) = 0.5 - 1.4 - 0.3 = -1.2$$

$$Y_{t+1} = 148 + (-1.2) = 146.8$$

So, the model predicts **146.8 passengers** for January.

Example: ARIMA

- Let’s say you're analyzing the **monthly number of airline passengers** for a small airline over a year (simplified):

Month	Passengers
Jan	112
Feb	118
Mar	130
Apr	140
May	145
Jun	150
Jul	170
Aug	165
Sep	160
Oct	155
Nov	150
Dec	148

It combines:

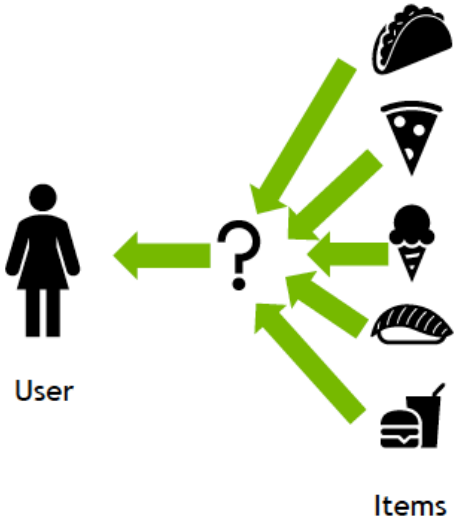
- AR: past values
- I: differencing for stationarity
- MA: past errors

• ARIMA is powerful and flexible for forecasting real-world data like sales, temperature, stock prices, etc.

Recommendation System

- **What is a recommendation system?**
  - A recommendation system suggests products, content, or services to users based on their preferences, behaviors, or interactions.
  - Examples: Netflix movie recommendations, Amazon product suggestions, Spotify playlists.

- **Why are recommendation systems important?**
  - Improve user experience.
  - Increase engagement and sales.
  - Help users discover new content/products.



Types of Recommendation Systems

Content-Based Filtering:

- Recommends items based on the features of items and user preferences.

Collaborative Filtering:

- Recommends items based on the past behavior of users. It assumes that if two users agreed in the past, they will agree in the future.

Hybrid Methods:

- Combine collaborative filtering with content-based filtering for better recommendations.

Collaborative Filtering

- **Definition**: Collaborative Filtering is a technique used by recommendation systems to predict the interests of a user by collecting preferences from many users.
- **Key Concept**: "People who agreed in the past will agree in the future."

How Collaborative Filtering Works?

Similarity Measurement: Measures the similarity between users (user-based) or items (item-based).

- Common similarity measures: Cosine similarity, Pearson correlation.

Building a Recommendation Model:

- **Step 1**: Create a user-item interaction matrix.
- **Step 2**: Compute similarities (between users or items).
- **Step 3**: Predict ratings based on similar users or items.
- **Step 4**: Recommend items that have the highest predicted ratings.

Example: Collaborative Filtering

Scenario: Suppose you have a movie recommendation system.

- Users rate movies on a scale from 1 to 5.
- The goal is to recommend movies to a user based on ratings of similar users.

User\Movie	Movie A	Movie B	Movie C	Movie D	Movie E
User 1	5	4	1	0	2
User 2	4	0	3	5	4
User 3	2	5	4	4	0
User 4	0	3	2	4	5
User 5	3	2	5	1	3

Example: Collaborative Filtering

- **Step 1**: Calculate similarity between users (e.g., using Cosine similarity).
- **Step 2**: Identify users who are most similar to a target user (e.g., User 1).
- **Step 3**: Predict ratings for unrated movies based on the ratings of similar users.
- **Step 4**: Recommend movies with the highest predicted ratings to the target user.

Example Calculation:

- For **User 1**, let's say **User 3** has a high similarity (cosine similarity = 0.9).
- Based on User 3's ratings, we predict User 1 might like **Movie D** and **Movie E**.

Predicted Recommendation:

- Movie D and Movie E could be recommended to User 1 since similar users (like User 3) have rated these highly.

Example: Collaborative Filtering

We simplify this with a hypothetical movie recommendation scenario:
Imagine you are using a movie app like Netflix.

User 1: Likes "Movie A" (5 stars), "Movie B" (4 stars), and dislikes "Movie C" (1 star).
User 2: Likes "Movie A" (4 stars) and dislikes "Movie B" (1 star), but likes "Movie C" (3 stars).
User 3: Likes "Movie B" (5 stars) and "Movie C" (4 stars), but dislikes "Movie A" (1 star).
Now, you want to recommend movies to **User 1**.

Collaborative Filtering steps:

- **Find Similar Users**: Compare User 1's ratings with others. It turns out that **User 2** has similar preferences (both like "Movie A").
- **Predict Ratings**: Since User 1 and User 2 have similar ratings for "Movie A," User 1 might also like **"Movie C"** based on User 2's positive rating for it.
- **Recommendation**: The app recommends **"Movie C"** to User 1.