Contents

[Topic 1 – Data integration 4](#_Toc95319789)

[Data Collection 4](#_Toc95319790)

[1.3 Data Collection 4](#_Toc95319791)

[1.4 Introduction to web scraping 5](#_Toc95319792)

[1.5 API 6](#_Toc95319793)

[1.6 A simple comparison of CRM and ERP 6](#_Toc95319794)

[Customer relationship management (CRM) 7](#_Toc95319795)

[Enterprise resource planning (ERP) 7](#_Toc95319796)

[1.7 What is a data warehouse? 8](#_Toc95319797)

[1.8 Star schema 9](#_Toc95319798)

[Snowflake Schema 9](#_Toc95319799)

[Topic 2 – Data Cleaning I 13](#_Toc95319800)

[Why pre-process data? 13](#_Toc95319801)

[2.2 Why pre-process data? 13](#_Toc95319802)

[Missing Value Imputation 14](#_Toc95319803)

[Censoring & Truncation 14](#_Toc95319804)

[Suspicious Data &Outliers 14](#_Toc95319805)

[Control Charts 14](#_Toc95319806)

[2.3 Missing Data 14](#_Toc95319807)

[2.4 Missing data mechanisms 15](#_Toc95319808)

[Missing Completely at Random (MCAR) 15](#_Toc95319809)

[Missing at Random (MAR) 15](#_Toc95319810)

[Missing Not at Random (MNAR) 16](#_Toc95319811)

[2.5 Imputing missing data 16](#_Toc95319812)

[2.6 Univariate approach to imputation 18](#_Toc95319813)

[2.7 Multivariate feature imputation 18](#_Toc95319814)

[2.8 Nearest neighbour imputation 19](#_Toc95319815)

[2.9 The hazards of imputation 19](#_Toc95319816)

[Quiz 19](#_Toc95319817)

[Rule of thumb 20](#_Toc95319818)

[Topic 3 – Data Cleaning II 21](#_Toc95319819)

[Data visualisation 21](#_Toc95319820)

[3.2 Data visualisation 21](#_Toc95319821)

[3.3 Univariate outlier detection 21](#_Toc95319822)

[3.4 Multivariate outlier detection 22](#_Toc95319823)

[3.5 Heteroscedasticity and linear regression 22](#_Toc95319824)

[3.6 Benford's Law 23](#_Toc95319825)

[Benford’s Law 23](#_Toc95319826)

[3.7 Noisy data 24](#_Toc95319827)

[3.8 Discretization techniques 26](#_Toc95319828)

[Simple Discretization Methods: Binning 26](#_Toc95319829)

[Binning Methods for Data Smoothing 27](#_Toc95319830)

[3.9 Moving average and exponential smoothing 28](#_Toc95319831)

[Topic 4 – Data reduction and transformation 32](#_Toc95319832)

[Data transformation: Normalization 32](#_Toc95319833)

[4.2 Data transformation: Normalization 32](#_Toc95319834)

[4.3 Data normalization in data mining 32](#_Toc95319835)

[Mix-Max normalization 33](#_Toc95319836)

[Z-score normalization 33](#_Toc95319837)

[Normalization by Decimal Scaling 33](#_Toc95319838)

[4.4 Data and numerosity reduction 34](#_Toc95319839)

[Numerosity reduction 34](#_Toc95319840)

[Dimensionality reduction 34](#_Toc95319841)

[4.5 Concept hierarchies 34](#_Toc95319842)

[4.6 Discretization continued 35](#_Toc95319843)

[Nominal Variables 35](#_Toc95319844)

[Ordinal Variables 36](#_Toc95319845)

[Continuous 36](#_Toc95319846)

[4.7 3-4-5 rule for attribute discretization 36](#_Toc95319847)

[4.8 Entropy 37](#_Toc95319848)

[Splitting an Attribute in Practice 38](#_Toc95319849)

[4.9 Dimensional reduction techniques 40](#_Toc95319850)

[What is dimensional reduction? 40](#_Toc95319851)

[Sampling 40](#_Toc95319852)

[Dimensionality Reduction 41](#_Toc95319853)

[Principal Component Analysis 42](#_Toc95319854)

Topic 1 – Data integration

# Topic 1 – Data integration

# Data Collection

## 1.3 Data Collection

In this step, I am going to discuss some concepts/ideas relating to data collection. I will then ask you to review a number of articles and answer a few questions. These questions are not part of the continuous assessment (CA), but will be really useful to drive a debate amongst yourselves.

Data collection has been continually used throughout history. Census data was extremely important to most imperial powers including the Chinese, British and Inca empires. The Egyptians even recorded their beer usage as far back as 3000 BC. We still conduct censuses today (in Ireland, we conduct one every five years, with the next one due in 2021). This data is very important for governments in order to help them determine strategies and national budgets. But private bodies also use this data to help them determine sampling strategies for marketing campaigns for example.

We all intuitively collect data when we see something. It might be someone’s face that we see, and we immediately make conclusions (rightly and wrongly) about this person. In this example, we are actually using a sample of information to make conclusions about the person in general. We also do this when we see groups of objects or people. We will collect some summary data and use this to make conclusions about the overall population of objects or subjects. Obviously, there is an inherent bias in simplified sampling strategies, and in 1895 the formal recognition of sampling techniques was born. This article by Jelke Bethlehem discusses the history of this process:

<https://drive.google.com/file/d/1Puz2TNUIFxS-5mGEo74wUt51Goj8mD5J/view>

In this paper, Bethlehem tells us about the use of the Central Limit Theorem. This theorem basically tells us that if we pick a random sample of 30 or more objects and repeat this process, the sample means will follow a normal distribution. The power this gives to an experimenter is vast and allows one to draw conclusions from a sample about a population. However, one must be careful to select the subjects correctly (this is no small task) in order to avoid bias and drawing the wrong conclusions. The data typically studied during these times predominantly originated from continuous variables. We have, however, advanced significantly in this regard. We now collect data from an enormous array of devices and in reality have a number of data types. As well as straight forward continuous numbers, we now have categories, text and images. They all in some way come back to numbers but the analyst will be faced with transforming and preprocessing these data sources in order to integrate them into their analysis. This article from Talend.com offers a nice view of how and why we need to do data integration:

<https://www.talend.com/resources/what-is-data-integration/>

As you may have guessed by now, data integration has developed into a science of its own as we have expanded our data requirements.

The introduction of IoT and vision technology has led us into collecting vast amounts of data. This is known as big data. This article from Forbes by Leetaru, tells us that sampling is still necessary as the cost of processing large batches of data is far greater than the storage cost:

<https://www.forbes.com/sites/kalevleetaru/2019/02/17/the-big-data-revolution-will-be-sampled-how-big-data-has-come-to-mean-small-sampled-data/#7a8ea8199ec6>

Leetaru states that the differential cost between storage and analysis has not really moved. You will also find that you will be faced with similar problems over the duration of this course. You can afford the storage but not the processing time. You will probably have to sample your data in order to build and process your models. Leetaru states that you need to be really careful with this strategy, but it can be useful. When faced with this problem in the past I have used a term called ‘bootstrapping’, which effectively selects samples of data, where each sample is processed and the results aggregated.

In my industry career, I have seen a number of projects make serious mistakes in the data collection process. In many of these cases, the investigator was unable to ask the question they wanted as the data collection process was flawed. It’s important to note that in medical research the data collection process is key when attempting to conduct drug trials and authorities put enormous emphasis on this during clinical trial submissions. You may find this in future projects, where the assessor or client is highly concerned with the data. This makes sense as we have already said, because wrongly collected data can bias your results and lead to wrong conclusions/forecasts. Some people believe that if they have petabytes of data they can predict anything. Not True. In fact, a correctly selected, small sample can have much better results than an ill-conceived, large sample.

Before, we finish up I would like to draw your attention to DCU’s ethics policy regarding data collection. It is worth noting that you need to be really careful here:

<https://www.dcu.ie/researchsupport/rec_guidance.shtml>

Remember, The General Data Protection Regulation (GDPR) is now an active requirement. It requires anyone who stores personal data to put the correct conditions in place to ensure this information is secure and is used in an ethical fashion. One must be extremely careful with images, people’s identities and any potential data that could identify someone. This could be a piece of data that identifies someone who drives a particular brand of car on a specific street. You may not give their name or house number, but if this car is unique to this street then the individual can be identified.

Finally, I would like to add one last statement: you will learn nothing if the data is incorrectly collected. Consider the following questions:

Can you think of ways that data can be incorrectly collected? Try and post two or three in the comments section below.

How much planning should you put into your data collection?

Does the question you are trying to answer come before the data collection or is it the other way round? What should it be and what actually happens in practice?

These are open questions and one could argue them either way. I have opinions on these. Once you have posted some answers in the comments section below I will share my thoughts with you.

## 1.4 Introduction to web scraping

You will now be provided with an overview of data scraping.

You will now be provided with an overview of data scraping.

In the following article, “strata scratch” have outlined some of the principles and techniques used to extract data from the web using Python “requests”:

<https://realpython.com/python-requests/>

and “beautiful soup”:

<https://pypi.org/project/beautifulsoup4/>

This is typically known as web scraping and is an incredibly powerful tool. There are other tools such as “urllib2”:

<https://docs.python.org/2/library/urllib2.html>

and “Selenium”:

<https://selenium.dev/about/>

that are also very useful, but for the purposes of this step we will focus mainly on “requests” and “beautiful soup”. “requests” helps us to extract data from a website in Python using get calls and “beautiful soup” is predominantly used to “prettify” our HTML data. Web data is full of extra text that can make it hard to extract the “real” data from the web page. When I discovered how to do this it opened up a whole new world for me. When you master these ideas the data sources that are available will expand exponentially and you will have entered the world of Big Data.

Go to the following Google Colab to see how you fare: Web scraping guide.

<https://drive.google.com/open?id=1ouoaHAAJl9jZLlZyd3qSrEi8XW7iqH7K>

## 1.5 API

API stands for **Application Programming Interfaces**.

The advent of these interfaces into government or company databases has resulted in considerable advancements in data accessibility for the common analyst. As an individual interested in the food sector, I have managed to get enormous access to food movements through the use of API’s. In fact, they have made data harvesting considerably easier. They can also be used as programming interfaces. For example, you can use the Google rest API to interface your code with Google Drive:

<https://developers.google.com/fit/rest/v1/get-started>

This allows you to expand the functionality of your programs greatly. The best way to think of them is that they give your little PC the power of the cloud to complete some action. The great thing is you don’t have to use the same programming language as the host machine. In fact, every time you use your browser you are actually using an API. The following example pulls data from the USDA (United States Department of Agriculture). You should go to this link to obtain a key:

[TO DO]

<https://fdc.nal.usda.gov/api-key-signup.html>

Go to the following Google Colab and see how you fare: Using API:

<https://drive.google.com/open?id=1bzZjKYvkNycI8ogf6bFPyU-4IkOczJMW>

## 1.6 A simple comparison of CRM and ERP

In this step, an overview of CRM and ERP is provided.

Customer relationship management (CRM):

<https://en.wikipedia.org/wiki/Customer_relationship_management>

and enterprise resource planning (ERP) programs:

<https://en.wikipedia.org/wiki/Enterprise_resource_planning>

are incredibly useful when attempting to extract data on business performance. Go to the following link and study the content:

<https://simply-crm.com/blog/difference-between-erp-and-crm/>

At a high level:

CRM 🡺 Customer data used by the sales and customer service departments

ERP 🡺 Primarily for financial data and the finance department

**ERPs focus on internal processes**, while **CRMs cover the external relationships between the company and its clients**.

### Customer relationship management (CRM)

CRM is a process in which a business or other organization administers its interactions with customers, typically using data analysis to study large amounts of information. CRM systems compile data from a range of different communication channels, including a company's website, telephone, email, live chat, marketing materials and more recently, social media. They allow businesses to learn more about their target audiences and how to best cater for their needs, thus retaining customers and driving sales growth.

Example CRM uses:

* Manage your contacts
* Handle the sales process
* Track data to acquire valuable insights into customer behaviour
* Generally operate on a smaller dataset than ERMs.

**Examples packages:** Pipedrive, HubSpot, Salesforce, Zoho.

### Enterprise resource planning (ERP)

ERM is the integrated management of main business processes, often in real time and mediated by software and technology. ERP is usually referred to as a category of business management software—typically a suite of integrated applications—that an organization can use to collect, store, manage, and interpret data from many business activities. ERP Systems can be local based or Cloud-based. Cloud-based applications have grown in recent years due to information being readily available from any location with internet access.

**Example ERM uses:**

* Accounting
* Manufacturing
* Inventory
* Service/product distribution
* Logistics
* Supply chain management
* Generally more suite to bigger companies and datasets.

**Example ERM packages:**

SAP, Sage, Oracle ERP, Microsoft Dynamics 365.

## 1.7 What is a data warehouse?

Data warehouses are generally constructed to store vast amounts of integrated organisational data and are used to help these organisations create reports.

They are usually derived from production data. This article outlines the use of a data warehouse and why we build them:

<https://www.computing.dcu.ie/~amccarren/mcm_papers/what-is-data-warehouse-2277-1891.1000117.pdf>

**Definitions:**

* Data warehouse software is a program that extracts, transforms, and loads (ETL) data from a variety of different sources and puts it into the same format, so companies can easily analyze it. ... The main goal of organizations using data warehouse software is to support analytics and make summary data easier to access.
* A data warehouse is a type of data management system that is designed to enable and support business intelligence (BI) activities, especially analytics. Data warehouses are solely intended to perform queries and analysis and often contain large amounts of historical data.
* Also known as enterprise data warehousing, data warehousing is an electronic method of organizing, analyzing, and reporting information. ... For example, data warehousing makes data mining possible, which assists businesses in looking for data patterns that can lead to higher sales and profits.
* Data warehousing is a process used to collect and manage data from multiple sources into a centralized repository to drive actionable business insights. With all your data in one place, it becomes simpler to perform analysis and reporting at different aggregate levels.
* It is the core of the BI system and helps you make better business decisions. In simple words, it is the electronic storage space for all your business data integrated from different marketing and other sources.
* A transactional (OLTP) database doesn’t lend itself to analytics. To effectively perform analytics, you need a data warehouse. A data warehouse is a database of a different kind: an OLAP (online analytical processing) database. A data warehouse exists as a layer on top of another database or databases (usually OLTP databases). The data warehouse takes the data from all these databases and creates a layer optimized for and dedicated to analytics

**Essentially, it consists of three tiers:**

1. The bottom tier is the database of the warehouse, where the cleansed and transformed data is loaded.
2. The middle tier is the application layer giving an abstracted view of the database. ...
3. The top-tier is where the user accesses and interacts with the data.

Consider the following questions:

Why would organisations need a data warehouse?

Have you used one before and what were the benefits over just storing the data in a single database?

## 1.8 Star schema

The following article was posted by Margaret Rouse on WhatIs.com:

<https://searchdatamanagement.techtarget.com/definition/star-schema>

* In data warehousing and business intelligence (BI), a star schema is the simplest form of a dimensional model, in which data is organized into

1. facts
2. dimensions.

* A **fact** is an **event** that is counted or measured, such as a sale or login.
* A **dimension** contains **reference information** about the fact, such as date, product, or customer.
* A star schema is diagrammed by surrounding each fact with its associated dimensions. The resulting diagram resembles a star.

A fact table sits at the center of a star schema database, and each star schema database only has a single fact table. The fact table contains the specific measurable (or quantifiable) primary data to be analyzed, such as sales records, logged performance data or financial data. It may be transactional -- in that rows are added as events happen -- or it may be a snapshot of historical data up to a point in time.

### Snowflake Schema

A snowflake schema database is similar to a star schema in that it has a single fact table and many dimension tables. However, for a snowflake schema, each dimension table might have foreign keys that relate to other dimension tables.

So, in a star schema there is no further branching from each dimension table. But in a snowflake schema each branch might have further branches -- like a snowflake with each branch having successively smaller branches coming out of a central core in a fractal pattern.

Star schemas are optimized for querying large data sets and are used in data warehouses and data marts to support online analytical processing cubes, business intelligence and analytic applications, and ad hoc queries.

Within the data warehouse or data mart, a dimension table is associated with a fact table by using a foreign key relationship. The dimension table has a single primary key that uniquely identifies each member record (row). The fact table contains the primary key of each associated dimension table as a foreign key. Combined, these foreign keys form a multi-part composite primary key that uniquely identifies each member record in the fact table. The fact table also contains one or more numeric measures.

For example, a simple Sales fact with millions of individual clothing sale records might contain a Product Key, Promotion Key, Customer Key, and Date Key, along with Units Sold and Revenue measures. The Product dimension would hold reference information such as product name, description, size, and color. The Promotion dimension would hold information such as promotion name and price. The Customer dimension would hold information such as first and last name, birth date, gender, address, etc. The Date dimension would include calendar date, week of year, month, quarter, year, etc. This simple Sales fact will easily support queries such as “total revenue for all clothing products sold during the first quarter of the 2010” or “count of female customers who purchased 5 or more dresses in December 2009”.

The star schema supports rapid aggregations (such as count, sum, and average) of many fact records, and these aggregations can be easily filtered and grouped (“sliced & diced”) by the dimensions. A star schema may be partially normalized (snowflaked), with related information stored in multiple related dimension tables, to support specific data warehousing needs.

Online analytical processing databases (data warehouses and data marts) use a denormalized star schema, with different but related information stored in one dimension table, to optimize queries against large data sets. A star schema may be partially normalized, with related information stored in multiple related dimension tables, to support specific data warehousing needs. In contrast, an online transaction processing (OLTP) database uses a normalized schema, with different but related information stored in separate, related tables to ensure transaction integrity and optimize processing of individual transactions.”

Having read the article above, consider the following question:

Can you outline which tables are facts and which are dimensions?

Diagram

Description automatically generated

A picture containing timeline

Description automatically generated

Graphical user interface, application, funnel chart

Description automatically generated

Graphical user interface

Description automatically generated with low confidence

Topic 2 – Data Cleaning I

# Topic 2 – Data Cleaning I

# Why pre-process data?

## 2.2 Why pre-process data?

This step illustrates the reasons why we pre-process data.

Having collected your data from various streaming or Internet of Things (IoT) devices, you will probably want to start analysing or using it to make predictions. You may have collected different types of data and they can come from structured or unstructured sources.

**Structured data is:**

* Loadable into “spreadsheets”
* Arranged into rows and columns
* Each filled or could be filled
* Data mining friendly

On the other hand, **unstructured data will come from**:

Word, HTML, PDF documents or PPTs

* Non-structured cells
* Variable record lengths such as notes free form survey-answers
* Text is relatively sparse, inconsistent and not uniform
* Images, video, music etc…

It is worth spending some time understanding your data and effectively pre-processing/cleaning it in order to give yourself the best chance of creating a useful model. As I said in the previous topic, if you put rubbish in you will get rubbish out.

You will be faced with the following challenges:

* How do I treat missing values in my feature set?
* How do I detect outliers and do I delete them from my dataset?
* What do I do with noisy data and will it impact my results?
* Can I reduce the volume of my data and still get comparable results?

These are all typical questions that we are faced with when we have collected our data, and in order to address them, we will have to use some pre-processing techniques. In the next three topics, we will outline techniques that can be used to address the questions outlined above.

Follow this link to the Google Colab for this step.

<https://drive.google.com/open?id=1aIHQUZWzNxSFNMcIPmHbcBd5UA_7VENJ>

### Missing Value Imputation

#### Standalone Imputation

It involves imputing the missing values with mean, median or mode. It is convenient and easy to implement, but introduces bias. Assume that the distribution of missing values is the same as non-missing values.

#### Regression Imputation

Better imputation than standalone imputation. It makes use of attribute relationships. Assume : all prior attributes are populated That is, monotonicity in missing values. Use linear regression, sweep left-to-right across the columns by row to predict the missing values.

#### Propensity Score

Within each group, estimate missing Xjs from known Xj’s using approximate Bayesian bootstrap.

### Censoring & Truncation

* **Censored** 🡺 Measurement is bounded but not precise e.g. Call duration > 20 are recorded as 20
* **Truncated** 🡺Data point dropped if it exceeds or falls below a certain bound e.g. customers with less than 2 minutes of calling per month

### Suspicious Data &Outliers

Watch out for outliers! They are, "departure from the expected". They are the datapoints which lie outside a specified pattern of distribution. Often, they are data model or glitches and are potentially legitimate.

### Control Charts

The control chart is a graph used to study how a process changes over time. Data are plotted in time order. This versatile data collection and analysis tool can be used by a variety of industries and is considered one of the seven basic quality tools. Control charts for variable data are used in pairs.

They are typically univariate: X-Bar, R, CUSUM.

Main steps (based on statistical inference):

1. Define “expected” and “departure” e.g. Mean and standard error based on sampling distribution of sample mean (aggregate);
2. Compute aggregate each sample
3. Plot aggregates vs expected and error bounds
4. “Out of Control” if aggregates fall outside bounds

## 2.3 Missing Data

Missing data can occur for a number of reasons such as; operator error, a faulty device or a respondent refusing to enter their data.

Missing data can be value attributes, entire records or entire sections. Typically, in any data analysis project, we will find that there are missing values in our data. Many people believe it is OK to just insert the average value of the non-missing values as their estimate for the missing values.

In the comments section below, discuss amongst yourselves why this is not such a good idea.

Frankly, this is the one thing that every newcomer to data analytics does. My suspicion is you will do the same. Let’s look at some data. In it, we have a small group of subjects who had their weight measured before and after a prescribed diet. Now you will see in the code in the Google Colab for this step that the average weight for both groups increases at the end of the trial.

Follow this link to go to the Google Colab for this step.

<https://drive.google.com/open?id=1TFnjHwYRRcVefCriUNw_oz1VHg8Url64>

## 2.4 Missing data mechanisms

When we find missing data values in our data, they are usually caused by one of three mechanisms outlined by Rubin (1976).

<https://www.jstor.org/stable/2335739?seq=1#metadata_info_tab_contents>

It is very important that you understand them as they will have a major influence on how you treat your analysis. It will also determine if it is appropriate to impute missing values. The following article gives a nice explanation of the area:

<https://www.jstor.org/stable/2335739?seq=1#metadata_info_tab_contents>

The three mechanisms are as follows:

1. Missing completely at random (MCAR)
2. Missing at Random (MAR)
3. Missing not at Random (MNAR)

Each of these mechanisms will now be discussed in turn.

### Missing Completely at Random (MCAR)

Missing Completely at Random is pretty straightforward. What it means is what it says: the propensity for a data point to be missing is completely random. There’s no relationship between whether a data point is missing and any values in the data set, missing or observed. The missing data are just a random subset of the data.

### Missing at Random (MAR)

This is where the unfortunate names come in.

Missing at Random means the propensity for a data point to be missing is not related to the missing data, but it is related to some of the observed data. Whether or not someone answered #13 on your survey has nothing to do with the missing values, but it does have to do with the values of some other variable.

We mean that the missingness is nothing to do with the person being studied. For example:

* a questionnaire might be lost in the post
* a blood sample might be damaged in the lab

**A better name would actually be Missing Conditionally at Random because the missingness is conditional on another variable**. But that’s not what Rubin originally picked, and it would really mess up the acronyms at this point.

The idea is, if we can control for this conditional variable, we can get a random subset. There is another alternative to this case and it exists where questions that have not been asked could determine if people answer the question in hand.

You can imagine that good techniques for data that are missing at random need to incorporate variables that are related to the missingness.

### Missing Not at Random (MNAR)

Data are missing not at random (MNAR) when the missing values on a variable are related to the values of that variable itself, even after controlling for other variables.

**Example:** data are missing on IQ and only the people with low IQ values have missing observations for this variable.

A problem with the MNAR mechanism is that it is impossible to verify that scores are MNAR without knowing the missing values.

Can you think of examples of each mechanism?

If I have a variable that is not missing at random can I impute the missing values?

Now I would like you to download the data from this link:

<https://drive.google.com/open?id=1Rp5Sqru6riXhceZoVcw17_TegO5P3Glss6Y8AUlaPm4>

Once you have done this, I would like you to analyse the data and record which variables have missing values. The following code in the Google Colab for this step should get you going:

<https://drive.google.com/open?id=1Id79rM0HSZmedfZta7LXFjCmxuLK1r1R>

Can you tell what type of missing process is occurring for the final exam mark?

Reference:

Rubin, D.B., 1976. Inference and missing data. Biometrika, 63(3), pp.581-592. Article available here:

<https://drive.google.com/open?id=1Id79rM0HSZmedfZta7LXFjCmxuLK1r1R>

## 2.5 Imputing missing data

As we have said previously, many real-world datasets have missing values. For many analysts, the natural thing to do with these missing values is to use some imputation technique.

The appropriateness of using imputation is dependent on the individual problem. Imputation techniques can be used to avoid losing important information that would be lost if one value was missing from a large row. This makes sense as datasets can be drastically reduced if every row with one missing value is removed from the analysis. Most statistical and machine learning packages will remove a row if one value is missing and generally you won’t know. The problem with imputation is as follows:

* When do I use it?
* What methods do I use?

There are a number of packages such as Amelia in R, statsmodels.imputation.mice.MICEData or sklearn.impute in python that will complete the imputation for you. Also, there is a substantial number of techniques ranging from replacing missing values with means of that variable to data simulators or propensity score modelling. The technique you use in your imputation is very much dependent on the cause of your missing values. My view is that you should not impute in the following situations:

When the variable in question is NMAR (unless you have evidence that this is a censored variable or you have apriori evidence on the distribution of the variable in question).

1. When there are missing values in the outcome variable.
2. The reason for both these situations is that I generally follow the guiding principle that your imputation technique should not bias your results. I use missing value imputation as little as possible.

I will regularly start any analysis I do by modelling variables with missing values against a range of other variables in my dataset. So I code a new variable as Missing (Y/N) from the variable with a large number of missing values. I will then conduct a simple logistic regression on the remaining variables with the new Missing (Y/N) variable and examine if there are any relationships between Missing (Y/N) and the remaining variables.

Let’s look at a simple example I have drafted below, where one variable has a number of missing values. We have 3 variables which are being proposed to predict “happiness”. The first is a measure of educational attainment, second is work experience and the third is salary. If you follow my analysis below, you will see there is a strong relationship between the salary variable, educational attainment and work experience. In fact, the correlation scores are all around the 60% mark. Work experience is correlated with some of the other variables, so you could conclude that MAR is the most likely missing value mechanism but the missing values could also be missing due to MCAR or NMAR.

We cannot be sure here. However, I would initially conclude in this situation the problem is a MCAR or a MNAR, because when I do a logistic regression on the Missing Y/N that I created it is not significantly related to either of the other 2 input variables.

Now my choice here would be MCAR because there is no intuitive reason to tell you that people of one or other experience level would choose to leave their answer out.

Follow this link to go to the Google Colab for this step.

<https://drive.google.com/open?id=1tSh07V0-agHBJ2RVgdbRUdf4apEQLPfR>

If there is a relationship then we are probably dealing with a MAR missing data process, and imputation would probably be ok. You can then use for example either an imputation by regression or a propensity score model.

If you don't have MAR then you have either a MCAR or NMAR process. To discount a NMAR process is difficult and generally requires quite a lot of content knowledge and research. You may in this situation have to make the following decisions:

* Do I impute using a random number generator, mean or median? This assumes the data is MCAR.
* Do I leave the variable out? What are the consequences of this? If so do I use imputation or do I just leave it out.

If a variable with missing values is highly correlated with other variables in your dataset then generally the R2 will by high (> 80%).

If the volume of missing values > 50% of the variable in question then I would recommend ignoring this variable or reframing your question to focus only on the subjects with near complete datasets.

Finally, if I impute I will always run my final analysis with the imputed values and without them. If there results are similar then I would be happy as you will probably imporve the power of your experiments. However, if they are not then I would reframe the problem. Be warned there is no hard and fast rule to deal with missing values so be careful.

## 2.6 Univariate approach to imputation

The next three steps are taken from the skikit learn repository.

The first of these is a simple tool to help you implement univariate imputation. Ski-learn’s SimpleImputer has options to use a constant, mean, median or mode for your missing values. It is important to visualise the data before you implement this process:

* When you have normally distributed variables you should probably use the mean.
* If you have outliers or skewed data use the median or the mode.

The code in the Google Colab file below gives a simple example of this. I would strongly recommend that you experiment with it so you can understand the implications of your choice.

Follow this link to go to the Google Colab for this step.

<https://drive.google.com/open?id=15ZFPAOkL_poOa0EH8ozgu0w2J2xPORq0>

## 2.7 Multivariate feature imputation

Multivariate imputation effectively uses complete variables to predict missing values. Skikit-learn describe it as follows:

“IterativeImputer models each feature with missing values as a function of other features and uses that estimate for imputation. It does so in an iterated round-robin fashion: at each step, a feature column is designated as output y and the other feature columns are treated as inputs X. A regressor is fit on (X, y) for known y. Then, the regressor is used to predict the missing values of y. This is done for each feature in an iterative fashion and then is repeated for max\_iter imputation rounds. The results of the final imputation round are returned.”

The estimates of The MICE package in R uses Multivariate Imputation by chained equations. The skikit-learn iterative imputer is inspired by MICE with the exception that it only produces one set of results.

If you go to the following link you will see a rather complicated approach to estimating missing values:

<https://scikit-learn.org/stable/auto_examples/impute/plot_missing_values.html#sphx-glr-auto-examples-impute-plot-missing-values-py>

The objective is to build a number of methods and the results are then combined/aggregated. This is very similar to a technique you will learn about called ‘random forest regression’. I like to think of this approach as getting a bunch of different experts into a room. Then they all put their opinion on a piece of paper and this is then aggregated to get the overall group opinion.

Follow this link to go to the Google Colab for this step.

<https://colab.research.google.com/drive/1WwRhJ2rhdWg7AM13YvlxpoYPSKrZO7AT?usp=drive_open>

## 2.8 Nearest neighbour imputation

Scikit-learn also gives you the KNNimputer which is a multivariate solution that effectively works on the rows of data rather than the columns.

The KNN stands for K-nearest neighbours. The algorithm effectively finds the rows with non-missing values that are closest to the row and with the missing value. The values of the non-missing neighbours are then used to create an estimate of the missing value. There are various approaches to this estimate and Scikit-learn outlines it as follows:

“When the number of available neighbours is less than n\_neighbors and there are no defined distances to the training set, the training set average for that feature is used during imputation. If there is at least one neighbor with a defined distance, the weighted or unweighted average of the remaining neighbors will be used during imputation. If a feature is always missing in training, it is removed during transform. For more information on the methodology.”

Let’s read the dataset from the previous example and run the algorithm in the following Google Colab notebook.

<https://colab.research.google.com/drive/1l6mOyruMWoQ8Ew7ZaMjSv98Slx7DoPRT?usp=drive_open>

## 2.9 The hazards of imputation

The following article from Towards Science provides a really useful recap on how to handle missing data.

I have already mentioned the pitfall of potentially biasing your results when you use the wrong technique or make unreasonable assumptions regarding missing data. It is really worth keeping focused on examining your assumptions. I continually tell students that statistics and machine learning are about trying to prove a point. Think of it as a court of law, you continually need evidence to prove your point. When you handle missing data make sure you examine your assumptions. Write them down and then complete an analysis for each one. For example, if you do decide the variable with the missing data is an MCAR then prove that the estimate that you intend to use for the missing values is appropriate. So look at the spread of the data and see if it suits a constant, mean, median or mode. I have asked students during their MSc vivas what technique they used. Nine out of ten say they imputed them with the mean and usually have no evidence to support it. This is a huge No-No.

The final point I will make is this; if you decide to impute, then run repeated analyses of your model with differing samples from your data. If you do this, the samples with imputed missing values should not stick out. If you were making a prediction model, do the test results change when you rely on missing data? You will come across this using K-folds neural networks in the future and this is also known as bootstrapping in statistics.

Bootstrapping is any test or metric that uses random sampling with replacement.

# Quiz

**Question 1 –** What does MAR mean?

* Missing Completely at Random
* Missing at Random
* Moderately Missing at Random

**Question 2 –** When can you use an imputation process with NMAR data? Choose the most appropriate answer.

* Never
* Anytime
* When you have a censored variable and you know why.
* Have evidence that this is a censored variable or you have apriori evidence on the distribution of the variable in question

**Question 3 –** Select the **false** statements:

* Missing Value imputation can cause bias.
* KNN imputation works best when the data is standardized.
* The mean is always better than the mode in univariate imputation
* You should drop the intercept when doing an imputation by regression

**Question 4 –** The random seed for multivariate feature imputation has an impact on the calculated missing values. Is this statement true or false?

* True
* False

**Question 5 –** Bootstrapping is a good idea when attempting to impute multivariate features. Is this statement true or false?

* False
* True

# Rule of thumb

* A chi-square test will determine whether missing data is MAR and MCAR.
* NEVER impute NMAR3

Topic 2 – Data Cleaning II

# Topic 3 – Data Cleaning II

# Data visualisation

## 3.2 Data visualisation

This step illustrates the reasons why we pre-process data.

When we do most things in life, we use our senses to understand and feel things.

Our brain absorbs an enormous amount of information, breaks it down or categorises it into groups in order to make decisions. Data visualisation is the process of making a picture from a vast amount of information. Remember, a picture paints a thousand words, so try and decide how you will present your visualisations to your final audience. It can give us the clarity we would not normally get when looking at a vast amount of data or words. For example, we have all looked at a spreadsheet or a database and gotten lost in the data. Watch the following TED talk by David McCandless on data visualisation. It is a little long (18 minutes) but it is worth it. It is incredibly interesting and shows the power of visualisation.

This is an additional video, hosted on YouTube: <https://youtu.be/5Zg-C8AAIGg>

I always tell students when completing their visualisations to be selective. Imagine you go to an art gallery or museum, such as the Lourve. We always enter these places with excitement (well, I do) and having walked around for a number of hours we get desensitised and can really miss the story of a wonderful painting. This is the same with visualisations in analytics. Sometimes we can have too many pictures. It is important to be selective and to try and get to the point as quickly as possible.

This article by Towards Science provides a really nice introduction to visualisation in Python:

<https://towardsdatascience.com/introduction-to-data-visualization-in-python-89a54c97fbed>

It works on the IRIS dataset and quickly demonstrates differing charts. Before we go to the next step I would like you to think about why we visualise data. We do it to tell a story but also to highlight characteristics such as anomalies or outliers. Ask yourself what an outlier is and what you should do with them.

## 3.3 Univariate outlier detection

This step explores univariate outlier detection.

In 1980, Hawkins described an outlier as “Observations which deviate so much from other observations as to arouse suspicion that they were generated by a different mechanism”. For me, this description is too simple and would lead one to conclude that these observations should be excluded or deleted from our data before we complete our analysis. In fact, my view of **an outlier is a data point that seems to stand out from the rest of the pack**. One should not immediately think this is an erroneous data point. It could, in fact, be a quirk of our data. For example, maybe after a certain period of time, our data was generated from a system that became unstable. While this is a change in process, it tells us that this can happen.

Sometimes, outliers represent people or subgroups that do strange or alternative things and could potentially tell us where future fashions or trends are going. They can perhaps help us to identify what is possible and can be a data miner’s gold and thus should not be excluded when identified. When you do identify an outlier, you should have strong evidence that the data point is erroneous before you delete or exclude it. The rule I use is this:

***A data point cannot be excluded unless there is scientific proof that the value you get is not possible.***

For example, if you were measuring heart rates of people and you found one with 1000bpm then this could be excluded from your analysis, as the fastest recorded heartbeat is 600bpm. You will note here that I researched the topic and found a paper that supported my evidence. The other thing you may say is that the data does not follow normal practice. This may be the case but you can only exclude the data point if your study is defined to be on normal or regular subjects. Let’s look at another example. Imagine you were predicting house prices and most of them ranged between €150k to €1m, but you found one or two at €40m. This is quite possible and these data points cannot be excluded unless you are going to narrow your models to houses under €1m.

Let’s look at some data that you can get from scikit-learn. It’s known as the Boston housing data and can be imported using the code in this Google Colab.

<https://drive.google.com/open?id=1U7CJvEQE52vb9V1QXnOA8_sp2mnOCxri>

## 3.4 Multivariate outlier detection

We are now going to look at multivariate outlier detection.

**Multivariate outlier detection is the use of more than one variable to identify an outlier.**

**In univariate outlier detection, we look for individual outliers in a single variable.**

These univariate outliers may not show up if there are unusual combinations between two or more variables. This is where multivariate outlier detection comes in. The same principles as mentioned in univariate outlier detection can be applied with multivariate outliers. In other words, don’t exclude particular rows of data unless you have the scientific evidence to support this.

There are numerous approaches to multivariate outlier detection, some of which are outlined below:

* Bivariate charts are based on bivariate normal assumptions.
* Depth based control charts (nonparametric) map n-dimensional data to one dimension using, for example, Mahalanobis and build control charts for depth.
* Multiscale process control wavelets detect abnormalities at multiple scales as large wavelet coefficients. They are useful for data with heteroscedasticity. For example, they can be used in chemical process control.

In this section, we will specifically focus on the depth-based control charts as it is a reasonably robust process and is quite quick to set-up. The multiscale process is beyond this material at this stage.

Follow this link to go to the Google Colab for this step:

<https://drive.google.com/open?id=1HavyI0iPtjDqvvx37EfESM0lYWl9q0gQ>

## 3.5 Heteroscedasticity and linear regression

In earlier topics, we discussed the assumptions when building regression models.

The simplest version of a linear regression model is:



Where is randomly and independently normally distributed around 0 with a constant variance. Heteroscedasticity occurs when the variation in the error terms is not constant. It can be caused when:

* There is a large range between the largest and smallest observed values (outliers).
* The model is not correctly specified.
* Observations are from mixed scales (Fahrenheit to Celcius).
* Skewness in the distribution of the regressor variable (dependent).

From the above list, we can see that we are assuming a linear model, but by using a linear model and examining the errors we can expose potential outliers in our data. I am not saying we have to stick with this model but using it initially can help you identify outlying observations. This article will help you understand what heteroscedasticity is and how to calculate it:

<https://www.geeksforgeeks.org/heteroscedasticity-in-regression-analysis/>

Have a look and use the code I gave you in the first course of Data Mining and Data Analytics in Step 3.9 to help you examine Boston housing data for outliers.

## 3.6 Benford's Law

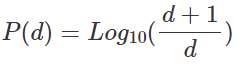
Have you ever observed the first digit of a number? Does the digit 1 have the same probability as 9 to be a leading digit?

According to Benford’s law, a.k.a. the first digit law, the frequency of occurrence of the leading digits in naturally occurring numerical distributions is predictable and nonuniform but more close to a power-law distribution. In fact, a given number is six times more likely to start with a 1 than a 9! This is very illogical, as most people would expect a uniform distribution U(1,9) where all the digits have the same likelihood to show up in the first slot so they expect a probability of 1/9 % ~ 11,1%.

In 1881, Simon Newcomb noticed that the earlier pages (that started with 1) were much more worn than those at the back and that’s why the leading digits were much more likely to be smaller than large. Then, in 1938, physicist Frank Benford rediscovered the theorem of Newcomb by collecting over 20,000 samples of real-world numbers, using data sources as distinct as the measurements of rivers, street addresses, numbers contained in Reader’s Digest magazine, molecular weights, baseball statistics, death rates, and more. As Benford popularized this scientific discovery, he received all of the credit.

### Benford’s Law

Benford’s Law **is an observation about the distribution of the frequencies of the first digits of the numbers in many different data sets. It is frequently found that the first digits are not uniformly distributed, but follow the logarithmic distribution**:



That is, numbers starting with 1 are more common than those starting with 2, and so on, with those starting with 9 being the least common. The probabilities are given in the table below:

|  |  |
| --- | --- |
| Digit | Probability |
| 1 | 0.301 |
| 2 | 0.176 |
| 3 | 0.125 |
| 4 | 0.097 |
| 5 | 0.079 |
| 6 | 0.067 |
| 7 | 0.058 |
| 8 | 0.051 |
| 9 | 0.046 |

Benford’s law can be applied to many situations and is **particularly useful in fraud detection**. I have downloaded the 2016 US election results for Illinois from here:

<https://www.futurelearn.com/courses/pre-processing-data-and-feature-impact-calculation/3/steps/1234470#:~:text=Benford%E2%80%99s%20law%20can,Google%20Colab%20file>.

However, I have edited the file (available below) to calculate the votes from the total votes multiplied by the percentages. In this analysis, we are going to examine the Republican votes for Illinois. This can be seen in the code in the following Google Colab file:

<https://drive.google.com/open?id=17KaQ8HhgoVvsWJADiFdPXHIcIi1AhGMg>

## 3.7 Noisy data

**Noise can generally be considered to be a random error or variance in a measured variable.**

It can generally come from the following situations:

* Faulty data collection instruments.
* Data entry problems.
* Data transmission problems.
* Technology limitations.
* Inconsistency in a naming convention.

Other data problems which require data cleaning are:

* Duplicate records.
* Incomplete data.
* Inconsistent data.

Can you think of an example that gives us noisy data?

Well, you come across many examples of noisy data in our daily lives. We have all been affected by bad mobile signals or fuzzy images on our TVs. These can come from interference or just plain old inconsistent data collection.

So how do we handle noisy data?

There are numerous techniques and as usual, there is no exact answer. We need to make sure that we do not over-process our data to the extent that it will remove inherent trends or cyclical patterns, but if there is too much noise in our data then we will be unable to make realistic predictions. So in effect, we want to be able to process our data enough so that it is useful. In addition, we may find noise in our data that is caused by some outside force or variable that we have not measured. One should also be careful to differentiate between noise and outliers. Outliers are relatively rare occurrences and noise can be systemic. Many commentators online fail in my opinion to differentiate between the two and treat them with the same processes.

**The following techniques are often used to handle noisy data:**

* Binning methods: smooths by bin means, bin median or by bin boundaries.
* Clustering: detects and removes outliers.
* Combined computer and human inspection: detects suspicious values and is checked by a human.
* Regression: smooths by fitting the data into regression functions.
* Moving Average/Exponential Smoothing etc: smooths time series values to remove inherent noise.
* Wavelet Analysis/Fourier Transforms: creates new variables from alternate frequencies.

In the next two steps, we will look at some discretization methods and a time series techniques such as moving average and exponential smoothing. There are many smoothing techniques and they designed to detect trends in the presence of noisy data in cases in which the shape of the trend is unknown. However, before you try any of these approaches always examine the variables/features in question using simple graphs. They can tell you a lot about your data and can certainly in a time series case, tell if there is a strong noise to signal ratio. Finally, if you smooth a variable and use this variable as a predictor for your model then remember you are removing the variation from your analysis. This can also potentially introduce bias. This will have a potential impact on the range of outcomes. So you might find that you need to adapt your models to take account of this.

**Signal-to-noise ratio (SNR or S/N)** is a measure used in science and engineering that compares the level of a desired signal to the level of background noise. SNR is defined as the ratio of signal power to the noise power, often expressed in decibels. A ratio higher than 1:1 (greater than 0 dB) indicates more signal than noise.

The following links will give you a feel for how much smoothing is viable and also some of the techniques required to do this:

Why smooth? <https://rafalab.github.io/dsbook/smoothing.html>

How much should we smooth? <https://www.stat.cmu.edu/~cshalizi/uADA/12/lectures/ch04.pdf>

Take a look at these links and try and get a feel for why we would smooth our data.

Chart, line chart

Description automatically generated

## 3.8 Discretization techniques

In the previous topic, we briefly mentioned a number of univariate discretization techniques.

These techniques can be used to categorize a continuous variable or simply to smooth a variable. We will briefly expand on a number of the discretization binning techniques in this topic.

# Simple Discretization Methods: Binning

The following binning techniques are the most simple. The first is where the bins are of equal size and the volume of data points differs for each bin. This can be very useful when trying to understand the density of your data. The second is where the bin sizes vary but the density of each bin remains the same. These can be useful in understanding possible outliers and the spread of your data.

* **Equal-width (distance) partitioning** divides the range into N intervals of equal size: uniform grid. If A and B are the lowest and highest values of the attribute, the width of intervals will be: W = (B-A)/N. This is the most straightforward approach but outliers may dominate the presentation if skewed data is not handled well.
* **Equal-depth (frequency) partitioning** divides the range into N intervals, each containing approximately the same number of samples. This approach is good data scaling and managing categorical attributes can be tricky.

The diagram below shows how they work and that they typically look like histograms. Neither of the above methods examines if there is any predictive power lost as we are not taking account of the effect of binning on the outcome variable.

Chart, bar chart, histogram

Description automatically generated

# Binning Methods for Data Smoothing

In this approach, we are simply using the binning technique to smooth our data. So we use the bins to calculate new values that will effectively replace the old ones. Let’s see how we do it for the following data:

Sorted data for price (in dollars): 4, 8, 9, 15, 21, 21, 24, 25, 26, 28, 29, 34

**Step 1** – Partition into (equi-depth) bins:

* Bin 1: 4, 8, 9, 15
* Bin 2: 21, 21, 24, 25
* Bin 3: 26, 28, 29, 34

**Step 2** – Smoothing by bin means (you could use the median):

* Bin 1: 9, 9, 9, 9
* Bin 2: 23, 23, 23, 23
* Bin 3: 29, 29, 29, 29

or Smoothing by bin boundaries:

* Bin 1: 4, 4, 4, 15
* Bin 2: 21, 21, 25, 25
* Bin 3: 26, 26, 26, 34

The following code example from GeeksforGeeks does the above analysis for the IRIS dataset.

<https://www.geeksforgeeks.org/python-binning-method-for-data-smoothing/>

Try it out for the remainder of the columns in the IRIS dataset in the following Google Colab file:

<https://drive.google.com/open?id=1SJ0b6bqakqMZvoBMkJRsOtr4w8ByYQvS>

## 3.9 Moving average and exponential smoothing

Moving averages are used and discussed quite commonly by technical analysts and traders alike.

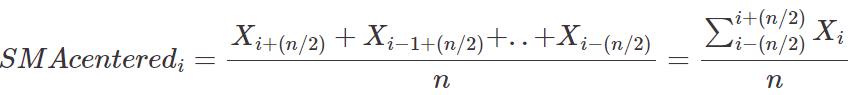
If you’ve never heard of a moving average, it is likely you have at least seen one in practice. A moving average can help an analyst filter noise and create a smooth curve from an otherwise noisy curve. Mathematically, a moving average is a type of convolution and so it can be viewed as an example of a low-pass filter used in signal processing. If you are using them for predictions then generally you will lag them on historical data (Posey, 2019). However, Python also provides an option which allows you to smooth the data by centring on a point. So values either side of this point would be included in the moving average.

In order to calculate a moving average, we basically apply a window to your data which slides as time moves by. In each window/period(n), you calculate the average value and create a new series with the sliding/moving averages. The simplest form of moving average is the simple moving average (SMA)and the formula is as follows:

A picture containing antenna

Description automatically generated

The centred moving average is as follows:



The following video explains moving average smoothing really nicely.

[VIDEO] <https://youtu.be/7Rz_ITRIADg>

There are many other moving average techniques such as the Cumulative and Weighted Moving Average.

During previous steps in this topic, we mentioned how smoothing can be used to remove noise. There are many different approaches to this and each one has its own strengths and weaknesses. In this step, we will focus on two very common approaches; Simple Moving Averages and Exponential Smoothing.

Follow this link to Google Colab for this step:

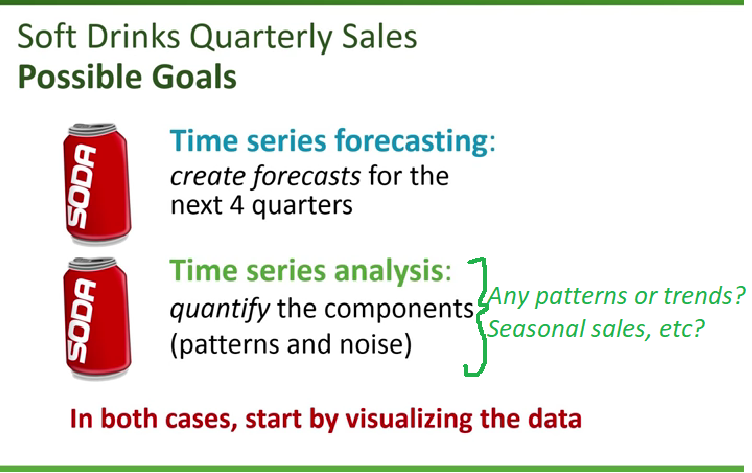
<https://colab.research.google.com/drive/1k3rhVIqkoU5f-mtLxihatDvCZ3T2m6-n?usp=drive_open>

Deciding how best to set your windows is vital:

* Weekly? Monthly? Quarterly? Annually?
* What type of moving average to use? Centred? Trailing?

Diagram

Description automatically generated



The centre line or **centred moving average** smooths out the seasonal data to give us a good picture of the overall trend.

Chart

Description automatically generated

Graphical user interface, application

Description automatically generated

Quiz

**Question 1 –** Which missing data mechanism should not be applied with missing value generation packages?

* MCAR
* MAR
* NMAR

**Question 2 –** If the censoring mechanism is known, we can mitigate the bias from our analysis. True or false?

* True
* False

**Question 3 –** Spot the false statements:

* Outliers are potentially legitimate.
* Outliers prevent us from getting a real feel for our data.
* Outliers can be a data miner’s dream.
* Outliers are data or model glitches.

**Question 4 –** A control chart can help us spot when a data stream is moving away from expected behaviour. Should we act only when the data exceeds the control limits?

* Yes
* No

**Question 5 –** Heteroscedasticity implies we have a constant variance. True or false?

* False
* True

**Question 6 –** Smoothing methods help us handle Noisy Data. Tick the correct statements below. Select all the answers you think are correct.

* Exponential smoothing gives larger weights to the most recent data.
* Exponential smoothing methods are most effective when the parameters describing the time series are not changing slowly.
* Moving average only uses the last k data points and gives them an equal weight.

Topic 4 – Data reduction and transformation

# Topic 4 – Data reduction and transformation

# Data transformation: Normalization

4.2 Data transformation: Normalization  
Data transformation is the application of a functional transformation or techniques that will transform or convert data to a new variable in order to help the analyst to meet model assumptions for a given algorithm or help summarize the data.

The following list helps to further explain the general areas where it can be applied:

* **Smoothing:** remove noise from data
* **Aggregation:** summarization, data cube construction
* **Generalization:** concept hierarchy climbing
* **Normalization:** scaled to fall within a small, specified range, including min-max normalization, z-score normalization and normalization by decimal scaling.

Analysts will transform data as a matter of convenience in order to reduce skewness and make the data easier to view. They will transform data to convert nonlinear relationships to linear relationships or to convert multiplicative relationships to additive relationships.

In the previous topics, we discussed the processes of smoothing and data aggregation. We will now discuss normalization, concept hierarchy and the use of transformation techniques to reduce our datasets without loss of information.

## 4.3 Data normalization in data mining

Normalization is used to scale the data of an attribute so that it falls in a smaller range, such as -1.0 to 1.0 or 0.0 to 1.0.

It helps certain machine learning algorithms avoid bias in their predictions.

Many machine learning algorithms require the input attributes to be scaled as the cost function used to optimize the weights/parameters will be influenced by values from differing scales. It is not a prerequisite for all algorithms though, and should only be used when required. Currently, there is a fashion in the machine learning world to do it as a matter of course. Any transformation approach will reduce the amount of information arising from the variable in question.

**Situations where you may want to standardise are when:**

* The variables are measuring different physical quantities.
* The numeric values are on vastly different scales of magnitude.
* There is no evidence that variables with high variation should be considered more important.

**Situations where you will not want to standardise are when:**

* Variables are the same physical quantity and are roughly the same magnitude.
* Standardised variables do not change between samples. It may be worthwhile excluding them
* You have such physically related variables, your measurement noise may be roughly the same for all variables but the signal intensity varies much more, i.e., variables with low values have higher relative noise. Standardizing would blow up the noise. In other words, you may have to decide whether you want relative or absolute noise to be standardized.

**Scaling or standardization should only be done unless it's absolutely necessary**. If the algorithm you use requires it then yes go ahead. So for example, **if you are using a neural network then you will have to do it**. However, **if you are using a Linear Regression model then there is absolutely no need to do it**.

In this step, we will examine the following data normalization techniques:

* Min-Max Normalization
* Z-Score Normalization
* Normalization by Decimal Scaling

### Mix-Max normalization

**Mix-Max normalization** uses the max and min values of series in order to convert the series to a series of values between 0 and 1. A Min-Max scaling is typically done using the following equation:

A picture containing text, clock, watch

Description automatically generated

### Z-score normalization

**Z-score normalization** converts a variable to a standard normal distribution, using the following formula:

A picture containing text, clock

Description automatically generated

Where X¯ is the series average and σ is the sample standard deviation. **The one benefit to this technique is** **outliers will have less impact** than the other 2 techniques. It will also center the attribute so the interpretation of the estimated weights/parameter estimates will change in your analysis.

### Normalization by Decimal Scaling

With **Normalization by decimal scaling**, we are basically taking the largest number in our dataset divided by 10 to the power of j such that this number is less than 1. It can be described as follows:

Logo

Description automatically generated

You can find the Google Colab for this step here: <https://drive.google.com/open?id=13JQ7BYJEC_6GWNyX2oq9f4GH4OUzLZ2p>

Once you have finished going through the Colab file, consider the following question:

Can you point out a drawback to where these scaling processes have a major weakness?

## 4.4 Data and numerosity reduction

We are all aware of the volumes of data that are currently being generated by modern technology such as multi-media, IoT or social networks.

In real-life situations, we may have to deal with 50,000+ features. This can be very helpful but also challenges us to come with machinery and algorithms that can handle such scale. Sometimes, this can be similar to looking for a needle in a haystack when we are trying to extract insights from such vast datasets.

In addition to the above, many of the machine learning algorithms that are available today specifically work as classifiers. So to use such algorithms you may need to convert a continuous variable into a categorical one.

So are there ways to handle such problems?

Well, yes. In the next couple of steps, we will address these problems as two broad groups of problems. These are groups are:

* Numerosity reduction
* Dimensionality reduction

### Numerosity reduction

Numerosity reduction allows us to transform continuous variables into categorical variables. The approaches we will look at for this group are; concept hierarchies and discretization.

**Concept hierarchies** reduce the data by collecting and replacing low-level concepts (such as numeric values for the attribute age) with higher-level concepts (such as young, middle-aged or senior). Example:

All ages 1, 2, 3… 100 🡺 ***replaced with*** 🡺 Child, Adolescent, Young Adult, Middle Age, Old

**Discretization** reduces the number of values for a given continuous attribute by **dividing the range of the attribute into intervals**. Interval labels can then be used to replace actual data values. We have covered some of the basics of this subject in previous topics but we will advance this to cover the concept of entropy and information gain.

### Dimensionality reduction

The second group covers the concept of dimensionality reduction. This is a vast area which will be covered in much more detail in the next course: Feature Engineering. However, firstly, we will introduce a number of tools that you can use to get started with. They are:

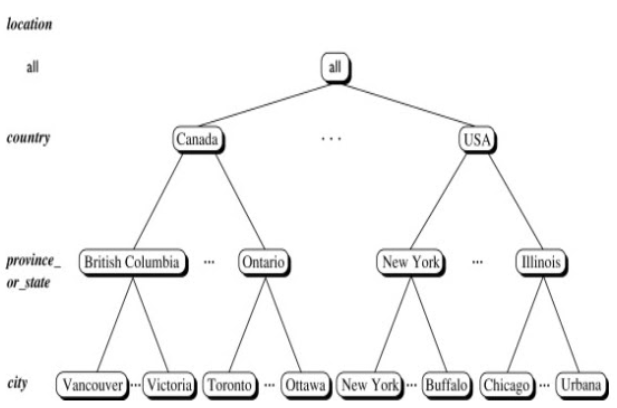
* A light introduction to Principal Components
* Lasso Regression
* Ridge Regression

## 4.5 Concept hierarchies

This step explores concept hierarchies.

The following content comes from Han et al, (2011: 143-144).

“A concept hierarchy defines a sequence of mappings from a set of low-level concepts to higher-level, more general concepts. Consider a concept hierarchy for the dimension location. City values for location include Vancouver, Toronto, New York and Chicago. Each city, however, can be mapped to the province or state to which it belongs. For example, Vancouver can be mapped to British Columbia and Chicago to Illinois. The provinces and states can, in turn, be mapped to the country (e.g., Canada or the United States) to which they belong. These mappings form a concept hierarchy for the dimension location, mapping a set of low-level concepts (i.e., cities) to higher-level, more general concepts (i.e., countries). This concept hierarchy is illustrated in the figure below.



Many concept hierarchies are implicit within the database schema. For example, suppose that the dimension location is described by the attributes number, street, city, province\_or\_state, zip\_code, and country. These attributes are related by a total order, forming a concept hierarchy such as “street < city < province\_or\_state < country.” This hierarchy is shown in Figure 4.10(a). Alternatively, the attributes of a dimension may be organized in a partial order, forming a lattice. An example of a partial order for the time dimension based on the attributes day, week, month, quarter, and year is “day <{month < quarter; week} < year.”

Concept hierarchies may also be defined by discretizing or grouping values for a given dimension or attribute, resulting in a set-grouping hierarchy. A total or partial order can be defined among groups of values. An example of a set-grouping hierarchy for a price dimension, where an interval (X…Y) denotes the range from X (exclusive) to Y (inclusive). This could be the conversion of the variable age to infant, teenager young adult or pensioner, defined by some predefined standard.”

Finally, in order to reduce data categories or numerosity, an examination of the data warehouse fact tables or the application of predefined standards can be used to categorize data.

## 4.6 Discretization continued

In Step 3.11, we discussed binning techniques that could be used to discretize a feature.

These processes, while useful in understanding and pre-processing our data, can also be considered data numerosity reduction techniques. Generally, we have three types of attributes/features or variables. These are nominal variables, ordinal variables and continuous. These are each discussed in turn below.

### Nominal Variables

Nominal variables are values that come from an unordered set such as; Male or Female, and Yes or No. They may contain more complex values such as the names of several political parties (e.g. Conservative, Labour, Liberal, Scottish National Party, Green Party).

### Ordinal Variables

Oridinal variable are values that come from an ordered set, such as; Very Unsatisfied – 1, Unsatisfied – 2, Neutral – 3, Satisfied – 4, Very Satisfied – 5.

### Continuous

These are real numbers and are often described as **interval scales** or **ratio scales**. Have a look at the following link for a more detailed explanation of ratio and interval variables:

<https://www.questionpro.com/blog/ratio-scale/>

Following example of ratio level of measurement to help understand the scale better.

**Please select which age bracket you fall in:**

* Below 20 years
* 21-30 years
* 31-40 years
* 41-50 years
* 50 years and above

As we said in Topic 3 of this MOOC, discretization allows us to divide a continuous variable into intervals/categories.

Further to the techniques we have looked at in previous steps in this course, we will look at a very simple rule natural partitioning approach known as the ‘**3-4-5 Rule**’ in the next step. Following this, we will look at an entropy-based discretization method which takes account of the output variable when deciding on how to discretize an input variable.

Please post any comments or questions that you may have in the comments section below.

## 4.7 3-4-5 rule for attribute discretization

The 3-4-5 rule can be used to segment numerical data into relatively uniform, “natural” intervals.

**RULE 1:** If an interval covers 3, 6, 7 or 9 distinct values at the most significant digit, partition the range into 3 equal-width intervals for 3,6,9 or 2-3-2 for 7.

**RULE 2:** If it covers 2, 4, or 8 distinct values at the most significant digit, partition the range into 4 equi-width intervals.

**RULE 3:** If it covers 1, 5, or 10 distinct values at the most significant digit, partition the range into 5 equi-width intervals

**Example**

* Suppose that profit data values for the year 2017 for a company range from -351,976 to +4,70,00,896.
* For the practical purpose of avoiding noise, extremely high or extremely low values are not considered. So first we need to smooth out our data. Let’s discard the bottom 5% and the top 5% values.
* Now suppose after discarding top 5% and the bottom 5% the new values for LOW and HIGH are {-159876,1838761} respectively.
* **The Most Significant Digit (MSD) is at the million position**, see highlighted digit: –159876 and 1838761.
* The next step is to ***round down LOW*** ***and*** ***round up HIGH*** ***to MSD*** that 1 million position.
* So…
* LOW = -1000000 (nearest down million from -159876)
* HIGH = 2000000 (nearest up million from 1838761)
* Now let’s identify the range of this interval.
* Range = HIGH – LOW that is:
* 2000000 – (-1000000) = 3000000.
* We consider only MSD here which is 3 (times 1 million position).
* Now that we know range **MSD = 3**, we can **apply** **rule #1**….
* **Rule #1** says that we can divide this interval into three equal-size intervals:
* *Interval 1:* -1000000 to 0
* *Interval 2:* 0 to 1000000
* *Interval 3:* 1000000 to 2000000

You should be thinking of **how 0 can be part of multiple intervals?** You’re right! We should represent it as follows:

* *Interval 1:* (-1000000 to 0]
* *Interval 2:* (0 to 1000000]
* *Interval 3:* (1000000 to 2000000]

Above uses notation for **half-open** interval: “(“ and “]”. E.g.

**(a … b]** denotes “**range that excludes a but includes b**”.

## 4.8 Entropy

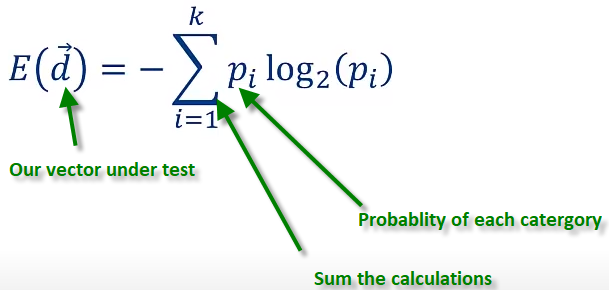
So far any of the methods we have used for data discretization have predominantly focused on the variable/attribute itself and not on the information it contributes to the outcome variable.

In order to advance what we have learnt before we need to do the following when discretizing a data series:

Consider the information we get from the new categories.

Ask how can we get the right splits in a series that will give us the most information.

Entropy refers to disorder or impurity. The decision tree classifier uses this measurement of information to determine if a split at a certain point improves the information gain obtained at that split or not. In effect, what we are trying to do is to make splits that will be most beneficial when categorising an attribute from a continuous variable. The formula for Entropy can be shown below:



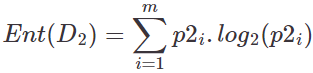
Say we have entropy for an attribute ***D***:



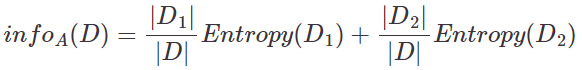
Where:

* ***pi*** is the probability of having outcome event ***i***
* ***D1*** is the first category of the proposed attribute ***D***

If we do this for another category for ***D2*** we can calculate the entropy for ***D2*** similar to that above:



So if we were proposing to split a continuous attribute ***D*** in two, we would determine the information that a proposed split would give with the following equation:



where ***D*** is the total number of items and ***D1*** and ***D2*** are the number of items in each class for the attribute split in ***D***. The challenge we face is to **pick the split that gives the most information gain from our first split**.

The following link from Natalie Meurer is a lovely description of how the information gain from a split is calculated. Natalie has also provided some really nice examples that should make it much clearer:

<https://natmeurer.com/a-simple-guide-to-entropy-based-discretization/>

### Splitting an Attribute in Practice

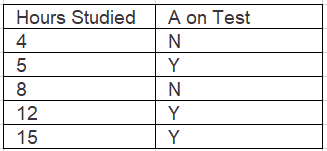
One way to split an attribute in practice is to model the outcome with the attribute in question, using a decision tree algorithm. The example we saw in the above link can be run through the python Scikit Learn decision tree library, using the **C4.5 algorithm**:

<https://scikit-learn.org/stable/modules/tree.html#classification>

The C4.5 algorithm is a child of the **ID3 algorithm**:

<http://www.learnbymarketing.com/481/decision-tree-flavors-gini-info-gain/>

They use information gain and entropy to optimize their cost functions. We will take the example from Natalie Meurer’s webpage and use the C4.5 decision tree to determine the optimum split. The example can be seen below.



The code to do this is really simple.

1. We first specify:

* input variable: ***X***
* output variable ***Y***

1. And implement the decision tree classifier.
2. We then use the ***tree.plot\_tree*** to present the tree 🡺 The tree shows where the optimum splits based on information gain.

You should be able to see that the first split is at 10. This is equivalent to that found by Natalie Meurer. It then goes further until all tree nodes have a gini index of zero where you have complete information. A gini index of 0 means all the elements of the branch in question belong to a certain class. When all the leaves have a gini index of 0, your tree cannot grow anymore.

You can access the Google Colab for this step here:

<https://drive.google.com/open?id=1MBldCnQ0J0lSKkqgrrm4l5p_Yfq1ROfy>

Also see:

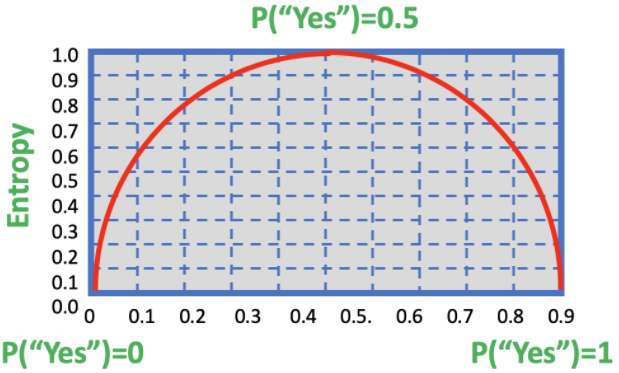
<https://www.analyticsvidhya.com/blog/2021/02/how-to-split-decision-tree-the-pursuit-to-achieve-purity/>

<https://www.analyticsvidhya.com/blog/2021/03/how-to-select-best-split-in-decision-trees-gini-impurity/>

To summarise:

* High entropy 🡺 high disorder, more uncertainty
* Low entropy 🡺 good idea of the answer
* 50:50 split 🡺 Might as well be heads or tails, 0 certainty, i.e. 100% entropy
* Known answer for certain 🡺 0 entropy
* LOOK FOR LOW ENTROPY!
* **Information gain** 🡺 **a measure of the reduction in the overall entropy** of a prediction task by testing on that feature.

The ideal discriminatory feature will partition the data into pure subsets where all the instances in each subset have the same classification.



## 4.9 Dimensional reduction techniques

In a previous step in this topic, we mentioned the fact that the volume of data available to analysts has exploded over the last 20 years.

This now presents new problems with regard to what methods we use, which features we use and whether we need to select all the data points for our analysis, in order to get insightful results. Another issue that arises is whether or not it is necessary to collect all this data. As in many cases, there will be a substantial amount of missing data. We have addressed techniques that can be used with missing data. Thankfully, there are a number of techniques which can be used to reduce the size of our datasets in terms of the number of features and the volume of data, without losing too much information. In fact, you can sometimes improve the model’s predictive performance and will definitely reduce the computational time required.

### What is dimensional reduction?

As data generation and collection keeps increasing, visualizing it and drawing inferences becomes more and more challenging. Many of us look at our data from the perspective of two or three-dimensional charts. When you have substantially more features, how can you extract insight from these charts? In fact, we can end up being quite confused, as one chart seems to show something but the other doesn’t. Also **if we have petabytes of data, do we need to use all of it? The answer to these questions is to use data reduction through a number of techniques:**

1. **Aggregation** (first course, Topic 2, Steps 3-10) and discretization (first course, Topic 3, Step 11)
2. **Sampling** (row reduction).
3. **Dimensionality** **reduction**/**Feature** **subset** **selection**.
4. **Feature** **creation**/**Attribute** **transformation** (Covered in the next course: Feature Engineering).

We have covered the use of aggregation using binning techniques or histograms, so we are now going to look at row reduction using sampling techniques and dimensionality reduction. In other words, we reduce either the number of rows we need to do our analysis or the number of features we need for our analysis. We will discuss feature creation in the next course: Feature Engineering.

Let’s have a look at row reduction or sampling first.

### Sampling

Sampling has been the traditional approach when collecting data for social surveys. We are all familiar with the use of samples to determine elections or when market research companies are assessing people’s product preferences. If the sample is picked correctly we get a sub-dataset that reflects the original dataset and thus we can make predictions about the original dataset. There are many techniques that can be used to sample your data. These include simple random sampling, systematic sampling, stratified sampling and cluster sampling. Each of these will be discussed in turn next.

#### Simple random sampling

Simple random sampling is easy to implement as **each row or individual in your dataset has an equal chance of being selected**. It does not guarantee that every group (if they exist) in your data will be represented.

#### Systematic sampling

Systematic sampling is easier to use than simple random sampling as the **selection is based on a regular interval (e.g. every 10th row or individual)**. It **may cause bias if there are regular fluctuations in your data**. It is not really appropriate for time series data.

#### Stratified sampling

Stratified sampling **clusters data into a number of subgroups** or categories and is **a good way of truly representing the data and reducing bias**. However, it requires a detailed knowledge of the data structure.

#### Clustered sampling

Clustered sampling is similar to stratified sampling. **It can introduce bias if the cluster does not represent the characteristics of the data**.

There are many non-probabilistic techniques ranging from convenience sampling, quota sampling, judgement sampling and snowball sampling.However, these techniques are predominantly used in social science surveying and would not be particularly relevant to data analytics projects. If you want to explore this more try this link:

<http://www.csun.edu/~hbsoc126/soc4/chapter_8_outline.pdf>

When choosing a sampling technique try and pick one that reflects the structure of your data. So for example**, in election samples practitioners will make sure there are individuals in all the relevant socio-economic groups**. **With Simple Random Sampling random sampling (SRS) there is no guarantee of this**. The choice of technique should be carefully thought out.

### Dimensionality Reduction

So let’s think about the second problem (feature/dimensionality reduction). In the past few steps, we discussed two very simple concepts known as correlation and multi-collinearity. Both of these have measurements such as Pearson’s correlation coefficient and VIF which allow us to measure the linear relationship between variables. In large datasets, you will probably find a considerable amount of variables that are correlated in some way. For example, blood pressure and BMI are regularly correlated in health studies. **For many data analysts, the first thing we all do is review the correlation matrix. If there are variables in the matrix with extremely high correlations >0.9 then this would lead us to believe we will gain nothing by having both variables in our models**.

Just a side note here: we will lose model performance if we have highly correlated variables in our models.

The next thing we might do would be to analyse the VIF results. If we had VIF results for specific variables >10, then we might consider dropping this variable. The problem with these approaches is that we don’t know which variable to select.

There are a large number of dimensionality reduction techniques, and they all have their positives and negatives. According to The NEWSTACK

<https://thenewstack.io/3-new-techniques-for-data-dimensionality-reduction-in-machine-learning/>

the most commonly used techniques for data-dimensionality reduction are:

1. Ratio of missing values
2. Low variance in the column values
3. High correlation between two columns
4. Principal component analysis (PCA)
5. Candidates and split columns in a random forest
6. Backward feature elimination
7. Forward feature construction
8. Linear discriminant analysis (LDA)
9. Neural autoencoder
10. t-distributed stochastic neighbour embedding (t-SNE)
11. Multiple Correspondence Analysis (MCA).

We will discuss a large number of these in the Feature Engineering section in the third course in this program, Feature Engineering. Over this and the next 2 steps we are going to outline the following techniques:

* Principal component Analysis (regularly used)
* Ridge regression (used as a reference for lasso regression)
* Lasso regression (helps select a subset of covariates)

### Principal Component Analysis

Principal component analysis (PCA) is a dimensionality reduction technique that is used regularly in data analytics. It examines the linear relationship between variables and attempts to create new variables that are orthogonal (not correlated). By selecting a small subset of the orthogonal variables we can maintain 90%+ of the information of the original variables, thus reducing the dimensionality of the proposed dataset. Each new orthogonal variable is known as a component and they are generally ordered with regard to the amount of variation they explain. So, for example, the first component will explain the most variation within the dataset, the second will explain the second most amount of variation and so on. The big advantage with PCA is that it reduces the dimensionality of the problem substantially and can in fact create new latent features that will allow us to group variables together without having to make a decision as to which of the original variables we should pick. The link above to Wikipedia will give you more detail on the background to PCA.

Now, I want to point out a very important issue that all students seem to forget when they use **PCA** and that is it **should only be used on continuous variables**. Most students I have had in the past try it on nominal variables and it frankly makes no sense to do this. You can use a method called Multiple Correspondence Analysis (MCA) to help you address this problem.

In the following Google Colab notebook there is a simple implementation of a PCA on the IRIS data. Review the code and then try it on the Scikit Learn Breast Cancer data. How much of the overall variation is explained by the first two components?

<https://cxolab.research.google.com/drive/1MIBEnJ6wwGildCIL8eFM5EfUJwHy7kqL?usp=drive_open>

## 4.10 Ridge regression L2

Recap:

* **Covariance** indicates the direction of the linear relationship between variables.
* **Correlation** measures both the strength and direction of the linear relationship between two variables. Correlation is a function of the covariance.
* **Collinearity** is the correlation between predictor variables (or independent variables), such that they express a linear relationship in a regression model. In other words, they explain some of the same variance in the dependent variable, which in turn reduces their statistical significance.
* **Variance inflation factor (VIF)** is a measure of the amount of multicollinearity in a set of multiple regression variables. Mathematically, the VIF for a regression model variable is equal to the ratio of the overall model variance to the variance of a model that includes only that single independent variable.

In Step 3.9 of the first course in this program, we mentioned that it is unwise to include variables in a model that possesses multicollinearity.

Multicollinearity can create inaccurate estimates of the regression coefficients, and inflate the standard errors of the regression coefficients. This will impact on the partial t-tests for the regression coefficients and will thus give false nonsignificant p-values and degrade the predictability of the model (and that’s just for starters).

There are five sources (see Montgomery et al. (2001) for more details):

1. **Data** **collection:** In this case, the data have been collected from a narrow subspace of the independent variables\*\*. The multicollinearity has been created by the sampling methodology - it does not exist in the population. Obtaining more data on an expanded range would cure this multicollinearity problem. The extreme example of this is when you try to fit a line to a single point.
2. **Physical** **constraints of the linear model or population**: This source of multicollinearity will exist no matter what sampling technique is used. Many manufacturing or service processes have constraints on independent variables (as to their range) either physically, politically or legally, which will create multicollinearity.
3. **Over-defined model:** Here, there are more variables than observations. This situation should be avoided.
4. **Model choice or specification:** This source of multicollinearity comes from using independent variables that are powers or interactions of an original set of variables. It should be noted that if the sampling subspace of independent variables is narrow, then any combination of those variables will increase the multicollinearity problem even further.
5. **Outliers:** Extreme values or outliers in the X-space can cause multicollinearity as well as hide it. We call this outlier-induced multicollinearity. This should be corrected by removing the outliers before ridge regression is applied.

The concept behind ridge regression, also known as L2 regularization, is to adjust the estimates that you would normally get from ordinary least squares regression to give new estimates (which have a small amount of bias). This will have unbiased variance and subsequently deals with inflated VIF and reduce overfitting. It doesn’t get rid of attributes but can point you to those which are less significant. In ridge regression, the cost function is altered by adding a penalty equivalent to the square of the magnitude of the coefficients. It can help us to identify and deal with overfitting.

In ordinary least squares regression, we minimise the following cost function:



Which gives:



In ridge regression we have the following :



Which gives:



and is equivalent to saying we are going to minimize the cost function for the ordinary least squares regression under the condition below:



Choosing a value for k is not a simple task, which is perhaps one major reason why **ridge regression isn’t used as much as least squares or logistic regression**. You can read one way to find k in Dorugade and D. N. Kashid’s paper Alternative Method for Choosing Ridge Parameter for Regression. The literature does recommend that the  value be kept under 0.3.

Have a look at the following Google Colab. In it, we do an implementation of ridge regression. Do you think it is a good solution?

<https://colab.research.google.com/drive/1DrDdnISDq5X45A6BWaywWhQb4fI4XGX1?usp=drive_open>

## 4.11 Lasso regression L1

Lasso regression is an extension of ridge regression but has one major benefit. This is due to its ability to directly help with feature reduction.

“The key difference between these techniques is that **lasso shrinks the less important feature’s coefficient to zero thus, removing some features altogether**. So, this works well for feature selection in case we have a huge number of features.”

In ridge regression we had the following formula:



which gives:

A picture containing clock, watch

Description automatically generated

Now, for lasso regression the formula changes slightly, we replace the with



The impact of this change is that some of the parameters estimate  will be set to zero, thus helping us with variable selection. There is a really nice explanation here.#:

<https://stats.stackexchange.com/questions/74542/why-does-the-lasso-provide-variable-selection>

### Elastic Net

Elastic Net is an ensemble of both the L1 and L2 regularization techniques. Generally, lasso regression will eliminate many features and reduce overfitting in your linear model. Ridge regression will reduce the impact of features that are not important in predicting your y values. Elastic Net combines feature elimination from lasso and feature coefficient reduction from the ridge model to improve your model’s predictions. We have completed a small example of it at the end of these notes.

Let’s look at the data we had in the ridge regression step and see how it performs under lasso regression. Follow this link to go to the Google Colab file for this step.

<https://colab.research.google.com/drive/15ENdWCdwYAVMugUua1ugQOp41Ilb-pwh?usp=drive_open>

# Quiz

**Question 1 –** Data transformations can help us in the following areas:

* Generalization: concept hierarchy climbing.
* Aggregation: summarization, data cube construction.
* Smoothing: remove noise from data.

**Question 2 –** Min-Max normalization uses standard deviation in its calculation:

* False
* True

**Question 3 –** With normalization by decimal scaling, we are basically taking the smallest number in our dataset divided by 10 to the power of j such that this number is less than 1:

* True
* False

*We are basically taking the* ***largest*** *number…*