Week 5
Artificial Intelligence & Machine Learning

Data Preprocessing in Machine learning

Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model.

When creating a machine learning project, it is not always a case that we come across the clean and formatted data. And while doing any operation with data, it is mandatory to clean it and put in a formatted way. So for this, we use data preprocessing task.

Why do we need Data Preprocessing?

A real-world data generally contains noises, missing values, and maybe in an unusable format which cannot be directly used for machine learning models. Data preprocessing is required tasks for cleaning the data and making it suitable for a machine learning model which also increases the accuracy and efficiency of a machine learning model.

Importance of data preprocessing

The need for data preprocessing is there because good data is undoubtedly more important than good models and for which the quality of the data is of paramount importance. Therefore, companies and individuals invest a lot of their time in cleaning and preparing the data for modeling. The data present in the real world contains a lot of quality issues, noise, inaccurate, and not complete. It may not contain relevant, specific attributes and could have missing values, even incorrect and spurious values. To improve the quality of the data preprocessing is essential. The preprocessing helps to make the data consistent by eliminating any duplicates, irregularities in the data, normalizing the data to compare, and improving the accuracy of the results. The machines understand the language of numbers, primarily binary numbers 1s and 0s. Nowadays, most of the generated and available data is

unstructured, meaning not in tabular form, nor having any fixed structure to the data. The most consumable form of unstructured data is text, which comes in the form of tweets, posts, comments. We also get data in the format of images, audio and as we can see, such data is not present in the format that can be readily ingested into a model. So, for the parsing, we need to convert or transform the data so that the machine can interpret it. Again to reiterate, data preprocessing is a crucial step in the Data Science process.

Data Cleaning

the real-world data is not all complete, accurate, correct, consistent, and relevant. The first and the primary step is to clean the data. There are various steps in this stage, it involves:

- Making the data consistent across the values, which can mean:
  - The attributes may have incorrect data types and are not in sync with the data dictionary. Correction of the data types is a must before proceeding with any type of data cleaning.
  - Replace the special characters for example: replace $ and comma signs in the column of Sales/Income/Profit i.e making $10,000 as 10000.
  - Making the format of the date column consistent with the format of the tool used for data analysis.
- Check for null or missing values, also check for the negative values. The relevancy of the negative values depends on the data. In the income column, a negative value is spurious though the same negative value in the profit column becomes a loss.
- Smoothing of the noise present in the data by identifying and treating for outliers.

Assess Data quality

There are many aspects of data quality and various dimensions that one can consider when evaluating the data at hand. Some of the most common aspects examined in the data quality assessment process are the following:

- **Number of missing values**. Most of the real-world datasets contain missing values, i.e., feature entries with no data value stored. As many machine learning algorithms do not support missing values, detecting the missing values and properly handling them, can have a significant impact.
- **Existence of duplicate values.** Duplicate values can take various formats, such as multiple entries of the same data point, multiple instances of an entire column, and repetition of the same value in an I.D. variable. While duplicate instances might be valid in some datasets, they often arise because of errors in the data extraction and integration processes. Hence, it is important to detect any duplicate values and decide if they correspond to invalid values (true duplicates) or form a valid part of the dataset.
- **Existence of outliers/anomalies.** Outliers are data points that differ substantially from the rest of data, and they may arise due to the diversity of the dataset or because of errors/mistakes. As machine learning algorithms are sensitive to the range and distribution of attribute values, identifying the outliers and their nature is important for assessing the quality of the dataset.
- **Existence of invalid/bad formatted values.** Datasets often contain inconsistent values, such as variables with different units across the data points and variables with incorrect data type. For example, it is often the case that some special numerical variables, such as percentages and fractions, are mistakenly stored as strings, and one should detect and transform such cases so that the machine learning algorithm can work with the actual numbers.

What is an anomaly?

Before talking about anomaly detection, we need to understand what an **anomaly** is.

Generally speaking, an anomaly is something that differs from a norm: a deviation, an exception. In software engineering, by anomaly we understand a rare occurrence or event that doesn't fit into the pattern, and, therefore, seems suspicious. Some examples are:

- sudden burst or decrease in activity;
- error in the text;
- sudden rapid drop or increase in temperature.

Common reasons for outliers are:

- data preprocessing errors;
- noise;
- fraud;
- attacks.

Normally, you want to catch them all; a software program must run smoothly and be predictable so every outlier is a potential threat to its robustness and security. Catching and identifying anomalies is what we call **anomaly or outlier detection**.
For example, if large sums of money are spent one after another within one day and it is not your typical behavior, a bank can block your card. They will see an unusual pattern in your daily transactions. This anomaly can typically be connected to fraud since identity thieves try to steal as much money as they can while they can. Once an anomaly is detected, it needs to be investigated, or problems may follow.

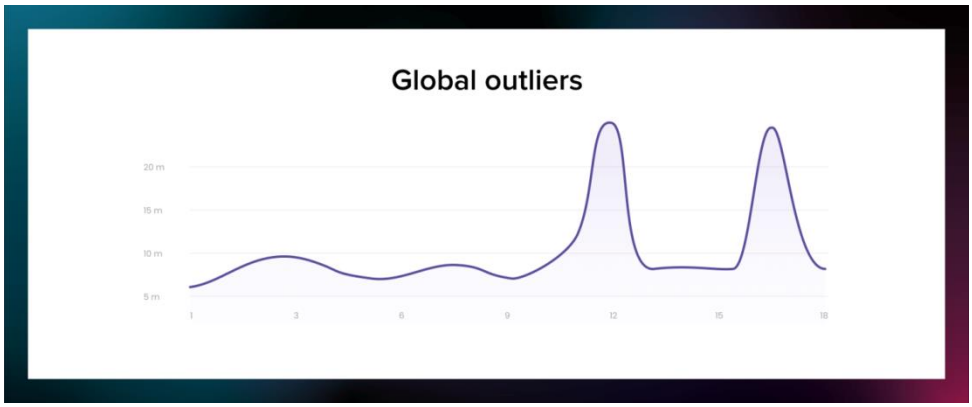
Types of anomalies

Now let's see what kinds of anomalies or outliers machine learning engineers usually have to face.

Global outliers

When a data point assumes a value that is far outside all the other data point value ranges in the dataset, it can be considered a global anomaly. In other words, it's a rare event.

For example, if you receive an average American salary to your bank accounts each month but one day get a million dollars, that would look like a global anomaly to the bank's analytics team.



Contextual outliers

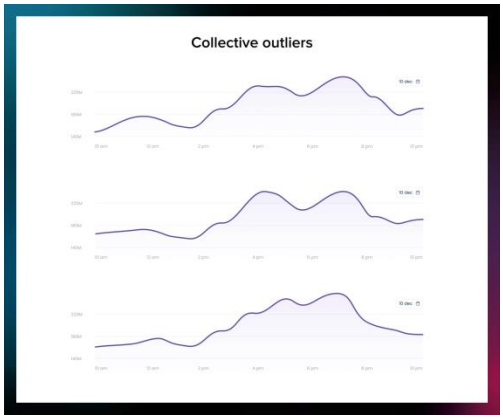
When an outlier is called contextual it means that its value doesn't correspond with what we expect to observe for a similar data point in the same context. Contexts are usually temporal, and the same situation observed at different times can be not an outlier.

For example, for stores it's quite normal to experience an increase in customers during the holiday season. However, if a sudden boost happens outside of holidays or sales, it can be considered a contextual outlier.

Collective outliers

Collective outliers are represented by a subset of data points that deviate from the normal behavior.

In general, tech companies tend to grow bigger and bigger. Some companies may decay but it’s not a general trend. However, if many companies at once show a decrease in revenue in the same period of time, we can identify a collective outlier.



Detect missing values with pandas dataframe functions: .info() and .isna()

Pandas DataFrame info() Method

Definition and Usage

The info() method prints information about the DataFrame.

The information contains the number of columns, column labels, column data types, memory usage, range index, and the number of cells in each column (non-null values).

Note: the info() method actually *prints* the info. You do not use the print() method to print the info.

Syntax

***dataframe*.info(verbose,buf,max_cols,memory_usage,show_counts,null_counts)**

Parameters

The parameters are keyword arguments.

Parameter	**Value**	**Description**
verbose	True|False	Optional. Whether to print all the information or not
buf	buffer	Optional. Outputs to buffer instead of to the sys.stdout. Useful when writing to files
max_cols	int	Optional. Specifies the number of columns you want information from
memory_usage	True|False|*string*	Optional. Specifies whether to print the memory usage or not, or 'deep' to do a real calclulation of the memory usage (at a high computer resource cost) instead of an estimate based on dtypes and number of rows (lower cost).
show_counts	True|False	Optional. Specifies whether to print the non-null counts or not

Return Value

None. The info() method does not return any value, it prints the information.

pandas.DataFrame.isna

DataFrame.isna()
Detect missing values.

Return a boolean same-sized object indicating if the values are NA. NA values, such as None or **numpy.NaN**, gets mapped to True values. Everything else gets mapped to False values. Characters such as empty strings '' or **numpy.inf** are not considered NA values (unless you set pandas.options.mode.use_inf_as_na = True).

Returns DataFrame
Mask of bool values for each element in DataFrame that indicates whether an element is an NA value.
OR
Pandas **dataframe.isna()** function is used to detect missing values. It return a boolean same-sized object indicating if the values are NA. NA values, such as None or numpy.NaN, gets mapped to True values. Everything else gets mapped to False values. Characters such as empty strings ” or numpy.inf are not considered NA values (unless you set pandas.options.mode.use_inf_as_na = True).
Syntax: *DataFrame.isna()*
Returns : *Mask of bool values for each element in DataFrame that indicates whether an element is not an NA value.*

The problem of missing value is quite common in many real-life datasets. Missing value can bias the results of the machine learning models and/or reduce the accuracy of the model. This article describes what is missing data, how it is represented, and the different reasons for the missing data. Along with the different categories of missing data, it also details out different ways of handling missing values with examples and how to handle missing values in dataset.

The following topics are covered in this guide:

What is a Missing Value?

Missing data is defined as the values or data that is not stored (or not present) for some variable/s in the given dataset.

Below is a sample of the missing data from the Titanic dataset. You can see the columns ‘Age’ and ‘Cabin’ have some missing values.

PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
1	0	3	male	22	1	0	A/5 21171	7.25		S
2	1	1	female	38	1	0	PC 17599	71.2833	C85	C
3	1	3	female	26	0	0	STON/O2. 3101282	7.925		S
4	1	1	female	35	1	0	113803	53.1	C123	S
5	0	3	male	35	0	0	373450	8.05		S
6	0	3	male		0	0	330877	8.4583		Q

Image 1

How is Missing Value Represented In The Dataset?

In the dataset, blank shows the missing values.

In Pandas, usually, missing values are represented by **NaN**.

It stands for **Not a Number**.

Out[3]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S
5	6	0	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583	NaN	Q
6	7	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.8625	E46	S
7	8	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	349909	21.0750	NaN	S
8	9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27.0	0	2	347742	11.1333	NaN	S
9	10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14.0	1	0	237736	30.0708	NaN	C

Image 2

The above image shows the first few records of the Titanic dataset extracted and displayed using Pandas.

Why Is Data Missing From The Dataset

There can be multiple reasons why certain values are missing from the data.

Reasons for the missing data from the dataset affect the approach of handling missing data. So it's necessary to understand why the data could be missing.

Some of the reasons are listed below:

- Past data might get corrupted due to improper maintenance.
- Observations are not recorded for certain fields due to some reasons. There might be a failure in recording the values due to human error.
- The user has not provided the values intentionally.

Types Of Missing Value

Formally the missing values are categorized as follows:

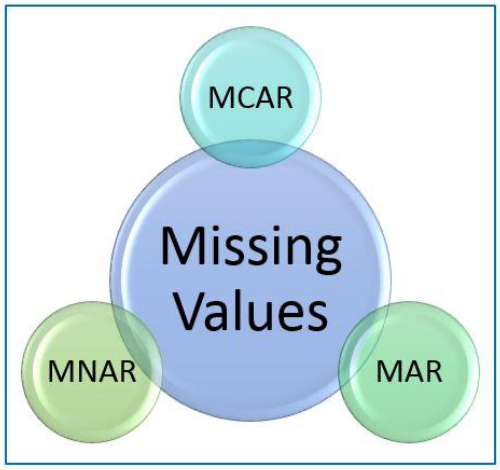


Figure 1 - Different Types of Missing Values in Datasets

Image 3

Missing Completely At Random (MCAR)

In MCAR, the probability of data being missing is the same for all the observations.

In this case, there is no relationship between the missing data and any other values observed or unobserved (the data which is not recorded) within the given dataset.

That is, missing values are completely independent of other data. There is no pattern.

In the case of MCAR, the data could be missing due to human error, some system/equipment failure, loss of sample, or some unsatisfactory technicalities while recording the values.

For Example, suppose in a library there are some overdue books. Some values of overdue books in the computer system are missing. The reason might be a human error like the librarian forgot to type in the values. So, the missing values of overdue books are not related to any other variable/data in the system.

It should not be assumed as it's a rare case. The advantage of such data is that the statistical analysis remains unbiased.

Missing At Random (MAR)

Missing at random (MAR) means that the reason for missing values can be explained by variables on which you have complete information as there is some relationship between the missing data and other values/data.

In this case, the data is not missing for all the observations. It is missing only within sub-samples of the data and there is some pattern in the missing values.

For example, if you check the survey data, you may find that all the people have answered their ‘Gender’ but ‘Age’ values are **mostly** missing for people who have answered their ‘Gender’ as ‘female’. (The reason being most of the females don’t want to reveal their age.)

So, the probability of data being missing depends only on the observed data.

In this case, the variables ‘Gender’ and ‘Age’ are related and the reason for missing values of the ‘Age’ variable can be explained by the ‘Gender’ variable but you can not predict the missing value itself.

Suppose a poll is taken for overdue books of a library. Gender and the number of overdue books are asked in the poll. Assume that most of the females answer the poll and men are less likely to answer. So why the data is missing can be explained by another factor that is gender.

In this case, the statistical analysis might result in bias.

Getting an unbiased estimate of the parameters can be done only by modeling the missing data.

Missing Not At Random (MNAR)

Missing values depend on the unobserved data.

If there is some structure/pattern in missing data and other observed data **can not explain** it, then it is Missing Not At Random (MNAR).

If the missing data does not fall under the MCAR or MAR then it can be categorized as MNAR.

It can happen due to the reluctance of people in providing the required information. A specific group of people may not answer some questions in a survey.

For example, suppose the name and the number of overdue books are asked in the poll for a library. So most of the people having no overdue books are likely to answer the poll. People having more overdue books are less likely to answer the poll.

So in this case, the missing value of the number of overdue books depends on the people who have more books overdue.

Another example, people having less income may refuse to share that information in a survey.

In the case of MNAR as well the statistical analysis might result in bias.

Approaches to deal with missing values

It is important to handle the missing values appropriately.

- Many machine learning algorithms fail if the dataset contains missing values. However, algorithms like K-nearest and Naive Bayes support data with missing values.
- You may end up building a biased machine learning model which will lead to incorrect results if the missing values are not handled properly.
- Missing data can lead to a lack of precision in the statistical analysis.

How To Handle Missing Value?

Checking for missing values

The first step in handling missing values is to look at the data carefully and find out all the missing values.

Figure Out How To Handle The Missing Data

Analyze each column with missing values carefully to understand the reasons behind the missing values as it is crucial to find out the strategy for handling the missing values.

There are 2 primary ways of handling missing values:

1. Deleting the Missing values
2. Imputing the Missing Values

Deleting the Missing value

Generally, this approach is not recommended. It is one of the quick and dirty techniques one can use to deal with missing values.

If the missing value is of the type Missing Not At Random (MNAR), then it should not be deleted.

If the missing value is of type Missing At Random (MAR) or Missing Completely At Random (MCAR) then it can be deleted.

The disadvantage of this method is one might end up deleting some useful data from the dataset.

There are 2 ways one can delete the missing values:

Deleting the entire row

If a row has many missing values then you can choose to drop the entire row.

If every row has some (column) value missing then you might end up deleting the whole data.

Deleting the entire column

If a certain column has many missing values then you can choose to drop the entire column.

Approaches to deal with missing values

- ***Keep the missing value as is***

- Missing data under 10% for an individual case or observation can generally be ignored, except when the missing data is a MAR or MNAR.
- The number of complete cases i.e. observation with no missing data must be sufficient for the selected analysis technique if the incomplete cases are not considered.

MAR: Missing At Random. This means that the missing values in any feature are dependent on the values of other features.

MNAR: Missing Not At Random. Missing not at random data is a more serious issue and in this case, it might be wise to check the data gathering process further and try to understand why the information is missing. For instance, if most of the people in a survey did not answer a certain question, why did they do that? Was the question unclear?

Deleting missing data

In my opinion, if the missing value percentage is above a certain threshold (say, 60%), it does not make much sense to try and impute them because it would likely influence our predictions due to the biased estimations. Deletion of the rows or columns with unknown values would be better suited. For illustrative purposes, suppose the data set looks like this (missing instances are denoted with the **NaN** notation):

id	col1	col2	col3	col4	col5
0	2.0	5.0	3.0	6.0	4.0
1	9.0	NaN	9.0	0.0	7.0
2	19.0	17.0	NaN	9.0	NaN

The Python pandas library allows us to drop the missing values based on the rows that contain them (i.e. drop rows that have at least one NaN value):

```
import pandas as pd

df = pd.read_csv('data.csv')
df.dropna(axis=0)
```

The output is as follows:

id	col1	col2	col3	col4	col5
0	2.0	5.0	3.0	6.0	4.0

Remove the attributes with missing values Dropping a variable

- If the data is MCAR or MAR and the number of missing values in a feature is very high, then that feature should be left out of the analysis. If missing data for a certain feature or sample is more than 5% then you probably should leave that feature or sample out.
- If the cases or observations have missing values for target variables(s), it is advisable to delete the dependent variable(s) to avoid any artificial increase in relationships with independent variables.

Case Deletion

In this method, cases which have missing values for one or more features are deleted. If the cases having missing values are small in number, it is better to drop them. Though this is an easy approach, it might lead to a significant decrease in the sample size. Also, the data may not always be missing completely at random. This may lead to biased estimation of parameters.

Estimate and impute missing values

Imputation is the process of substituting the missing data by some statistical methods. Imputation is useful in the sense that it preserves all

cases by replacing missing data with an estimated value based on other available information. But imputation methods should be used carefully as most of them introduce a large amount of bias and reduce variance in the dataset.

Imputation by Mean/Mode/Median

If the missing values in a column or feature are numerical, the values can be imputed by the mean of the complete cases of the variable. Mean can be replaced by median if the feature is suspected to have outliers. For a categorical feature, the missing values could be replaced by the mode of the column. The major drawback of this method is that it reduces the variance of the imputed variables. This method also reduces the correlation between the imputed variables and other variables because the imputed values are just estimates and will not be related to other values inherently.

What are outliers in the data?

Definition of outliers	An *outlier* is an observation that lies an abnormal distance from other values in a random sample from a population. In a sense, this definition leaves it up to the analyst (or a consensus process) to decide what will be considered abnormal. Before abnormal observations can be singled out, it is necessary to characterize normal observations.

Detecting outliers

As outliers are very different values—abnormally low or abnormally high—their presence can often skew the results of statistical analyses on the dataset. This could lead to less effective and less useful models.

But dealing with outliers often requires domain expertise, and none of the outlier detection techniques should be applied *without* understanding the data distribution and the use case.

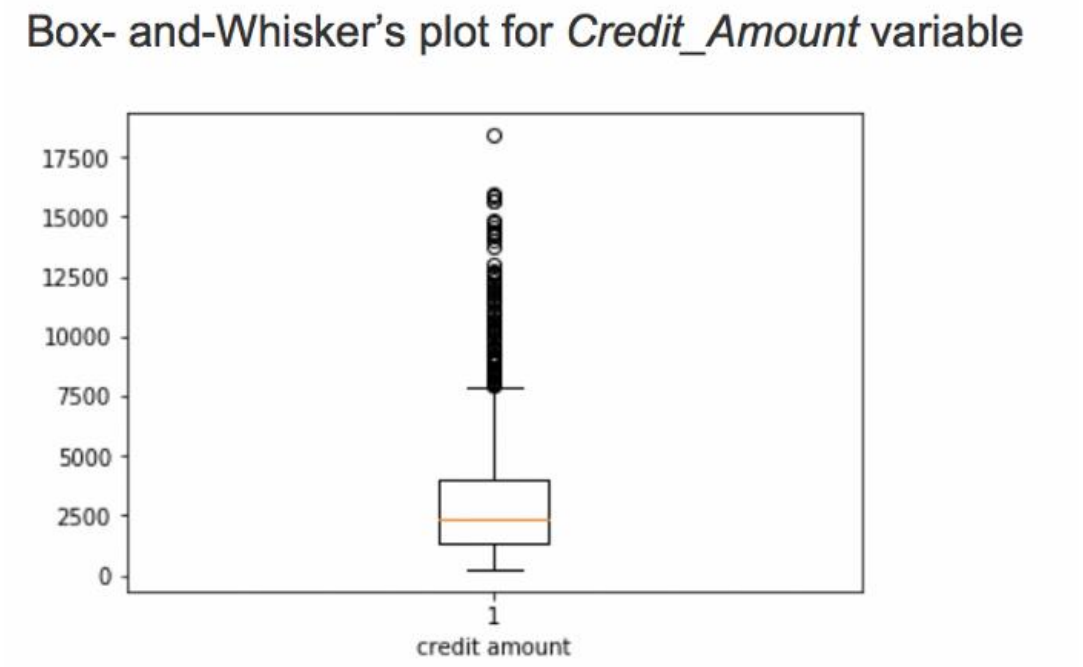
For example, in a dataset of house prices, if you find a *few* houses priced at around $1.5 million—much higher than the median house price, they're likely outliers. However, if the dataset contains a significantly large number of houses priced at $1 million and above—they may be indicative of an increasing trend in house prices. So it would be *incorrect* to label them all as outliers. In this case, you need some knowledge of the real estate domain.

Univariate Outlier Detection in Python

Univariate Outlier Detections Methods

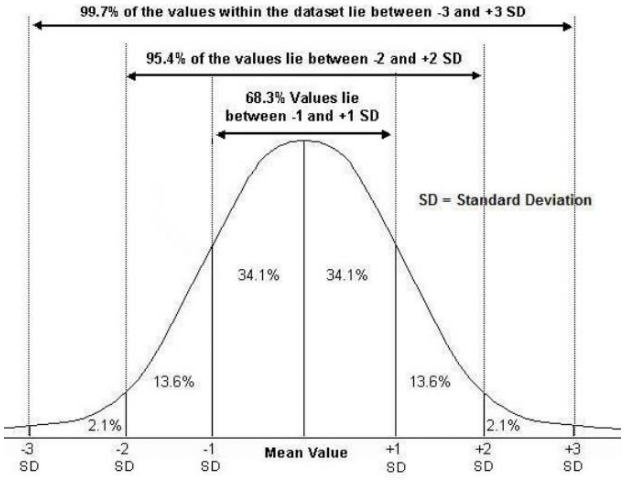
Box-and-Whisker's plot

Box-and-Whiskers plot uses quartiles to plot the shape of a variable. The interquartile range is the range between the first and the third quartiles (the edges of the box). Any data point that falls outside of either 1.5 times the IQR below the first quartile or 1.5 times the IQR above the third quartile is considered an outlier.



2. Using Standard Deviations Rule of Normal Distribution

A limitation of this method is that it can only be used when the data is 'not' strongly skewed. It requires the data to be close to normal.



As in the figure above, if data is normally distributed, 99.7% of the values of the data should lie between +/- 3 standard deviations and 95.4% of the values within +/-2 standard deviations of the data. So, we may consider any data points away from +/- 2 or 3 standard deviations as outliers.

This can be achieved by calculating the *standard scores or z-scores* of the data points. Wikipedia states a z-score as:

*The **standard score** or **z-score** is the signed number of standard deviations by which the value of an observation or data point is above the mean value of what is being observed or measured.*

We calculate the z-score by subtracting the mean of all the data values from a data point and dividing by standard deviation. Instead of using '*scipy's zscore function*', let's define a function for the same and then we add a threshold to bring out the indices of the outlier data points.

Time series outlier detection

Time series are everywhere! In user behavior on a website, or stock prices of a Fortune 500 company, or any other time-related example. Time series data is evident in every industry in some shape or form.

Naturally, it's also one of the most researched types of data. As a rule of thumb, you could say time series is a type of data that's sampled based on some kind of time-related dimension like years, months, or seconds.

Time series are observations that have been recorded in an orderly fashion and which are correlated in time.

While analyzing time series data, we have to make sure of the outliers, much as we do in static data. If you've worked with data in any capacity, you know how much pain outliers cause for an analyst. These outliers are called "anomalies" in time series jargon.

Anomaly detection techniques in time series data

There are few techniques that analysts can employ to identify different anomalies in data. It starts with a basic statistical decomposition and can work up to autoencoders. Let's start with the basic one, and understand how and why it's useful.

NoteSTL decomposition

STL stands for seasonal-trend decomposition procedure based on LOESS. This technique gives you the ability to split your time series signal into three parts: **seasonal, trend, and residue**.

It works for seasonal time-series, which is also the most popular type of time series data. To generate an STL-decomposition plot, we just use the ever-amazing *statsmodels* to do the heavy lifting for us.

Pros
It's simple, robust, it can handle a lot of different situations, and all anomalies can still be intuitively interpreted.
Cons
The biggest downside of this technique is rigid tweaking options. Apart from the threshold and maybe the confidence interval, there isn't much you can do about it. For example, you're tracking users on your website

that was closed to the public and then was suddenly opened. In this case, you should track anomalies that occur before and after launch periods separately.

Data Integration Overview

Data integration involves bringing together technical and business processes that combine data from disparate sources both traditional and non-traditional into meaningful datasets for business intelligence and analytics. Enterprises gain insights from this big data facilitating better and more timely decision-making that help ensure business competitiveness. However, one major challenge that may arise with the integration of traditional and non-traditional data sources is the vast amount of data generated in the governance ecosystem.

A complete data integration solution integrates data collected from multiple on-premises and cloud sources to support an efficient, business-ready data pipeline for DataOps. Data architecture defines the flow of data from such source systems and the workflow to capture, cleanse, normalize, and store the data for processing.

Techniques of data integration

A typical data integration architecture addresses data governance policies, security, data privacy, and data quality requirements along with helping enterprises consolidate data into a single, trusted view for analysis. The architecture establishes the working environment for the application of various techniques including

ETL (extract, transform, load)

Extracting, transforming, and loading data from multiple data sources to a single data store which is then loaded into a data warehouse. This technique of cleansing and preparing raw data in a staging area instead of a source system like a data warehouse or any other target system improves performance and mitigates the risk of data corruption.

ELT (extract, load, transform)

Extracting and loading raw data from multiple data source locations into a target data lake or cloud data warehouse, where the data can be transformed when required. This technique is ideal for supporting artificial intelligence, machine learning, and predictive analytics, among other applications where real-time data is used to capture intelligence and gain insights.

Data replication is another data integration technique. It delivers near-real-time data synchronization and distribution with low-impact, moving log-based data from one database to another database in order to synchronize and consolidate the information (for operational use and/or for backup). Data is replicated at regular intervals, in real-time or sporadically, depending on the requirements of the enterprise.

Data integration challenges

Common challenges that IT and data management teams encounter on data integration include keeping up with growing data volumes; unifying inconsistent data silos; dealing with the increasingly broad array of databases and other data platforms in IT infrastructures; integrating cloud and on-premises data; and resolving data quality issues. In large organizations with global operations, the number and distributed nature of the systems that need to be integrated add to the complexity.

The amount of data being generated and collected by organizations creates particularly big integration challenges. Data volumes continue to grow quickly, and the rate of that growth is only likely to increase as big data applications expand, the use of low-cost cloud object storage services rises and the IoT develops further. Data integration is essential to realizing the full potential business value of all that data, but

successfully planning and managing the required integration work is a complicated process.

To start with, data managers and data integration developers need full documentation of the source and target systems in an organization's data architecture so they can do the required mapping between them. They also must have a solid understanding of both internal and external data sources, the business rules that are embedded in the data, and how often data is updated and modified.

As a result, it's imperative that they work closely with business users. Data integration efforts should also be aligned with data governance programs, as well as related data quality and master data management (MDM) initiatives, to ensure that data is clean and consistent and that data lineage documentation is available to help integration developers better understand what's in data sets.

Approaches for Data Integration

Not only does AI greatly diminish the heavy lifting often associated with data integration efforts, AI and machine learning technology has shown to improve overall data integration project results such as

- **Data mapping** powered by AI can automate data transformation mapping by providing advanced features and agile data mapping predictions with machine learning algorithms. AI also enables users with less technical knowledge to embark on the data mapping exercise with simple drag and drop features, thus, reducing the time required to create data mappings.
- **The autonomous learning capability** of AI and machine learning enable enterprises to learn more about hidden patterns and trends from large datasets, so that accurate business intelligence and insights are derived from them through applying statistical models.

- **Data processing** with machine learning-powered data integration tools can parse big data generating precise data models that require less human intervention whereas traditional data integration tools require much longer setup and processing times to handle volumes of semi-structured or unstructured data formats.

  Data integrations infused with AI and machine learning solve complex data processing problems and improve integration flow propelling business forward, and providing business advantage across the enterprise. These new-age integration tools help enterprises gain insights from big data facilitating better and more timely decision-making that help ensure business competitiveness.

  Tight Coupling:
  - Here, a data warehouse is treated as an information retrieval component.
  - In this coupling, data is combined from different sources into a single physical location through the process of ETL – Extraction, Transformation, and Loading.

  Loose Coupling:
  - Here, an interface is provided that takes the query from the user, transforms it in a way the source database can understand, and then sends the query directly to the source databases to obtain the result.
  - And the data only remains in the actual source databases.

  Issues in Data Integration:

  There are three issues to consider during data integration: Schema Integration, Redundancy Detection, and resolution of data value conflicts. These are explained in brief below.

  1. Schema Integration:
  - Integrate metadata from different sources.
  - The real-world entities from multiple sources are referred to as the entity identification problem.

  2. Redundancy:
  - An attribute may be redundant if it can be derived or obtained from another attribute or set of attributes.

- Inconsistencies in attributes can also cause redundancies in the resulting data set.
- Some redundancies can be detected by correlation analysis.

3. Detection and resolution of data value **conflicts:**

- This is the third critical issue in data integration.
- Attribute values from different sources may differ for the same real-world entity.
- An attribute in one system may be recorded at a lower level of abstraction than the "same" attribute in another.

Data Transformation in Data Mining

The data are transformed in ways that are ideal for mining the data. The data transformation involves steps that are:

1. Smoothing:

It is a process that is used to remove noise from the dataset using some algorithms It allows for highlighting important features present in the dataset. It helps in predicting the patterns. When collecting data, it can be manipulated to eliminate or reduce any variance or any other noise form.

The concept behind data smoothing is that it will be able to identify simple changes to help predict different trends and patterns. This serves as a help to analysts or traders who need to look at a lot of data which can often be difficult to digest for finding patterns that they wouldn't see otherwise.

2. Aggregation:

Data collection or aggregation is the method of storing and presenting data in a summary format. The data may be obtained from multiple data sources to integrate these data sources into a data analysis description. This is a crucial step since the accuracy of data analysis insights is highly dependent on the quantity and quality of the data used. Gathering accurate data of high quality and a large enough quantity is necessary to produce relevant results.

The collection of data is useful for everything from decisions concerning financing or business strategy of the product, pricing, operations, and marketing strategies.

For example, Sales, data may be aggregated to compute monthly& annual total amounts.

3. Discretization:
It is a process of transforming continuous data into set of small intervals. Most Data Mining activities in the real world require continuous attributes. Yet many of the existing data mining frameworks are unable to handle these attributes.

Also, even if a data mining task can manage a continuous attribute, it can significantly improve its efficiency by replacing a constant quality attribute with its discrete values.

For example, (1-10, 11-20) (age:- young, middle age, senior).

4. Attribute Construction:
Where new attributes are created & applied to assist the mining process from the given set of attributes. This simplifies the original data & makes the mining more efficient.

5. Generalization:
It converts low-level data attributes to high-level data attributes using concept hierarchy. For Example Age initially in Numerical form (22, 25) is converted into categorical value (young, old).
For example, Categorical attributes, such as house addresses, may be generalized to higher-level definitions, such as town or country.

6. Normalization: Data normalization involves converting all data variable into a given range.
Techniques that are used for normalization are:

- **Min-Max Normalization:**
  - This transforms the original data linearly.

Z-Score Normalization:

- In z-score normalization (or zero-mean normalization) the values of an attribute (A), are normalized based on the mean of A and its standard deviation

- A value, v, of attribute A is normalized to v' by computing

Decimal Scaling:

- It normalizes the values of an attribute by changing the position of their decimal points
- The number of points by which the decimal point is moved can be determined by the absolute maximum value of attribute A.

Need for data transformation.

Businesses need to transform high volumes of data for several reasons, such as migrating data to the cloud, consolidating records, deleting duplicates, and changing formatting, etc.

Transformations are also applied to concatenate and validate data, perform lookups, or route data to different destinations. It is beneficial to have a data transformation tool with a wide array of transformation options to be able to manipulate data in the best possible way.

Let's look at a transformation example: suppose a bank acquires an insurance firm operating in the same region. Once the acquisition is complete, it is decided that a single payroll will be generated for all the employees. The payroll generation process would have been straightforward if all the employee data was stored in a unified system, such as a data warehouse or database.

What is data mining normalization?

The data normalization (also referred to as data pre-processing) is a basic element of data mining. It means transforming the data, namely converting the source data in to another format that allows processing data effectively. The main purpose of data normalization is to minimize or even exclude duplicated data. This is a very essential and important issue because it is increasingly problematic to keep in data in relational databases, which store identical data in more than one place.

The use of data mining normalization has a number of advantages:

- the application of data mining algorithms becomes easier
- the data mining algorithms get more effective and efficient
- the data is converted in to the format that everyone can get their heads around
- the data can be extracted from databases faster
- it is possible to analyze the data in a specific manner

What is data standardization?

Data standardization is the process of converting data to a common format to enable users to process and analyze it. Most organizations utilize data from a number of sources; this can include data warehouses, lakes, cloud storage, and databases. However, data from disparate sources can be problematic if it isn't uniform, leading to difficulties down the line (e.g., when you use that data to produce dashboards and visualizations, etc.).

Data standardization is crucial for many reasons. First of all, it helps you establish clear, consistently defined elements and attributes, providing a comprehensive catalog of your data. Whatever insights you're trying to get or problems you're attempting to solve, properly understanding your data is a crucial starting point.

Getting there involves converting that data into a uniform format, with logical and consistent definitions. These definitions will form your metadata — the labels that identify the what, how, why, who, when, and where of your data. That's the basis of your data standardization process.

From an accuracy perspective, standardizing the way you label data will improve access to the most relevant and current information. This will help make your analytics and reporting easier. Security-wise, mindful cataloging forms the basis of a powerful authentication and authorization approach, which will apply security restrictions to data items and data users as appropriate.

Data Reduction in Data Mining

Data mining is applied to the selected data in a large amount database. When data analysis and mining is done on a huge amount of data, then it takes a very long time to process, making it impractical and infeasible.

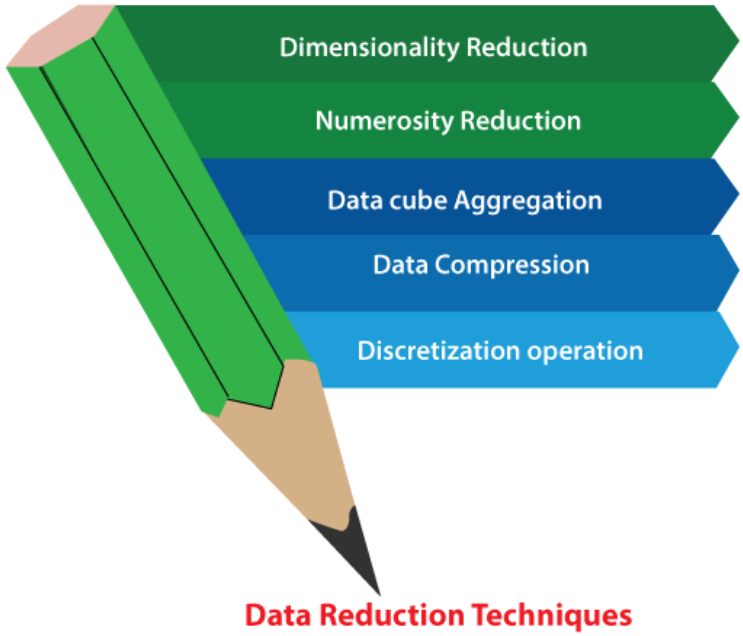
Data reduction techniques ensure the integrity of data while reducing the data. Data reduction is a process that reduces the volume of original data and represents it in a much smaller volume. Data reduction techniques are used to obtain a reduced representation of the dataset that is much smaller in volume by maintaining the integrity of the original data. By reducing the data, the efficiency of the data mining process is improved, which produces the same analytical results.

Data reduction does not affect the result obtained from data mining. That means the result obtained from data mining before and after data reduction is the same or almost the same.

Data reduction aims to define it more compactly. When the data size is smaller, it is simpler to apply sophisticated and computationally high-priced algorithms. The reduction of the data may be in terms of the number of rows (records) or terms of the number of columns (dimensions).

Techniques of Data Reduction

Here are the following techniques or methods of data reduction in data mining, such as:



1. Dimensionality Reduction

Whenever we encounter weakly important data, we use the attribute required for our analysis. Dimensionality reduction eliminates the attributes from the data set under consideration, thereby reducing the volume of original data. It reduces data size as it eliminates outdated or redundant features. Here are three methods of dimensionality reduction.

i. **Wavelet Transform:** In the wavelet transform, suppose a data vector A is transformed into a numerically different data vector A' such that both A and A' vectors are of the same length. Then how it is useful in reducing data because the data obtained from the wavelet transform can be truncated. The compressed data is obtained by retaining the smallest fragment of the strongest wavelet coefficients. Wavelet transform can be applied to data cubes, sparse data, or skewed data.

ii. **Principal Component Analysis:** Suppose we have a data set to be analyzed that has tuples with n attributes. The principal component analysis identifies k independent tuples with n attributes that can represent the data set.
In this way, the original data can be cast on a much smaller space,

and dimensionality reduction can be achieved. Principal component analysis can be applied to sparse and skewed data.

iii. **Attribute Subset Selection:** The large data set has many attributes, some of which are irrelevant to data mining or some are redundant. The core attribute subset selection reduces the data volume and dimensionality. The attribute subset selection reduces the volume of data by eliminating redundant and irrelevant attributes.

The attribute subset selection ensures that we get a good subset of original attributes even after eliminating the unwanted attributes. The resulting probability of data distribution is as close as possible to the original data distribution using all the attributes.

2. Numerosity Reduction

The numerosity reduction reduces the original data volume and represents it in a much smaller form. This technique includes two types parametric and non-parametric numerosity reduction.

i. **Parametric:** Parametric numerosity reduction incorporates storing only data parameters instead of the original data. One method of parametric numerosity reduction is the regression and log-linear method.

- **Regression and Log-Linear:** Linear regression models a relationship between the two attributes by modeling a linear equation to the data set. Suppose we need to model a linear function between two attributes.

  $$y = wx + b$$

  Here, y is the response attribute, and x is the predictor attribute. If we discuss in terms of data mining, attribute x and attribute y are the numeric database attributes, whereas w and b are regression coefficients.

  Multiple linear regressions let the response variable y model linear function between two or more predictor variables.

  Log-linear model discovers the relation between two or more

discrete attributes in the database. Suppose we have a set of tuples presented in n-dimensional space. Then the log-linear model is used to study the probability of each tuple in a multidimensional space.

Regression and log-linear methods can be used for sparse data and skewed data.

ii. **Non-Parametric:** A non-parametric numerosity reduction technique does not assume any model. The non-Parametric technique results in a more uniform reduction, irrespective of data size, but it may not achieve a high volume of data reduction like the parametric. There are at least four types of Non-Parametric data reduction techniques, Histogram, Clustering, Sampling, Data Cube Aggregation, and Data Compression.

- **Histogram:** A histogram is a graph that represents frequency distribution which describes how often a value appears in the data. Histogram uses the binning method to represent an attribute's data distribution. It uses a disjoint subset which we call bin or buckets.

  A histogram can represent a dense, sparse, uniform, or skewed data. Instead of only one attribute, the histogram can be implemented for multiple attributes. It can effectively represent up to five attributes.

- **Clustering:** Clustering techniques groups similar objects from the data so that the objects in a cluster are similar to each other, but they are dissimilar to objects in another cluster.

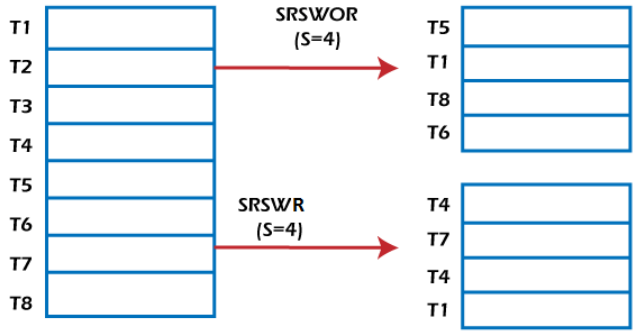
  How much similar are the objects inside a cluster can be calculated using a distance function. More is the similarity between the objects in a cluster closer they appear in the cluster.

  The quality of the cluster depends on the diameter of the cluster, i.e., the max distance between any two objects in the cluster.

  The cluster representation replaces the original data. This

technique is more effective if the present data can be classified into a distinct clustered.

- **Sampling:** One of the methods used for data reduction is sampling, as it can reduce the large data set into a much smaller data sample. Below we will discuss the different methods in which we can sample a large data set D containing N tuples:
    a. **Simple random sample without replacement (SRSWOR) of size s:** In this s, some tuples are drawn from N tuples such that in the data set D (s<N). The probability of drawing any tuple from the data set D is 1/N. This means all tuples have an equal probability of getting sampled.
    b. **Simple random sample with replacement (SRSWR) of size s:** It is similar to the SRSWOR, but the tuple is drawn from data set D, is recorded, and then replaced into the data set D so that it can be drawn again.



    c. **Cluster sample:** The tuples in data set D are clustered into M mutually disjoint subsets. The data reduction can be applied by implementing SRSWOR on these clusters. A simple random sample of size s could be generated from these clusters where s<M.
    d. **Stratified sample:** The large data set D is partitioned into mutually disjoint sets called 'strata'. A simple random sample is taken from each stratum to get stratified data. This method is effective for skewed data.

The distinction between data reduction and data redundancy

Data Fusion and Data Integration, we discussed and saw an example of the **data redundancy challenge**. While data redundancy and data reduction have very similar names and their terms use words that have connected meanings, the concepts are very different. Data redundancy is about having the same information presented under more than one attribute. As we saw, this can happen when we integrate data sources. However, data reduction is about reducing the size of data due to one of the following three reasons:

- **High-Dimensional Visualizations**: When we have to pack more than three to five dimensions into one visual, we will reach the human limitation of comprehension.
- **Computational Cost**: Datasets that are too large may require too much computation. This might be the case for algorithmic approaches.
- **Curse of Dimensionality**: Some of the statistical approaches become incapable of finding...

What is Data Redundancy?

Data redundancy means that, same piece of data is held in two different places. This includes two copies of data in a single database or two different locations in multiple software environments or platforms. Simply put, it is the repetition of data. Data redundancy is done for data backup and recovery purposes. Successful data redundancy requires attention to remote storage, networking and server hardware deployment needs which are based on your disaster recovery (DR) strategy.

- There are two types of data redundancy based on what's considered appropriate in database management and what's considered excessive. The two are:

Wasteful data redundancy:

- Wasteful data redundancy occurs when the data doesn't have to be repeated but it is duplicated due to inefficient coding or process complexity.

Positive Data redundancy:

- This type of data redundancy works to safeguard data and promote consistency. The key is to have a way to update all of the places where the data is redundant through one central access point.