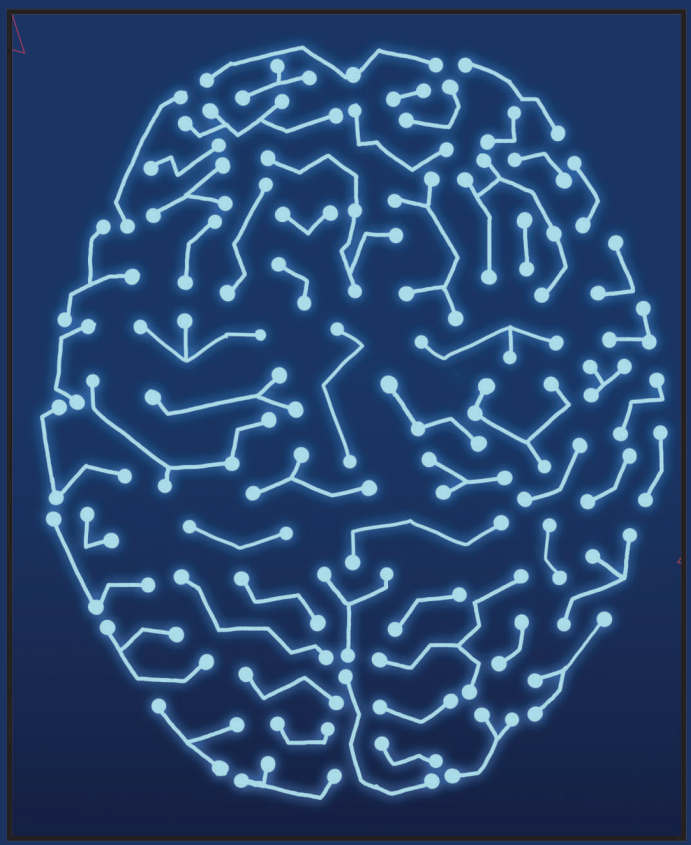
Women University Press
Jayoti Publication Desk
ISBN No. : 978-93-90892-53-2

DATA SCIENCE AND BIG DATA



JV'n Dr. Anamika Ahirwar

JAYOTI VIDYAPEETH WOMEN'S UNIVERSITY, JAIPUR

UGC Approved Under 2(f) & 12(b) | NAAC Accredited | Recognized by Statutory Councils

Printed by :
JAYOTI PUBLICATION DESK

Published by :
Women University Press
Jayoti Vidyapeeth Women's University, Jaipur

Faculty of Education & Methodology

Title: Data Science and big data

Author Name Dr. Anamika Ahirwar

Published By: Women University Press

Publisher's Address: Jayoti Vidyapeeth Women's University, Jaipur
Vedaant Gyan Valley,
Village-Jharna, Mahala Jobner Link Road, NH-8
Jaipur Ajmer Express Way,
Jaipur-303122, Rajasthan (INDIA)

Printer's Detail: Jayoti Publication Desk

Edition Detail: I

ISBN: 978-93-90892-53-2

Contents

Unit I- Introduction of Data Science & R Programming

6.6 Applications of R Programming

6.7 What is R In Data Science important?

6.8 What Makes R Suitable For Data Science?

6.9 Data Science Companies that Use R

7. Exploratory Data Analysis and the Data Science Process
   7.1 Exploratory Data Analysis
   7.2 Data Science Process
8. Basic tools (plots, graphs and summary statistics) of EDA
   8.1 Exploratory data analysis
   8.2 Types of Data
9. The Data Science Process - Case Study, Real Direct (online real estate firm)

UNIT- II (Basic Machine Learning Algorithms & Applications)

1. Linear Regression for Machine Learning
2. k-Nearest Neighbors (k-NN)
   2.1 Working of KNN Algorithm
   2.2 Implementation in Python
   2.3 Pros and Cons of KNN
   2.4 Applications of KNN
3. k-Means
   3.1 Applications
4. Filtering Spam
   4.1 What is spam?
   4.2 Purpose of Spam
   4.3 Spam Life Cycle
   4.4 Types of Spam Filters
   4.5 Spam Filters Properties
   4.6 Bayesian Classification
   4.7 Computing the Probability
   4.8 How to Design a Spam Filtering System with Machine Learning Algorithm
5. Linear Regression and k-NN for filtering spam
6. Naive Bayes
7. Data Wrangling

8. Feature Generation
   8.1 INTRODUCTION
   8.2 BACKGROUND
   8.3 SYSTEM AND METHODS
9. Feature Selection algorithms
   9.1 The Problem the Feature Selection Solves
   9.2 Feature Selection Algorithms
   9.3 How to Choose a Feature Selection Method for Machine Learning

UNIT- III Mining Social-Network Graphs & Data Visualization

1. What is Social networks as graphs
2. Clustering of graphs
   a. Graphic Clustering Methods
   b. What kinds of cuts are perfect for drawing clusters in graphs?
3. Direct discovery of communities in graphs
4. Partitioning of graphs
5. Neighborhood properties in graphs
6. Data Visualization, Basic principles
   a. Data Visualization
   b. History of Data Visualization
   c. Why is data visualization important?
   d. What makes data visualization effective?
   e. Five types of Big Data Visualization groups
7. Common data visualization tools
8. Examples of inspiring (industry) projects
9. Data Science and Ethical Issues

Unit I- Introduction of Data Science & R Programming

1. INTRODUCTION: WHAT IS DATA SCIENCE?

What is Data Science?

Data science is an interdisciplinary field that uses scientific methods, processes, algorithms and systems to extract knowledge and insights from data in various forms, both structured and unstructured, similar to data mining.

Why is Data Science?

• Because you have too many data such as money, reviews, customer data, people working, etc.
• You want to keep it clear and easy to understand so you can make a change that's why data science is relevant.
• Data analysis lets people make better decisions, either faster or better.

Why Data Science is important?

Every company, however, has information, and its business value depends on how much information it thinks.

Since late, Information Science has acquired significance in the light of the fact that it can assist companies with growing business estimation of their accessible knowledge and thus allow them to take the upper hand against their rivals.

It can help us know our customers better, it can help us refine our processes and it can help us make better decisions. Knowledge, in the light of information technology, has become a vital instrument.

Role of Data Scientist

• Data scientists help organizations understand and handle data, and address complex problems using knowledge from a range of technology niches.

• They are typically built in the fields of computer science, modeling, statistics, analytics and mathematics, coupled with modeling statistics and mathematics combined with a clear business sense.

How to do Data Science?

A typical data science process looks like this, which can be modified for specific use case:

- Understand the business
- Collect & explore the data
- Prepare & process the data
- Build & validate the models
- Deploy & monitor the performance

Tools for Data Science

1. R
2. Python
3. SQL
4. Hadoop
5. Tableau
6. Weka

Applications of Data Science

1. Data Science in Healthcare
2. Data Science in E-commerce
3. Data Science in Manufacturing
4. Data Science as Conversational Agents
5. Data Science in Transport

Lifecycle of Data Science

A brief overview of the main phases of the Data Science Lifecycle is shown in Figure 1:

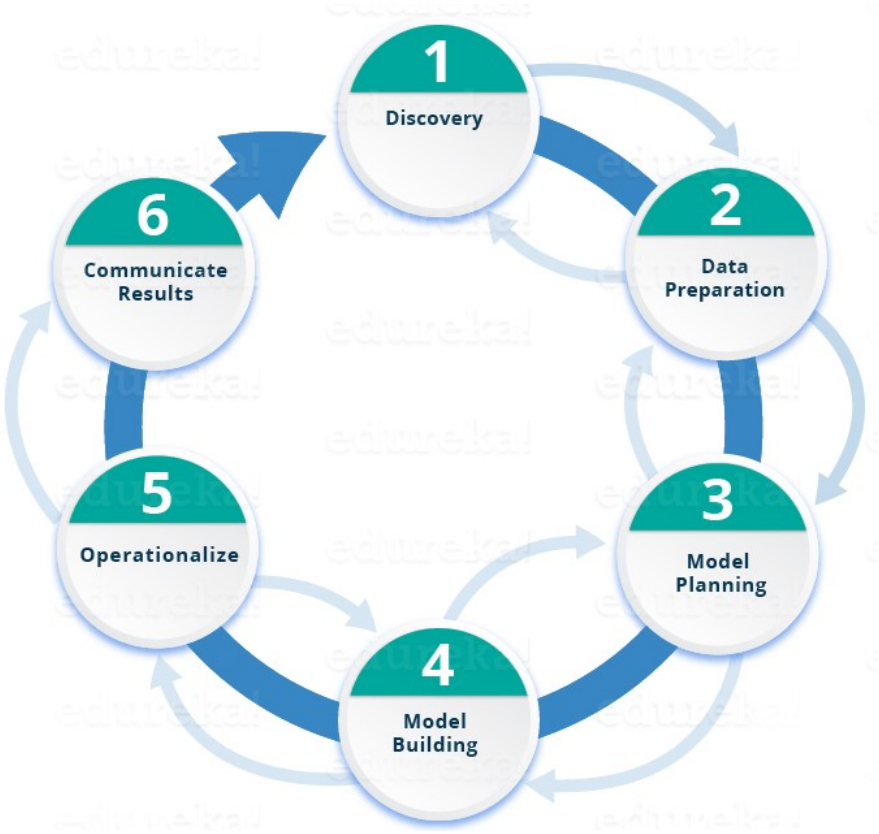


Figure 1: Data Science Lifecycle

Phase 1—Discovery: Phase 1 — It's important to understand the various criteria, requirements, goals and necessary budget before you start the project. You ought to have the courage to ask the

right questions. Here, you determine whether you have the resources needed for supporting the project in terms of people, equipment, time and data. You must also frame the business problem in this process, and formulate initial hypotheses (IH) for testing.

Phase 2—Data preparation: You need analytical sandbox in this process, in which you can conduct analytics for the entire duration of the project. Before modeling, you need to search, preprocess, and condition data. In addition, you must perform ETLT (extracting, converting, loading, and converting) to bring data into the sandbox. Let's look at the flow of statistic analysis in Figure 2 below.

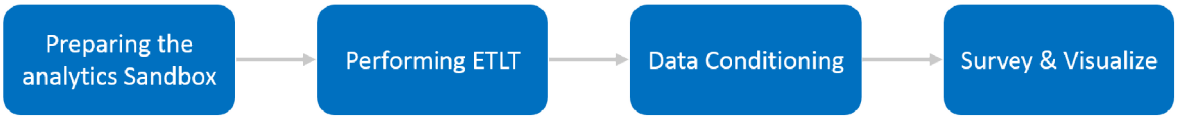


Figure 2: Statistical Analysis flow of Data preperation

R may be used for data cleaning, retrieval, and visualization. It will help you identify the outliers and create a relationship between the variables. When the data has been cleaned and packed, it is time to do some exploratory analytics on it. Let's see if you can get this done.

Step 3 — Model planning: You can decide the methods and strategies for drawing the relationship between variables here. These relationships will set the basis for the algorithms you will be implementing over the next step. Using through statistical formulas and visualization tools, you'll apply Exploratory Data Analytics (EDA).

Let’s have a look at various model planning tools in Figure 3.

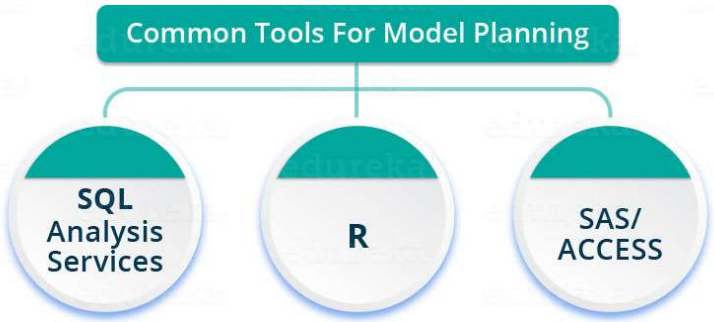


Figure 3: Common Tools for Model planning

1. R has a full range of modeling capabilities and offers a strong setting for interpretive model building.

2. SQL Analysis services can use can data mining functions and simple predictive models to perform in-database analytics.

3. SAS / ACCESS can be used to access Hadoop data, and is used to construct repeatable and reusable flow diagrams for model.

While there are many tools on the market but R is the most commonly used tool.

Now that you have insights into the essence of your results, and you have chosen to use the algorithms. You will apply the algorithm in the next step, and you will create a model.

Phase 4—Model building: You will be designing data sets for training and testing purposes during this process. You should decide whether your current resources are adequate to run the models, or whether you need a more robust environment (such as fast and parallel processing). To construct the model, you'll examine different learning strategies such as grouping, association, and clustering.

Model building can be done using the following methods shown in Figure 4.

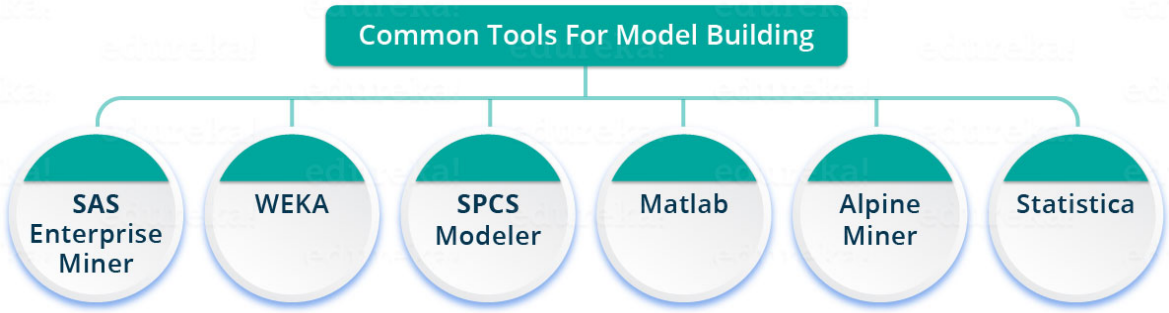


Figure 4: Common Tools for Model Building.

Step 5 — Operationalize: You provide final reports, presentations, code and technical documents during this process. Alternatively, a pilot project is often often applied in an area of real-time output. It will give you a simple, small-scale image of the results and other relevant constraints before full deployment.

Phase 6—Communicating results: Now it's necessary to determine whether you were able to accomplish your target that you designed in the first step. Therefore, in the last step, you define all the main outcomes, communicate to the stakeholders and decide if the project's results are a success or a failure based on the Step 1 criteria.

2. BIG DATA AND DATA SCIENCE HYPE

What does that mean by "Big Data?"

Huge Data is a term used to describe a gigantic amount of both organized and unstructured knowledge that is so immense that using traditional database and programming methods is impossible to handle. The amount of knowledge is too high in most undertaking situations or it moves too quickly, or it reaches the existing planning cap.

(1) Data collection by significant quantities a. Via machines, sensors, men, events.

(2) To do something about it. Decision taking, testing observations, gaining perspective, forecasting the future.

Types of Big Data

There are three types of data behind Big Data- structured, semi-structured, and unstructured shown in Figure 5. There's a lot of useful knowledge in each category that you can mine to use in various projects.

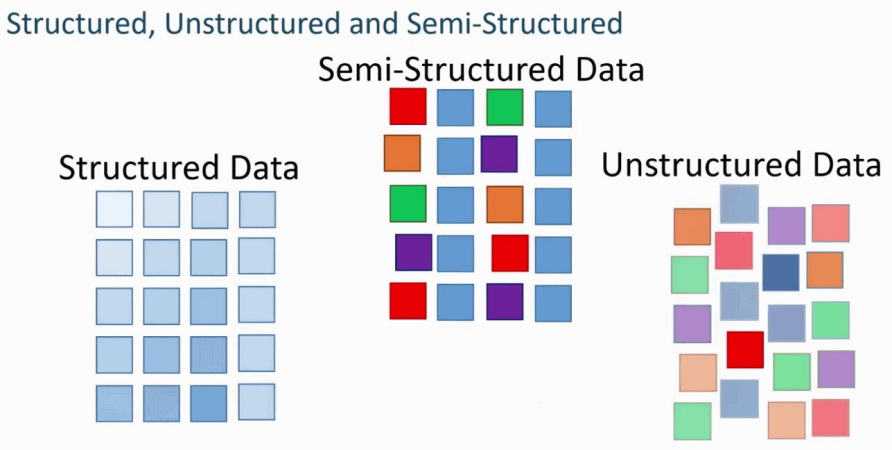


Figure 5: Types of Big data

- **Structured**

  By structured data, we mean data that can be processed, stored, and retrieved in a fixed format. It refers to highly organized information that can be readily and seamlessly stored and accessed from a database by simple search engine algorithms. **For instance, the employee table in a company database will be structured as the employee details, their job positions, their salaries, etc.,** will be present in an organized manner.

- **Unstructured**

  Unstructured information alludes to the information that does not have a particular structure or structure at all. This makes it exceptionally troublesome and tedious to process and investigate unstructured information. Email is a case of unstructured information.

- **Semi-organized**

  Semi-organized information relates to the information containing both the arrangements referenced over, that is, organized and unstructured information. To be exact, it alludes to the information that in spite of the fact that has not been grouped under a specific storehouse (database), yet contains essential data or labels that isolate singular components inside the information.

Big Data also requires several sources and often comes often from different types. But it is not always an easy job to know how to combine all of the tools you need to work with different styles.

Three Characteristics of Big Data-V3c

1. Volume - Data quantity
2. Velocity - Data Speed
3. Variety- Data Types

1. Volume

A run of the mill PC may have had 10 gigabytes of storage in 2000. Today, Facebook is regularly ingesting 500 terabytes of new information. Boeing 737 can generate 240 terabytes of flight information on a single trip across the US. The advanced cell phones, the information they generate and expend; sensors implanted into ordinary items can take care of containing human, field, and other data, including video, in billions of new, continually refreshed information, before long outcome.

2. Velocity

• Clickstreams and ad impressions capture consumer activity at millions of events per second • High-frequency stock trading algorithms represent market movements within microseconds • machine-to-machine processes share data between billions of devices • networks and sensors produce huge real-time log data • online gaming systems help millions of competitor users, each pro

3. Variety

• Big data are not just numbers, dates, and strings. Huge data is also geospatial information, 3D information, sound and video, and unstructured content, including web-based log documents.

• Modern database systems were built to handle smaller structured information volumes, fewer changes or an expected, steady information structure. • Big Data inquiry involves details of different kinds

Benefits of Big Data

Capacity to process Big Data gets different advantages, for example,

1. Organizations can use outside insight while taking choices

Access to social information from web indexes and locales like facebook, twitter are empowering associations to calibrate their business procedures.

2. Improved client assistance

Customary client criticism frameworks are getting supplanted by new frameworks structured with Big Data advances. In these new frameworks, Big Data and common language handling innovations are being utilized to peruse and assess buyer reactions.

3. Early recognizable proof of hazard to the item/administrations, assuming any

4. Better operational effectiveness

Huge Data innovations can be utilized for making an arranging territory or landing zone for new information before recognizing what information ought to be moved to the information distribution center. Also, such coordination of Big Data advances and information distribution center encourages an association to offload inconsistently got to information.

Big Data Tools for Data Analysis

1) Apache Hadoop
2) CDH **(Cloudera Distribution for Hadoop)**
3) Cassandra
4) Knime
5) Datawrapper
6) MongoDB
7) Lumify
8) HPCC
9) Storm
10) Apache SAMOA
11) Talend
12) Rapidminer
13) Qubole
14) Tableau
15) R

Big Data Techniques

Six big data analysis techniques

Big data is defined by the three V's: the vast amount of data, the pace in which it is processed, and the broad variety of data.7 Due to the second descriptor, the pace, data analytics has extended into the technical fields of machine learning and artificial intelligence. In addition to the emerging computer-based analytical methods, data harnesses are now used to analyze t.

In addition to the emerging data harnesses of computer-driven research techniques, analyzes are often focused on conventional statistical methods.9 Essentially, how data analysis techniques operate within an enterprise is doubled; broad data analysis is generated by streaming data as it appears, and then conducting batch analysis of data as it grows – to search for behavioral patterns and trends.

When data becomes more informative in its size, scope and depth, the more creativity it drives.

1. A/B testing

2. Data fusion and data integration

3. Data mining

4. Machine learning

5. Natural language processing (NLP).

6. Statistics.

Underfitting and Overfitting

Machine learning uses data to create a "model" and uses model to make predictions

- Customers who are women over age 20 are likely to respond to an advertisement
- Students with good grades are predicted to do well on the SAT
- The temperature of a city can be estimated as the average of its nearby cities, unless some of the cities are on the coast or in the mountains

- **Underfitting**
  Model used for predictions is too simplistic
  - 60% of men and 70% of women responded to an advertisement, therefore all future ads should go to women
  - If a furniture item has four legs and a flat top it is a dining room table
  - The temperature of a city can be estimated as the average of its nearby cities, unless some of the cities are on the coast or in the mountains

- **Overfitting**
  Model used for predictions is too specific
- The best targets for an advertisement are married women between 25 and 27 years with short black hair, one child, and one pet dog
- If a furniture item has four 100 cm legs with decoration and a flat polished wooden top with rounded edges then it is a dining room table

Data Science Hype

The noise around AI, data science, machine learning and profound learning is hitting a level of fever. Our industry has experienced a difference in what people mean when they say "AI," "machine learning" or "data science" as this noise has evolved. It can be argued that a growing taxonomy is missing for our industry. If there is taxonomy then we have not done a very good job of adhering to it, as data science professionals. This would have consequences. Two implications include creating a hype-bubble that leads to unrealistic expectations and an growing inability to interact, especially with colleagues from non-data sciences. I will cover succinct concepts in this post and then argue how important it is.

Concise Definitions

Data Science: a discipline that produces predictions and explanations using code and computer to create models that are put into action.

Machine Learning: a class of algorithms or techniques to capture complex data patterns in model form automatically.

Deep learning: A class of machine learning algorithms that uses more than one hidden layer of neural networks.

AI: a group of systems functioning in a manner comparable to humans in both degree of autonomy and reach.

Hype

There is a lot of star strength in these words. We encourage people to dream and envision a better future leading to their unnecessary use. More buzz around our industry is elevating the tide that raises all sails, no? Sure, we all hope the tide keeps rising. Yet, if it bursts, we will aim for a sustainable rise and stop a publicity bubble that would cause widespread disillusionment.

Numerous leaders are requesting guidance on how to assist executives, mid-level managers and even emerging data scientists have reasonable standards of data science initiatives without losing data science excitement. Unrealistic expectations delay development by swelling excitement when projects yield less than utopian results

A major cause of this hype has been the constant overuse of "AI" when referring to any solution which allows some kind of prediction. Owing to constant overuse, people automatically equate data science ventures with near-perfect autonomous human-like solutions. Or, at the very least, people believe that data science can easily solve their particular predictive need, without questioning whether their organizational data supports such a concept.

Communication Inappropriate words are often used to clutter up conversations. It can be especially detrimental when a cross-functional team assembles to communicate priorities and develop the end solution in the early planning phases of a data science project.

I know a data science manager who demands that his data sciences team be practically locked with business executives in a room for an hour before he approves any new data science project. Okay, the door isn't actually closed, but it's shut, and for a full hour he wants them to discuss the project. They saw a reduction in the rework of the project, as they concentrated on early communication with business stakeholders. The difficulty of describing principles related to data science is as challenging as it is. We only make it more complicated if we cannot describe our own words.

Because AI and deep learning have come onto the scene, conversations have to be interrupted and questions answered to figure out what people are.

- Because AI and deep learning have come onto the scene, discussions are constantly required to pause and ask questions and figure out what people actually mean by using those words. For starters, how would you interpret those conversation-based statements?
- "Our goal is to make our technology AI-driven within 5 years." • "We need to improve machine learning before we invest in deep learning." • "We use AI to predict fraud so that our customers can spend with confidence." • "Our research showed that AI-investing organizations had a 10 percent increase in revenue." The most common term misunderstanding is when someone speaks about AI solutions, or when they do AI, whatever they do.

The most common term-confusion is when someone talks about AI solutions, or when they do AI, when they should actually talk about creating a model of deep learning or machine learning. It seems like the exchange of words is all too frequently deliberate, with the speaker hoping to get a hype-boost by saying "AI." Let's dive through each of the meanings, and see if we can find a taxonomy agreement.

Data Science

First of all, like every other academic discipline I see data science as a technical discipline. Take Biology for example. Biology requires a variety of concepts, theories, processes, and instruments. Experimentation is normal. The bio-research group continuously contributes to the knowledge base of the discipline. It is no different from data science. Practitioners do science of the results. Scientists are moving the field forward with new hypotheses, principles and methods.

The data science activity includes the marriage of code (usually some mathematical programming language) with data for model building. This involves the initial essential and dominant steps of obtaining, cleaning, and preparing data. Models of data science generally make predictions (e.g., predicting loan risk, predicting diagnosis of disease, predicting how to respond to a conversation, predicting what objects are in an image). Models of data science may also explain or characterize the environment (e.g. whic) for us.

Data science models may also illustrate or define the environment for us (e.g., which combination of variables is most important in making a diagnosis of the disease, which consumers are most similar and how). Eventually, when applied to new data, these models are put into action for making predictions and explications. Data science is a discipline that produces predictions and explanations using code and data to create models that are put into action.
A description for data science can be difficult to formulate while at the same time separating it from statistical analysis. Via educational training in math and statistics as well as professional experience as a statistician I came to the data sciences profession. I used to do data science like many of you before it became a thing.

Statistical analysis is focused on samples, experimental conditions, probabilities and distributions. It typically refers to questions increasing the probability of events or the validity of claims. It uses various algorithms such as t-test, chi-square, ANOVA, DOE, surface designs for the answer, etc. Often these algorithms create models too. For example, surface response designs are techniques for estimating a physical system's polynomial model based on observed explanatory factors, and how they contribute to the response factor.

In my interpretation, one important point is that data science models are applied to new data in order to make future predictions and explanations, or "put into development." Although it is true that surface-response models can be used to predict a response on new data, it is typically a hypothetical prediction of what would happen if the inputs were modified. The engineers then adjust the inputs and analyze the responses the physical device produces in its new environment. The surface model of the reaction is not put into development. This doesn't take the thousands of new input settings, in batches or streams over time, and predicts responses.

This concept of data science is by no means foolproof but it starts capturing the essence of data science by bringing predictive and descriptive models into action.

Machine Learning

Machine learning as a word reigns in the 1950s. Today, data scientists see it as a collection of techniques used within data science. It is a tool collection, or a class of techniques used to construct the above mentioned models. Machine learning helps computers to create (or learn) models on their own, rather than a person directly articulating the reasoning for a model. This is achieved by analyzing an initial collection of data, finding complex hidden patterns in that data and storing those

patterns in a model so that they can be later applied to new data for predictions or interpretations to be made.
The magic behind this automated pattern-discovery process lies in the algorithms. Algorithms are Machine Learning Workhorses. Popular machine learning algorithms include the various approaches to the neural network, clustering strategies, gradient boosting machines, random forests and much more. If data science is a discipline like biology, then microscopy or genetic engineering is like machine learning. It is a set of methods and techniques which is used to exercise the discipline.

Deep Learning

Deep learning is the simplest interpretation of those concepts. Deep learning is a class of machine learning algorithms that employs more than one hidden layer of neural networks. The own neural networks date back to the 1950s. Recently, deep learning algorithms were very popular beginning in the 1980s, with a lull in the 1990s and 2000s, followed by resurgence in our decade due to fairly minor changes in how deep networks were designed that proved to have incredible impacts. Deep learning can be applied to a wide range of applications, including image recognition, chat assistants, and recommender systems. Google Voice, Google Images and Google Search for example are some of the roots. For example, Google Speech, Google Photos, and Google Search are some of the original solutions built using deep learning.

AI

AI has been around for a long time. Long before the recent hype storm that has co-opted it with buzzwords. How do we, as data scientists, define it? When and how should we use it? What is AI to us? Honestly, it's not sure anyone really knows. This might be our "emperor has no clothes" moment. We have the ambiguity and the resulting hype that comes from the promise of something new and unknown. The CEO of a well-known data science company was recently talking with our team at Domino when he mentioned "AI". He immediately caught himself and said, "I know that doesn't really mean anything. I just had to start using it because everyone is talking about it. I resisted for a long time but finally gave in."

That said, I'm going to take a stab on it: AI is a category of systems that people aim to build that have the distinguishing characteristic that in the degree of autonomy and scope of activity they will be comparable with humans.

If data science is like biology and machine learning is like genetic engineering, and then AI is like resistance to disease, to expand our analogy. It's the end product, a series of solutions or structures we seek to build by applying machine learning (often deep learning) and other techniques.

Here is the concluding line. I think we need to distinguish between strategies that are part of AI solutions, AI-like solutions and actual AI solutions. This includes AI building blocks, solutions with AI-ish qualities, and solutions that approach human autonomy and scope. These are three separate things. People just say "AI" for all three far too often.

For example,

- Deep learning is not AI. It is a technique that can be used as part of an AI solution.
- Most data science projects are not AI solutions. A customer churn model is not an AI solution, no matter if it used deep learning or logistic regression.
- A self-driving car is an AI solution. It is a solution that operates with complexity and autonomy that approaches what humans are capable of doing.

Gartner's 2019 Hype Cycle

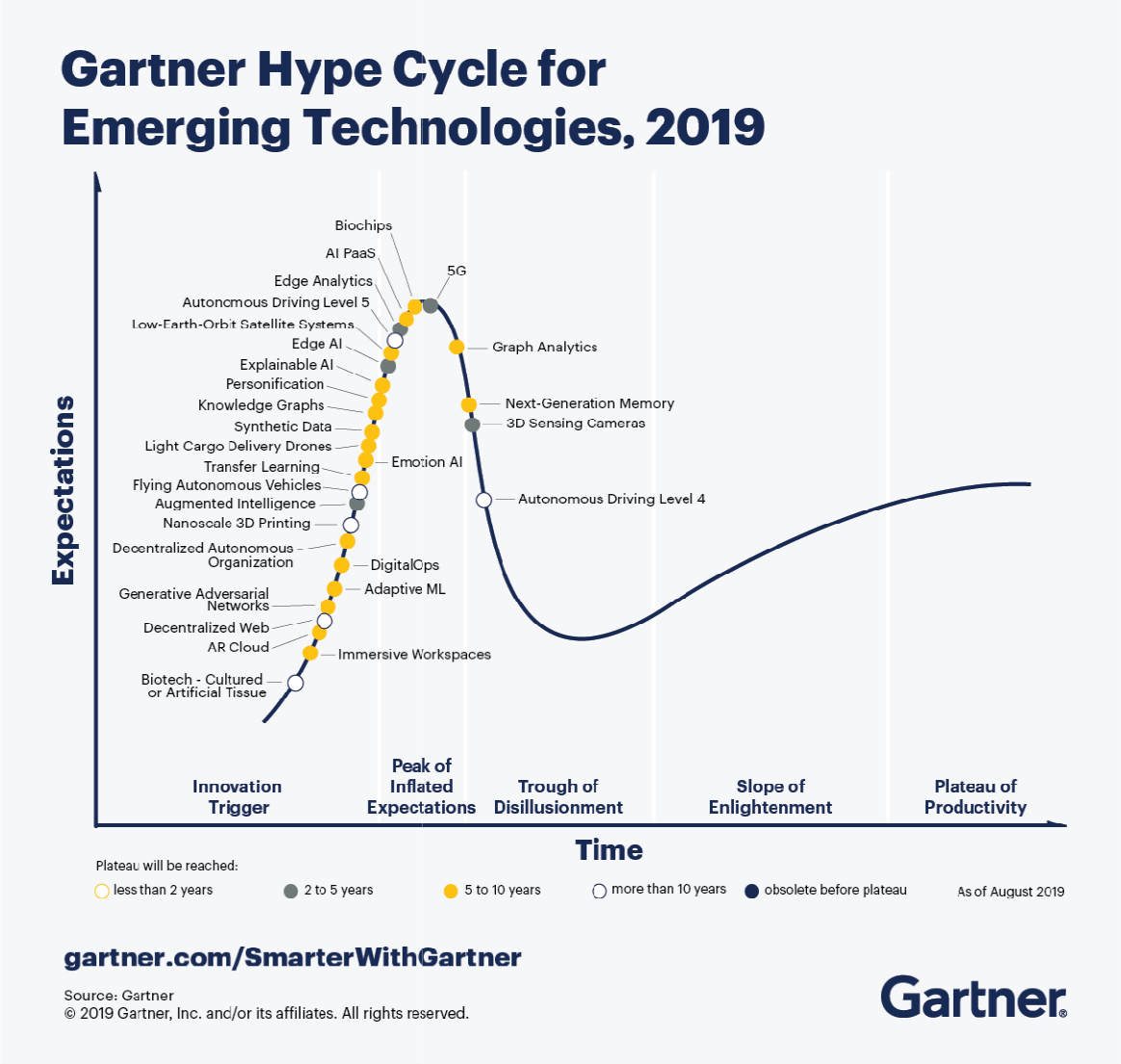


Figure 6: **Gartner's Hype Cycle**

Gartner's 2019 Hype Cycle for Emerging Technologies is out shown in Figure 6, so it is a good moment to take a deep look at the report and **reflect on our AI strategy as a company**.

First and foremost, and before going into depth about the report's content and its consequences for the AI strategy of companies, I would like to answer a very recurring message that I have seen in social networks these last days. Several people were shocked to see those developments totally missing from the study despite appearing during previous years. As Gartner states in his study, his Hype Cycle encompasses a wide variety of subjects, and if a specific technology is not highlighted, it does not automatically mean that they are not relevant, quite the opposite. Some technologies that disappear from the Hype Cycle due to the fact that they are no longer "emerging," but essential to **business and IT.**

So let's start by actually reviewing the fundamental AI related technologies that were omitted from the report this year but are still important for business:

- **Deep neural networks (DNNs).** Gartner also speaks about DNNs as a basis for many other new technologies used in the Hype Cycle.
- **Conversation AI platforms**. Gartner no longer considers new technologies to Conversational AI systems, though focusing on their importance to market.
- **Digital Helpers**. Gartner no longer considers new technologies as virtual assistants, though focusing on their importance as industry.
- **Artificial General Intelligence (AGI)**. In my opinion, a good call by Gartner to favor a pragmatic vision around AI, moving away from hype. As Gartner mentions, AGI will not be mature for decades.

According to Gartner, what areas of focus will be the AI leaders for companies? Based on the 2019 Emerging Technologies Priority List, those will be:

• Augmented Intelligence. Gartner recognizes this evolving technology as the key to the design strategy of new business technologies, combining short-term automation with a mid-/long-term strategy that ensures quality enhancement not only by means of automation, but also through growing human talent.

• Edge AI. In those situations where contact costs, latency or high volume ingestion can be crucial. This, of course, means ensuring that the correct AI technologies and techniques are available for our use case (e.g. Deep Learning) are available for the IoT infrastructure we want to deploy, among other conditions.

Finally, which are the new emerging technologies related to AI in 2019 Hype Cycle**:**

- Adaptive ML
- Emotion AI.
- Explainable AI.
- Generative Adversarial Networks (GANs).
- Transfer Learning.

3. STATISTICAL INFERENCE, STATISTICAL MODELING

STATISTICAL INFERENCE

Inferential Statistics

Inferential statistics allows you to make inferences about the *population* from the *sample* data shown in Figure 7.

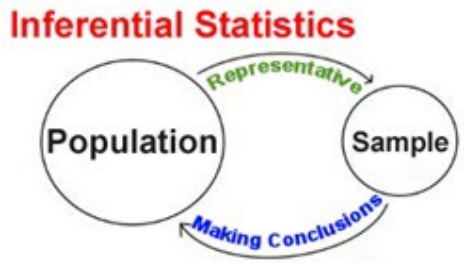


Figure 7: Inferential statistics

Population & Sample

A sample of a population is a representative subset. In certain cases carrying out a population census is an ideal yet unrealistic solution. Sampling is much more practical though sampling error is vulnerable. A sample that is not population representative is called bias, the approach chosen for such sampling is called sampling bias. The key forms of sampling bias are comfort bias, judgment bias, size bias, response bias. Randomisation is the best method to reduce bias in the sampling. Simple random sampling is the simplest randomisation method; other systematic sampling techniques are cluster sampling & stratified sampling.

Sampling Distributions

Sample means being spread more and more naturally around the true mean (the population parameter), as we increase our sample size. Sample variation means decreasing as sample size increases.

Central Limit Theorem

The Central Limit Theorem is used to help us understand the following facts, whether or not the distribution of populations is normal:

1. The mean of the sample is the same as mean of population
2. The default sample deviation means is always equal to the standard error.
3. Sample distribution means will become ever more natural as the sample size increases.

Confidence Intervals

A *sample mean* can be referred to as a *point estimate* of a *population mean*. A *confidence interval* is always cantered around the mean of your sample. To construct the interval, you add a *margin of error*. The *margin of error* is found by multiplying the *standard error* of the mean by the *z-score* of the percent confidence level shown in Figure 8:

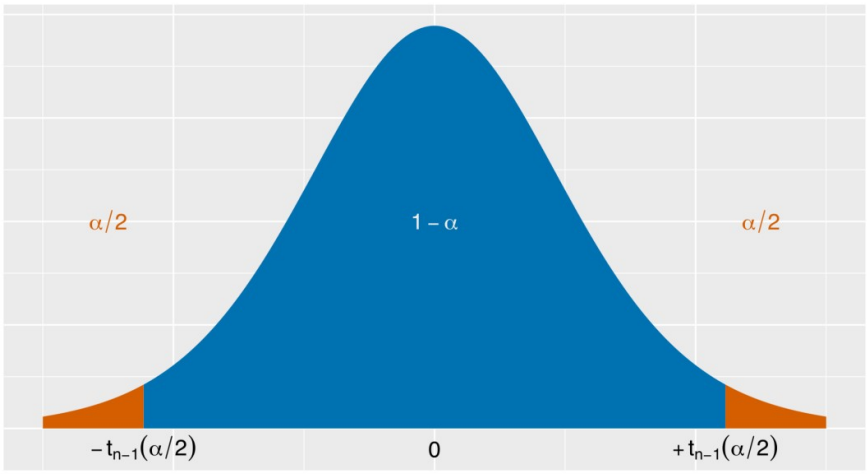
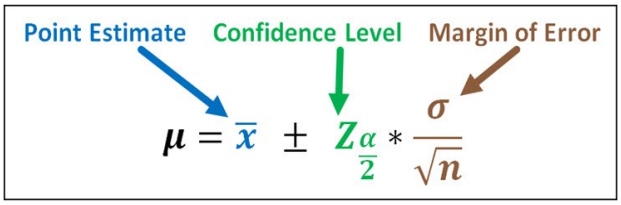


Figure 8: Confidence intervals graph



The confidence level represents the number of times out of 100 that the population average would be within the specified sample mean interval.

Hypothesis Testing

Hypothesis testing is a kind of statistical inference involving asking a question, collecting data and then analysing what the data informs us on how to precede. The experimental hypothesis is called the null hypothesis and given the H_o symbol. We test the null hypothesis against an alternative hypothesis to which the symbol H_o is assigned shown if Figure 9.

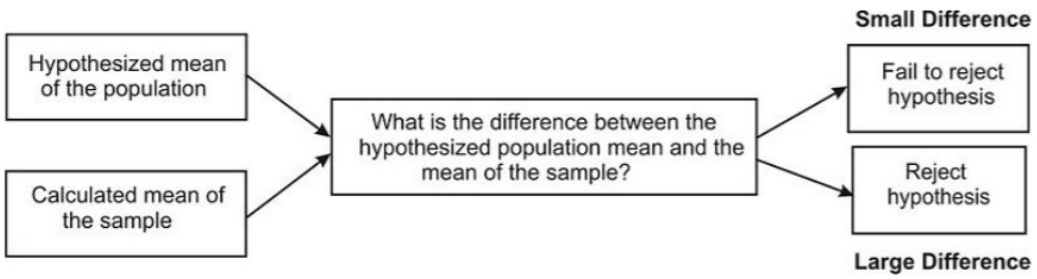


Figure 9: Hypothesis testing

When testing a hypothesis we have to determine how much of a difference between means is required to refute the null hypothesis. For their hypothesis check, Statisticians first select a level of significance or degree of alpha (α).

Decision Made	Null Hypothesis is True	Null Hypothesis is False
Reject Null Hypothesis	Type I Error	Correct Decision
Do not Reject Null Hypothesis	Correct Decision	Type II Error

Values that show the edge of the critical area are important. Critical regions define the entire value area which means you are rejecting the null hypothesis.

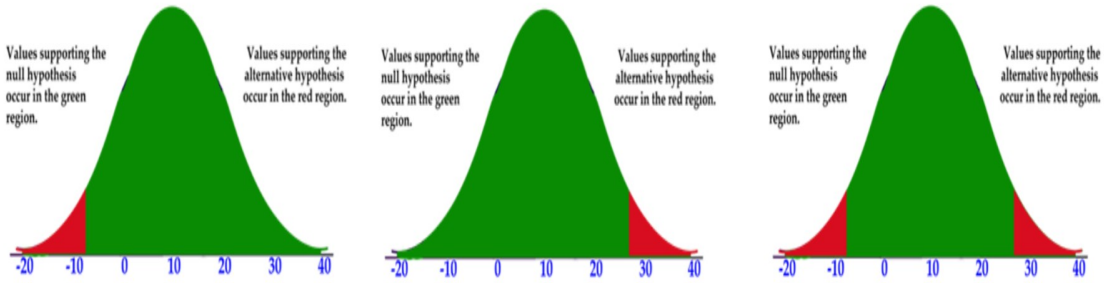


Figure 10: left, right & two-tailed tests

These are the *four basic steps* we follow for (one & two group means) hypothesis testing:

1. State the *null* and *alternative* hypotheses.
2. Select the *appropriate significance level* and check the *test assumptions*.
3. Analyse the data and compute the *test statistic*.
4. Interpret the *result*.

Hypothesis Testing (One and Two Group Means)

Hypothesis Test on One Sample Mean When the Population Parameters are Known

We find the *z-statistic* of our sample mean in the *sampling distribution* and determine if that *z-score* falls within the *critical(rejection) region* or not. This test is *only appropriate when* you know the *true mean* and *standard deviation* of the *population*.

z-Test

Formula to find the value of Z (z-test) Is:

$$Z = \frac{\overline{x} - \mu_0}{\sigma/\sqrt{n}}$$

$\bar{x}$ = mean of sample
μ_0 = mean of population
σ = standard deviation of population
n = no. of observations

Hypothesis Tests When You Don't Know Your Population Parameters

The *Student's t-distribution* is similar to the normal distribution, except it is more spread out and wider in appearance, and has thicker tails. The differences between the *t-distribution* and the *normal distribution* are more exaggerated when there are fewer data points, and therefore *fewer degrees of freedom* shown in Figure 11.

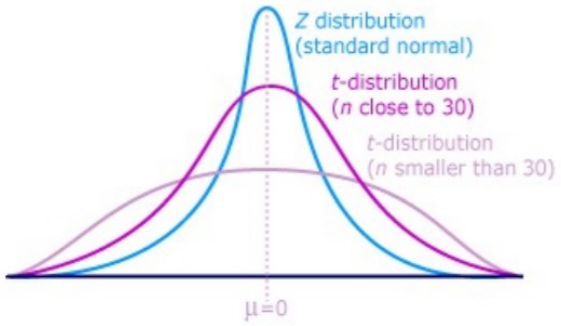


Figure 11: Distribution graph

t-Test

$$t = \frac{\bar{x} - \mu_0}{\frac{s}{\sqrt{n}}}$$

where:

t is the test statistic and has $n - 1$ degrees of freedom.

$\bar{x}$ is the sample mean

μ_0 is the population mean under the null hypothesis.

s is the sample standard deviation

n is the sample size

$\frac{s}{\sqrt{n}}$ is the estimated standard error

Estimation as a follow-up to a Hypothesis Test

When a *hypothesis is rejected*, it is often useful to turn to estimation to try to capture the *true value* of the *population mean*.

Two-Sample T Tests

Independent Vs Dependent Samples

When we have *independent samples* we assume that the scores of one sample *do not affect* the other.

$$t = \frac{(\bar{x}_1 - \bar{x}_2) - (\mu_1 - \mu_2)}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}}$$

Where:

$\bar{x}_1 - \bar{x}_2$ is the difference between the sample means

$\mu_1 - \mu_2$ is the difference between the hypothesized population means

$\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}$ is the standard error of the difference between the sample means

unpaired t-test

In two *dependent samples* of data, each score in one sample *is paired with* a specific score in the other sample.

$$t = \frac{\bar{d} - \delta}{SE_{\bar{d}}}$$

paired t-test

Hypothesis Testing (Categorical Data)

Chi-square test is used for *categorical data* and it can be used to estimate how closely the distribution of a categorical variable matches an expected distribution (the *goodness-of-fit* test), or to estimate whether two categorical variables are independent of one another (the *test of independence*).

$$\chi^2 = \sum \frac{(observed - expected)^2}{expected}$$

goodness-of-fit

degree of freedom (d f) = no. of categories(c)−1

$$expected\ cell\ value = \frac{C \times R}{n}$$

test of independence

degree of freedom (df) = (rows−1)(columns−1)

Hypothesis Testing (More Than Two Group Means)

Analysis of Variance (ANOVA) allows us to test the hypothesis that *multiple population means and variances* of scores are *equal*. We can conduct a series of t-tests instead of ANOVA but that would be tedious due to various factors.

We follow a series of steps to perform ANOVA:

1. Calculate the *total sum of squares* (SST)
2. Calculate the *sum of squares between* (SSB)
3. Find the *sum of squares within groups* (SSW) by subtracting
4. Next solve for *degrees of freedom* for the test
5. Using the values, you can now calculate the *Mean Squares Between* (MSB) and *Mean Squares Within* (MSW) using the relationships below
6. Finally, calculate the *F statistic* using the following ratio
7. It is easy to fill in the Table from here — and also to see that once the SS and df are filled in, the remaining values in the table for MS and F are simple calculations
8. Find *F critical*

$SS_T = \sum(y - \bar{y}^2) = \sum y^2 - \frac{(\sum y)^2}{N}$

Where:
y = each observation
N = total number of scores
$\bar{y}$ = grand mean (mean of all scores)

$SS_B = \sum n_k (\bar{y}_k - \bar{y})^2$

Where:
k = the number of groups
n_k = the number of scores in group k
$\bar{y}_k$ = the mean of group k

$SS_W = SS_T - SS_B$

$df_{Total} = N - 1$
$df_{Between} = k - 1$
$df_{Within} = N - k$

$MS_B = \frac{SS_B}{df_{Between}}$

$MS_W = \frac{SS_W}{df_{Within}}$

$F = \frac{MS_B}{MS_W}$

ANOVA formulas

If *F-value* from the ANOVA test is greater than the *F-critical value*, so we would reject our Null Hypothesis.

One-Way ANOVA

One-way ANOVA method is the procedure for testing the null hypothesis that the population means and variances of *a single independent variable* are equal.

Two-Way ANOVA

Two-way ANOVA method is the procedure for testing the null hypothesis that the *population means and variances* of *two independent variables* are equal. With this method, we are *not only* able to study the *effect of two independent variables*, but also the *interaction between these variables*.

We can also do two separate one-way ANOVA but two-way ANOVA gives us *Efficiency, Control & Interaction*.

Quantitative Data (Correlation & Regression)

Correlation

Correlation refers to a *mutual relationship* or *association* between quantitative variables. It can help in predicting one quantity from another. It *often* indicates the *presence of a causal relationship*. It used as a basic quantity and *foundation* for many other *modeling techniques*.

$$\rho_{X,\,Y} = \frac{E[(X - \mu_X)(Y - \mu_Y)]}{\sigma_X \sigma_Y}$$

Pearson Correlation

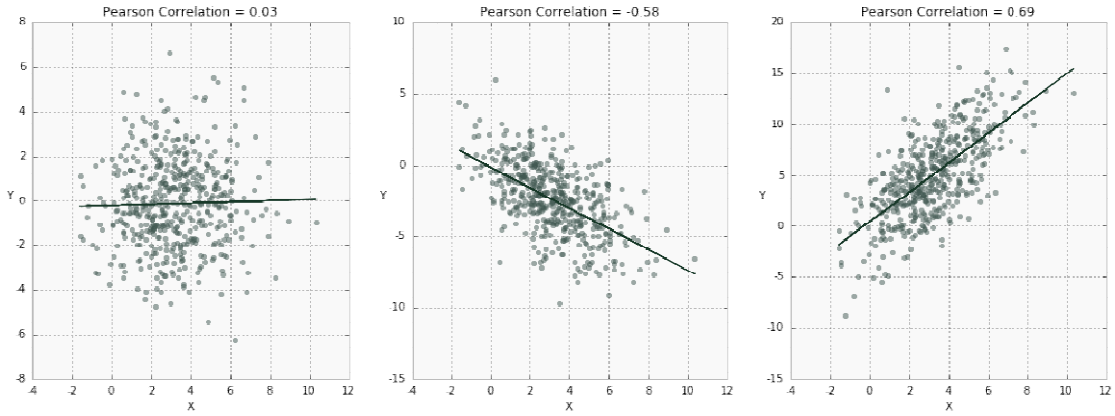


Figure 12: Pearson Correlation

Regression

Regression analysis is a *set of statistical processes* for *estimating the relationships* among variables shown in Figure 13.

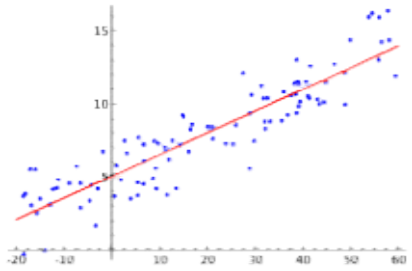


Figure 13: Regression

Simple Regression

This method uses a *single independent variable* to predict a dependent variable *by fitting* the best relationship shown in Figure 13.

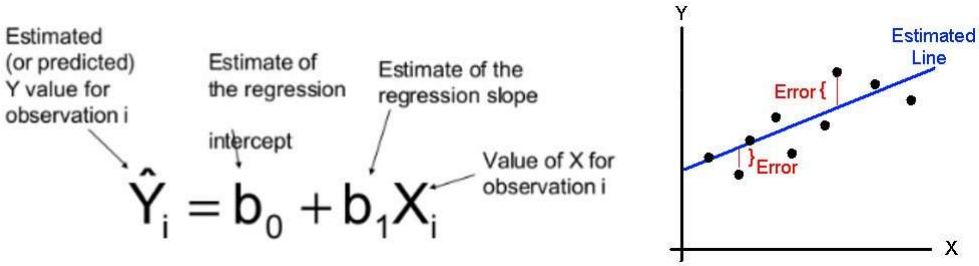


Figure 13: Simple Regression

Multiple Regression

This method uses *more than one independent variable* to predict a dependent variable *by fitting* the best relationship shown in Figure 14.

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_n X_n + \varepsilon$$

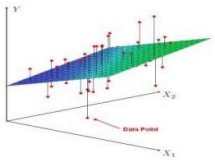


Figure 14: Multiple Regression

It works best when *multicollinearity* is absent. It's a phenomenon in which two or more predictor variables are *highly correlated*.

Nonlinear Regression

In this method, observational data are modelled by a function which is a *nonlinear combination* of the model parameters and depends on *one or more independent variables* shown in Figure 14.

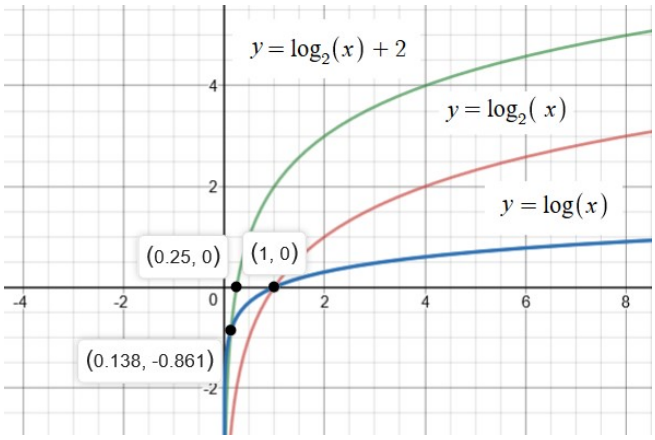


Figure 14: Nonlinear regression

Significance in Data Science

In *data science*, *inferential statistics* is used is many ways:

- Making *inferences* about the *population* from the *sample*.
- Concluding whether a sample is *significantly different* from the population.
- If *adding or removing a feature* from a model will really help to improve the model.
- If one *model is significantly better* than the other?
- *Hypothesis testing* in general.

4. PROBABILITY DISTRIBUTIONS

What is Probability?

Probability is the chance that something will happen — how likely it is that some event will happen.

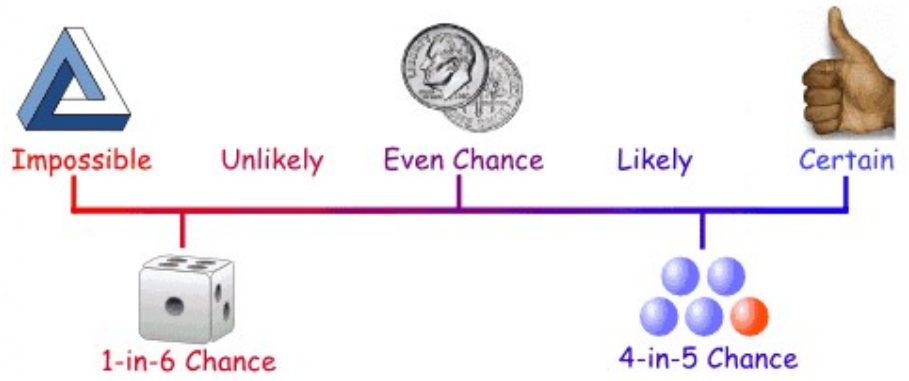


Figure 15: Probability

Probability of an event happening ***P(E)*** *= Number of ways it can happen* ***n(E)****/ Total number of outcomes* ***n(T)***

Probability is the measure of the likelihood that an event will occur. Probability is quantified as a number between 0 and 1, where 0 indicates impossibility and 1 indicates certainty shown in Figure 15.

Why Probability is important?

In certain areas of our everyday lives, confusion and randomness exist and having a strong knowledge of probability allows us to make sense of these uncertainties. Knowing about chance allows us to make educated judgments on what is likely to happen, based on a trend of previously collected data or estimation.

How to use Probability in Data Science?

Data Science also makes use of statistical inferences to forecast or interpret computer patterns, while statistical inferences use data distribution of probabilities. Therefore it is important to know the likelihood and its implementations to work effectively on data science problems.

What is Conditional Probability?

Conditional probability is a measure of the likelihood of an occurrence (some particular circumstance occurring), provided that another occurrence has occurred (by inference, hypothesis, conclusion, or evidence).

$$P(B \mid A) = \frac{P(A \text{ and } B)}{P(A)}$$

The probability of event B provided event A equals the likelihood of event A and event B divided by the likelihood of event A.

How conditional probability is used in data science?

Most techniques in data science (i.e., Naive Bayes) depend on the theorem of Bayes. Bayes' theorem is a formula that explains how, when given proof, to change the probabilities of the hypotheses.

$$P(A \mid B) = \frac{P(B \mid A)\, P(A)}{P(B)}$$

Using the Bayes' theorem, it's possible to build a learner that predicts the probability of the response variable belonging to some class, given a new set of attributes.

$$\text{posterior} = \frac{\text{prior} \times \text{likelihood}}{\text{evidence}}$$

What are random variables?

A random variable is a set of possible values from a random experiment shown in Figure 16.

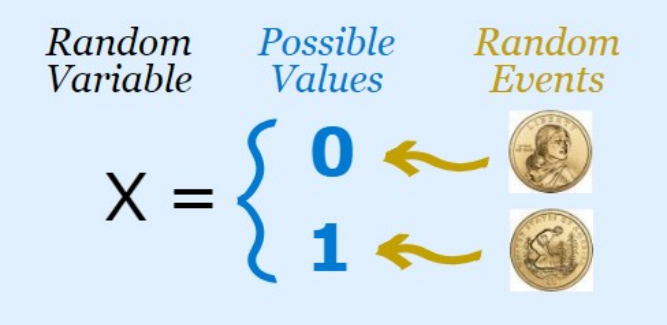


Figure 16: Random variables

A random variable (random quantity, aleatory variable, or stochastic variable) is a variable whose possible values are outcomes of a random phenomenon.

Random variables can be discrete or continuous. *Discrete random variables* can only take certain values while *continuous random variables* can take any value (within a range).

What are probability distributions?

Within a random variable the distribution of probability determines how the probabilities are distributed over the variable random values.

To define the probability distribution, x is used for a discrete random variable, a probability mass function denoted by f(x). This function gives random variable probability for any value.

A continuous random variable is defined for the probability that a continuous random variable will lie within a given interval, since there is an infinite number of values at every interval. Thus here the probability distribution is defined by the probability density function, which is also denoted by f(x).

What are the types of probability distributions?

A binomial distribution is a statistical experiment with the following characteristics: The experiment is made up of n repeated trials. Each trial can produce only two possible results. We call one such outcome a success and the other a loss. The probability of success, denoted by P, for every trial is the same shown in Figure 17.

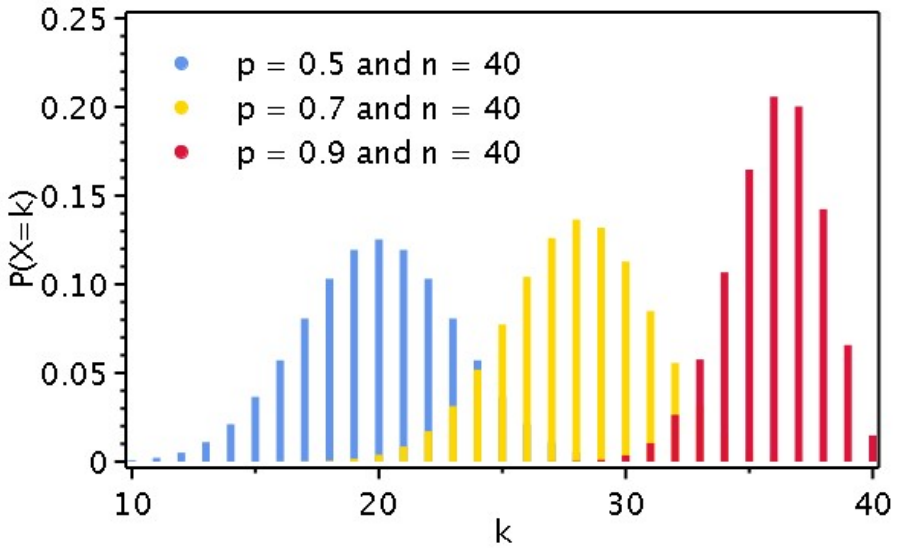


Figure 17: Probability Distributions

Probability of k out of n ways:

$$P(k \text{ out of } n) = \frac{n!}{k!(n-k)!} p^k(1-p)^{(n-k)}$$

The General Binomial Probability Formula

The standard distribution, also known as the Gaussian distribution, is a distribution of probability that is symmetrical around the mean, indicating that the occurrence of data near the mean is more common than that far from the mean.

This has the following characteristics:

• The typical curve is symmetrical around the mean μ;

• The mean is in the middle and splits the area into halves;

• The total area below the curve is equal to 1;

• It is entirely defined by its mean and standard deviation (or variance σ2)

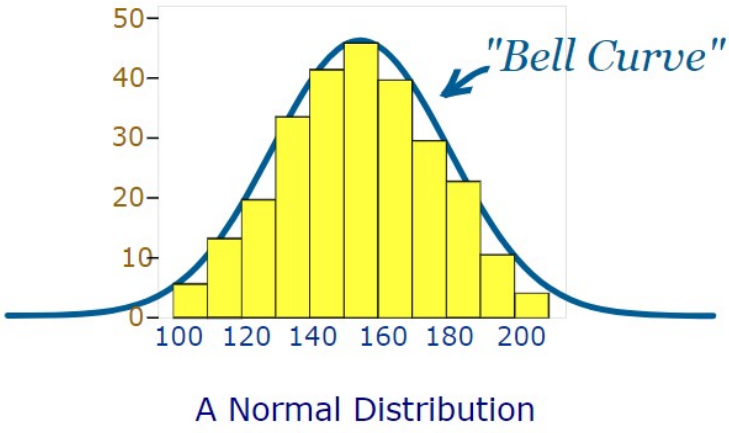


Figure 18: Normal Distribution

How random variables & probability distributions are used in data science?

Data science also makes use of statistical inferences to forecast or interpret computer patterns, while statistical inferences use data distribution of probability. Therefore it is important to know random variables & their distributions of probability to function effectively on data science problems.

5. FITTING A MODEL

Fitting a model means that you are making your algorithm know the relationship between the predictors and the outcome so that you can predict the future value of the outcome.

And the best fitted model has a particular set of parameters that best describes the question at hand.

Objectives of Model Fitting

There are two main goals for model fitting

1. Make inferences about the relationship between variables in a given data set.

2. Predictions/forecasting of potential events, based on models calculated using historical evidence.

Why are we fitting models to data?

1. Estimate the distributional properties of variables, theoretically subject to certain variables.

2. Concisely describe the relationship between the variables and make inferential assumptions about the relationship

3. Predict the values of interest variables on the basis of values of other predictor variables, and define the volatility of predictions.

6. INTRODUCTION TO R

What is R?

R programming language is one that allows statistical computation and is commonly used by data miners and statisticians to analyze data. It was created by Ross Ihaka and Robert Gentleman in 1995 where they derived the word 'R' from the first letters of their names. R is a common choice for statistical computing and graphical techniques in data analytics and in data science.

In its CRAN repository R holds a set of over 10,000 packages. Such products are appealing to specific statistical applications. R may give a steep learning curve for beginners. But while R's syntax can be simple to understand. It is an interactive method used for the implementation of statistical learning. Hence, a. user without knowledge of statistics may not be able to get the best out of R.

Why We Choose R for Data Science?

Data Science has emerged as 21st Century's most popular industry. That is because the need to evaluate and create knowledge from the data is pressing. Industries are converting the raw data into goods supplied with data. To do so, several essential tools are needed to churn the raw data. R is one of the programming languages that provide you with an intensive environment for process analysis, transformation, and knowledge visualization.

To several statisticians who want to get interested in the design of mathematical models to solve complex problems, this is the primary option. R includes a sea of packages attracting all types of disciplines such as astronomy, biology, etc. Though R was originally used for academic purposes, it is now also being used in industry.

R is a technical language used for complex mathematical modeling. Alternatively, R also supports array, matrix, and vector operations. R is renowned for its graphical libraries, which allow users to delineate visual graphs and make them intractable to users. In addition, R helps its users to create using Web applications
In addition to this, **R provides you with several options of advanced data analytics** like the development of prediction models, **machine learning algorithms**, etc. R also provides several packages for image processing.

History of R

• R was conceived by John Chambers at Bell Laboratories in 1976. R was created as an extension, as well as a programming language implementation of S.

• Ross Ihaka and Robert Gentleman developed the R project and released its first version in 1995 and a stable beta version in 2000

R Features

There are essential R features, which we will explore in order to understand R's role in data science shown in Figure 19.

Figure 19: Features of R

1. Open-source – R is an open-source platform that allows you to access and change the code, and even create your own libraries. This is safe to download, as well.

2. A complete language – Although R is commonly regarded as a statistical programming language, it also contains many features of an Object Oriented programming language.

3. Analytical support-With R, through its wide variety of support libraries, you can perform analytical operations. You can clean, arrange, analyze, display your data and construct predictive models, too.

4. Help extensions – R allows developers to write their own libraries and packages as distributed add-ons, and to promote such packages. This makes R a developer-friendly language that allows its users to make changes and updates.

2. Facilitates database connectivity – This consists of a variety of add-on modules linking R to databases such as the RODBC package, the Open DataBase Connectivity Protocol (ODBC) and the ROracle package allowing connectivity with Oracle databases. The programming language R also contains MySQL extensions as RMySQL.

3. Extensive community participation – It has an active community that is further enhanced by R being an open-source programming language. To many this makes R a perfect alternative. R offers various boot camps and seminars around the world.

4. Simple and easy to understand – Although many would argue that R provides the beginners with a steep learning curve, this is because R is a statistical language. To use R to its maximum, you need to have statistical experience. R, however, has a syntax which is easy to understand. That helps you to better recall and appreciate R.

How R is different than Other Technologies

There are certain special features of R programming that make it different compared to other technologies: • Graphical Libraries-R stays ahead of the curve with its elegant graphical libraries. Libraries such as ggplot2,plotfully encourage attractive libraries for well-defined plot creation.

• Availability / Cost – R is free to use, indicating universal accessibility.

• Technology advancement – R supports different advanced tools and features that allow you to create robust statistical models.

• Job Scenario – R is the primary Data Science device, as stated above. With Data Science's exponential growth and growing demand, R has become the world's most in demand programming language today.

• Customer and Public Service Support – You will experience good community service with R.

• Portability – R is extremely compact. For the best results a lot of different programming languages and software frameworks can easily be combined with the R environment.

Applications of R Programming

• R is used in the finance and banking industries for fraud prevention, customer turnover reduction and potential decision taking.

• Bioinformatics is also used to study genetic sequence sequences, to conduct drug discovery and computer neuroscience.

• R is used to discover potential consumers in online ads through social media research. Organizations often use insights from social media to evaluate consumer emotions and enhance their goods.

• E-commerce firms use R to evaluate consumer transactions and their input.

• Production companies use R to evaluate input from customers. Manufacturing companies use R to analyze customer feedback. They often use it to forecast future demand to adjust their output velocities and increase profits.

What is R In Data Science important?

Several of R's essential data science framework features are-

1. R includes many essential data wrangling sets, such as dplyr, purrr, readxl, google sheets, datapasta, jsonlite, tidyquant, tidyr etc.

2. R facilitates robust mathematical modeling. Considering that data science is heavy statistics, R is an ideal method to execute various statistical operations on it.

3. R is an appealing method for various applications in the data sciences, as it offers esthetic visualization software such as ggplot2, scatterplot3D, lattice, high charter etc.

4. R is used widely in the ETL (Extract, Transform, Load) data science applications. This offers an interface to various databases such as SQL and even spreadsheets.

5. Another essential skill of R is to interface and analyze unstructured data with the NoSQL databases. This is particularly useful for applications in Data Science where a collection of data needs to be analyzed.

6. Data scientists may use machine learning algorithms with R to gain insights into future events. Various packages are available such as rpart, CARET, randomForest, and nnet.

What Makes R Suitable For Data Science?

R is the most popular choice for data scientists. Following are some of the key reasons as to why they use R –

1. R has been reliable and useful for many years at academia. R was generally used at the academy for research purposes as it offered various statistical analytical instruments. With advancements in data science and the need to analyze data, R has also become a common option within the industry.

2. R is an ideal resource when it comes to wrangling data. It allows many preprocessed packages to be used which make it much easier to wrangle the data. This is one of the key reasons why in the Data Science culture R is favored.

3. R offers its popular ggplot2 bundle that is best known for its visualizations. Ggplot2 offers esthetic visualizations which are appropriate for all data operations. In addition, ggplot2 provides users with a degree of interactivity so they can understand more clearly the data contained in the visualisation.

4. R includes modules for the machine learning of different operations. If it is boosting, constructing random forests or doing regression and classification, machine learning offers a wide variety of products.

Data Science Companies that Use R

Some of the major data science companies that use R analysis and statistical modeling are shown in Figure 20 –

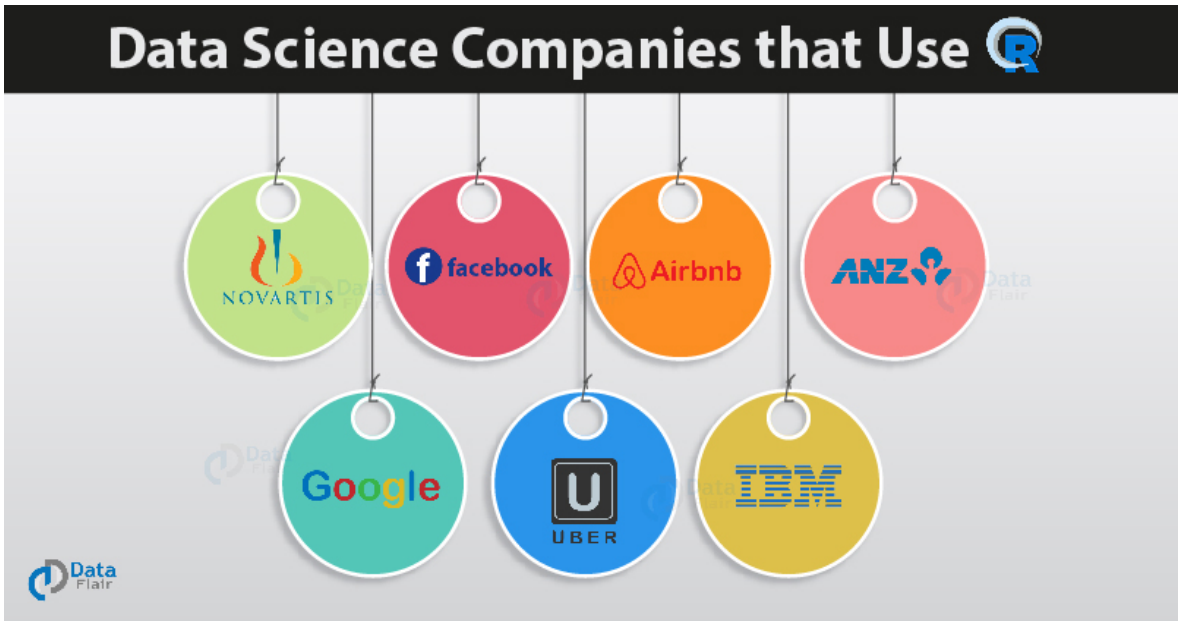


Figure20: Data Science companies that use R

1. Facebook-Facebook uses R extensively for monitoring of social networks. It uses R to gain insights into users actions and create relationships between them.
2. Airbnb – Supports R with its complex day-to-day data operations. R uses the dplyr packet to slic and dictate the data. It also makes use of the ggplot2 graphic package to visualize the results. This also makes use of the pwr kit for different experiments and statistical studies.
3. Uber-Uber makes excellent use of the R kit to navigate its charting components. Shiny is an interactive web application developed with R for interactive graphics embedding.
4. Google – R is a common option at Google for carrying out several analytical operations. The project Google Flu Trends makes use of R to examine flu-related trends and patterns in searches. In addition, Google's predictive API uses R to evaluate the historical data and render possible predictions.
5. ANZ-ANZ is one of Australia's biggest banks. It uses R for credit risk analytics that includes predicting loan defaults based on the clients' transactions and credit score.
6. Novartis – Norvatis is a leading pharmaceutical firm that relies on R for FDA applications for clinical data analysis.
7. IBM-IBM is one of R's largest investors. It joined the consortium in R recently. IBM also makes use of R to develop various analytical solutions. It used R in IBM Watson-an open forum for computing. Furthermore, IBM supports R projects and helps the organization flourish by making some significant contributions.

7. EXPLORATORY DATA ANALYSIS AND THE DATA SCIENCE PROCESS

Exploratory Data Analysis

Exploratory Data Analysis refers to the essential phase of initial data analyses in order to identify patterns, find anomalies, test hypotheses, and use descriptive statistics and graphical representations to verify conclusions.
Second, knowing the data is a good idea, and trying to gain as many ideas from it. EDA is all about making sense of the data in hand, before they get it dirty.

EDA explained using sample data set:

In order to share my interpretation of the definition and techniques I know, I will take an example of the white version of the Wine Quality data set available on the UCI Machine Learning Repository and seek to obtain as many insights from the data set using EDA as possible.

To start with, I imported required libraries (pandas, numpy, matplotlib, and seaborn for this example) and loaded the data set.

Note: Any inferences I've been able to draw, with bullet points I described.

```
In [2]: df = pd.read_csv('winequality-white.csv',sep=';')
        df.head()
```

Out[2]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.0	0.27	0.36	20.7	0.045	45.0	170.0	1.0010	3.00	0.45	8.8	6
1	6.3	0.30	0.34	1.6	0.049	14.0	132.0	0.9940	3.30	0.49	9.5	6
2	8.1	0.28	0.40	6.9	0.050	30.0	97.0	0.9951	3.26	0.44	10.1	6
3	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.9956	3.19	0.40	9.9	6
4	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.9956	3.19	0.40	9.9	6

- Original data is separated by delimiter " ; " in given data set.
- To take a closer look at the data took help of " .head()"function of pandas library which returns first five observations of the data set. Similarly ".tail ()" returns last five observations of the data set.

I found out the total number of rows and columns in the data set using ".shape".

```
In [3]: df.shape
Out[3]: (4898, 12)
```

- Dataset comprises of 4898 observations and 12 characteristics.
- Out of which one is dependent variable and rest 11 are independent variables — physico-chemical characteristics.

It is also a good practice to know the columns and their corresponding data types, along with finding whether they contain null values or not.

```
In [5]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4898 entries, 0 to 4897
Data columns (total 12 columns):
fixed acidity           4898 non-null float64
volatile acidity        4898 non-null float64
citric acid             4898 non-null float64
residual sugar          4898 non-null float64
chlorides               4898 non-null float64
free sulfur dioxide     4898 non-null float64
total sulfur dioxide    4898 non-null float64
density                 4898 non-null float64
pH                      4898 non-null float64
sulphates               4898 non-null float64
alcohol                 4898 non-null float64
quality                 4898 non-null int64
dtypes: float64(11), int64(1)
memory usage: 459.3 KB
```

- Data has only float and integer values.
- No variable column has null/missing values.

The describe() function in pandas is very handy in getting various summary statistics. This function returns the count, mean, standard deviation, minimum and maximum values and the quantiles of the data.

```
In [6]: df.describe()
```

Out[6]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol
count	4898.000000	4898.000000	4898.000000	4898.000000	4898.000000	4898.000000	4898.000000	4898.000000	4898.000000	4898.000000	4898.000000
mean	6.854788	0.278241	0.334192	6.391415	0.045772	35.308085	138.360657	0.994027	3.188267	0.489847	10.514267
std	0.843868	0.100795	0.121020	5.072058	0.021848	17.007137	42.498065	0.002991	0.151001	0.114126	1.230621
min	3.800000	0.080000	0.000000	0.600000	0.009000	2.000000	9.000000	0.987110	2.720000	0.220000	8.000000
25%	6.300000	0.210000	0.270000	1.700000	0.036000	23.000000	108.000000	0.991723	3.090000	0.410000	9.500000
50%	6.800000	0.260000	0.320000	5.200000	0.043000	34.000000	134.000000	0.993740	3.180000	0.470000	10.400000
75%	7.300000	0.320000	0.390000	9.900000	0.050000	46.000000	167.000000	0.996100	3.280000	0.550000	11.400000
max	14.200000	1.100000	1.660000	65.800000	0.346000	289.000000	440.000000	1.038980	3.820000	1.080000	14.200000

- Here as you can notice mean value is less than median value of each column which is represented by 50% (50th percentile) in index column.
- There is notably a large difference between 75th %tile and max values of predictors “residual sugar ”,” free sulfurdioxide”,”totalsulfur dioxide”.
- Thus observations 1 and 2 suggests that there are extreme values-Outliers in our data set.

Few key insights just by looking at dependent variable are as follows:

```
In [7]: df.quality.unique()
```

```
Out[7]: array([6, 5, 7, 8, 4, 3, 9], dtype=int64)
```

- Target variable/Dependent variable is discrete and categorical in nature.
- “quality” score scale ranges from 1 to 10;where 1 being poor and 10 being the best.

- 1,2& 10 Quality ratings are not given by any observation. Only scores obtained are between 3 to 9.

```
In [8]: df.quality.value_counts()

Out[8]: 6    2198
        5    1457
        7     880
        8     175
        4     163
        3      20
        9       5
        Name: quality, dtype: int64
```

- This tells us vote count of each quality score in descending order.
- "quality" has most values concentrated in the categories 5, 6 and 7.
- Only a few observations made for the categories 3 & 9.

Data Science Process

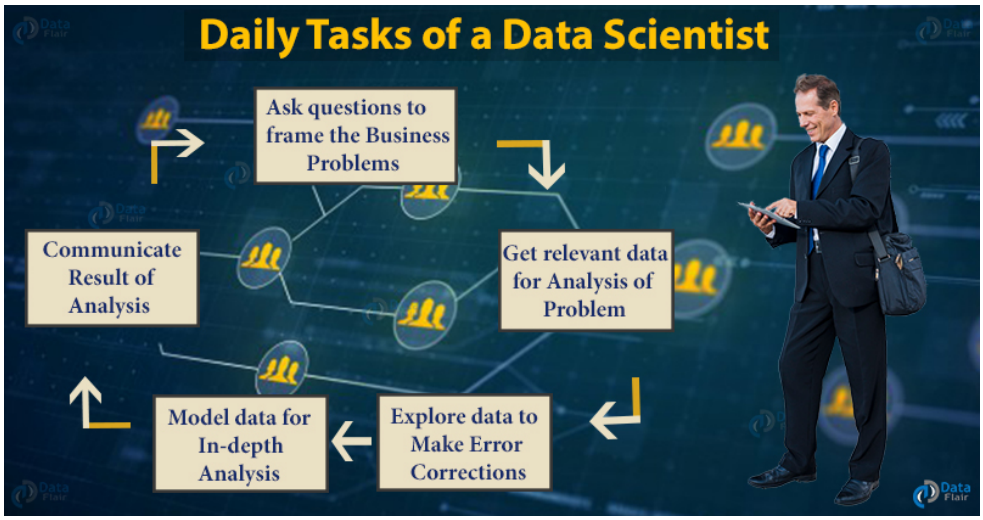
Data Science is a multidimensional discipline that uses scientific methods, techniques and algorithms to derive information and insights from structured and unstructured data. I accept that much of his research is data-related but includes a variety of other data-based processes.

Data Science represents a multidisciplinary field. This includes the systematic blend of scientific and statistical methods, procedures, creation of algorithms, and technology to obtain useful data information.

Yet how do all these dimensions work together? To grasp this, you need to learn the data science process / the day-to-day life of a data scientist

Data Science Process – Daily Tasks of Data Scientist

The steps involved in the complete data science process are:



Step One: Ask Questions to Frame the Business Problem

Seek to get an idea of what a organization wants in the first step, and collect data based on it. You start the data science process by asking the right questions to figure out what the issue is. Let's take a very popular bag company question-the sales issue.

To analyze the issue, you need to begin by asking a lot of questions:

• Who are the target audience and the clients?

• How do you approach the target market?

• How does the selling cycle look like at the moment?

• Which details do you have about the target market?

• How do we classify those clients who are more likely to buy our product?

• You agree to work on the issue after a conversation with the marketing team: "How do we find potential buyers who are more likely to purchase our product? "

• The next move for you is to find out what all the details you have with you to answer the questions above.

Stage two: Get Relevant Data for Problem Analysis

• Now that you are aware of your business concern, it's time to collect the data that will help you solve the problem. Before collecting the data you should ask if the organization already has the correct data available?

• In certain cases, you can get the previously collected data sets from other investigations. The following data are required: age, gender, *previous customer's transaction history, etc.*

You find that most of the customer-related data is available in the company's **Customer Relationship Management (CRM)** software, managed by the sales team.

• SQL database is a rear device with many tables for CRM applications. As you go through the SQL database, you will find out that the system stores detailed customer identification, contact and demographic details (that they gave the company) and their detailed sales process as well.

• If you do not think the available data is adequate, then plans must be made to collect new data. By showing or circulating a feedback form you can also take input from your guests and customers. I agree with that, that's a lot of engineering work and it takes time and effort.

• In addition, the data you obtained is 'raw data' containing errors and missing values. And before the data is analyzed, you need to clean (wrangle) the data.

Step Three: Continue. Explore the Data to Make Error Corrections

Exploring the data cleans and organizes it. This method is focused on more than 70 per cent of the data scientist's time. While gathering all the data, you're not able to use it, because the raw data you've gathered most likely contains odds.

First, you need to make sure that the data is error free and clean. It is the most significant step in the cycle that requires patience and concentration.

For this function specific tools and techniques are used, such as Python, R, SQL, etc.

So you start answering these questions:

• Are the data values missing, i.e. are consumers without their contact numbers?

• Has null values in it? If it happens, how do you fix it?

• Have multiple datasets? Was it a sensible idea to fuse data sets? If so, then how can you bring them together?

When the tests show the missing and false values they are ready for review. Remember that it is easier to get the data incorrect than to have no experience at all.

Stage four. After analyzing the data, you have sufficient knowledge to construct a model to address the question:

"How can we identify potential customers who are more likely to be?"

In this phase, you analyze the data in order to extract information from it. Analyzing the data involves the application of various algorithms which will derive meaning from it:

• Build data model to answer the query.

• Test the model against gathered data.

• Use of different visualization software for presenting data.

• Carry out the algorithms and statistical analysis required.

• Findings align with other methods and sources.

But answering those questions will only give you theories and hints. Modeling data is an easy way to simulate data within a proper equation the machine understands. You will be able to make model-based predictions. You might need to try out multiple models to find the best fit.

Coming back to the issue of sales, this model will help you predict which clients are more likely to buy. The prediction can be specific, like Female, 16-36 age group living in India.

Step five. Communicate the Analytical findings.

Communication skills are an important part of a work for data scientists but are often widely underestimated. This will in fact be a very difficult part of your work, because it includes explaining the results to the public and other team members in a way that they will clearly understand.

• Graph or chart the information for presentation with tools – R, Python, Tableau, Excel.

• Use "storytelling" to fit the results.

• Answer the different follow-up questions.

• Present data in different formats-reports, websites.

• Believe me; answers will always spark more questions, and the process begins again.

8. BASIC TOOLS (PLOTS, GRAPHS AND SUMMARY STATISTICS) OF EDA

Exploratory data analysis

Exploratory Data Analysis (EDA) is a very critical step that takes place after the feature development and data acquisition process and should be carried out prior to the modeling process. This is because it is really important for a data scientist to be able to understand the essence of the data without making assumptions about it.

The goal of EDA is to use summary statistics and visualizations to better understand the data and to find clues about the patterns, the quality of the data and the assumptions and assumptions of our study. EDA is NOT about creating fancy visualizations or even aesthetically appealing visualizations, the aim is to try and answer data questions. Your goal should be to be able to produce a chart that anyone can look at in a few seconds and understand what's going on. If not, the visualization is too complex (or fancy) and something similar has to be used.

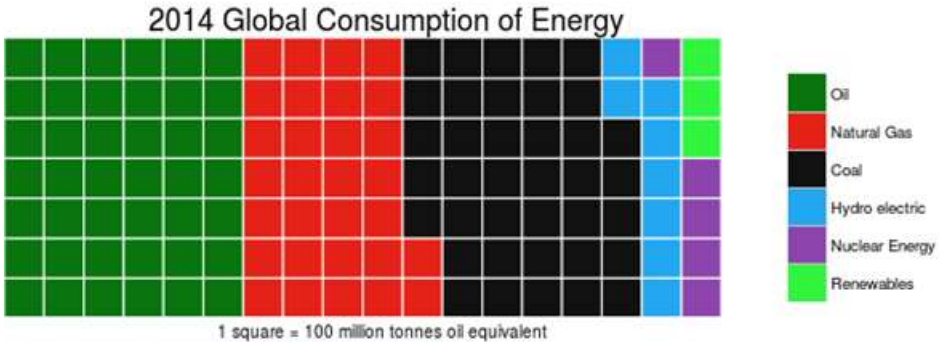
EDA is also very iterative since we first make assumptions based on our first exploratory visualizations, then build some models. We then make visualizations of the model results and tune our models.

Types of Data

Once we can start talking about data discovery, let's learn the various types of data or measurement rates first. I highly recommend that you read Measurement Rates in the online stats book and continue reading the section to browse your statistical information. This segment is simply a synopsis. Data comes in different forms but can be classified into two major groups: structured and unstructured data. Structured data is data that is a form of high-degree or organizational data, such as numerical or categorical data. Examples of standardized data are Temperature, phone numbers, gender. Unstructured data is data in a form that doesn't have the framework of which we are using directly. Types of unstructured data include pictures, videos, audio, text in the language and many others. There is an emerging field called Deep Learning which uses a specialized collection of algorithms with unstructured data that perform well. We will concentrate on structured data in this guide but include brief details

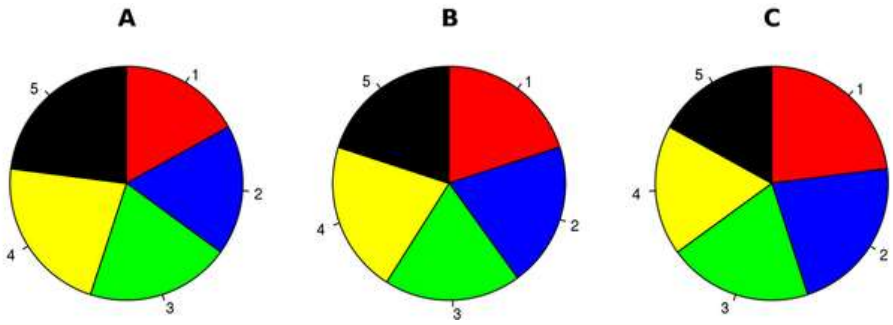
Categorical Variables

Categorical variables can also be nominal or ordinal. ***Nominal data*** has no intrinsic ordering to the categories. For example gender (Male, Female, Other) has no specific rdering. ***Ordinal data*** as clear ordering such as three settings on a toaster (high medium and low). A frequency table (count of each category) is the common statistic for describing categorical data of each variable, and a bar chart or a waffle chart (shown below) are two visualizations which can be used.

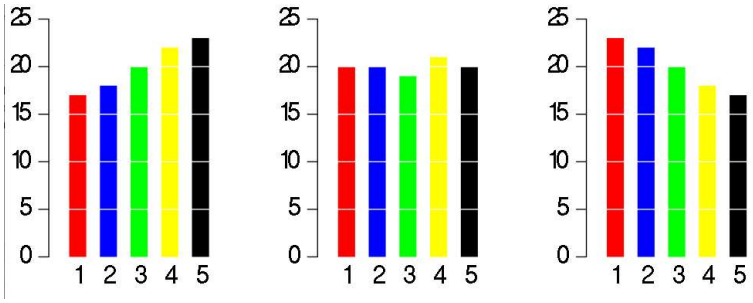


While a pie chart is a very common method for representing categorical variables, it is not recommended since it is very difficult for humans to understand angles. Simply put by statistician and visualization professor Edward Tufte: "pie charts are bad", others agree and some even say that "pie charts are the worst".

For example what you you determine from the follow pie chart?

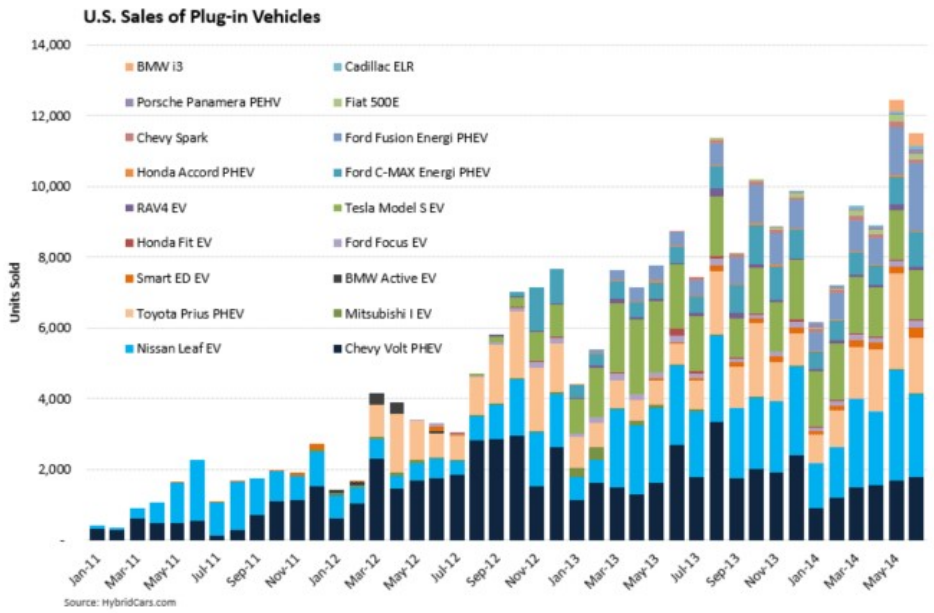


The charts look identical, and it takes more than a couple of seconds to understand the data. Now compare this to the corresponding bar chart

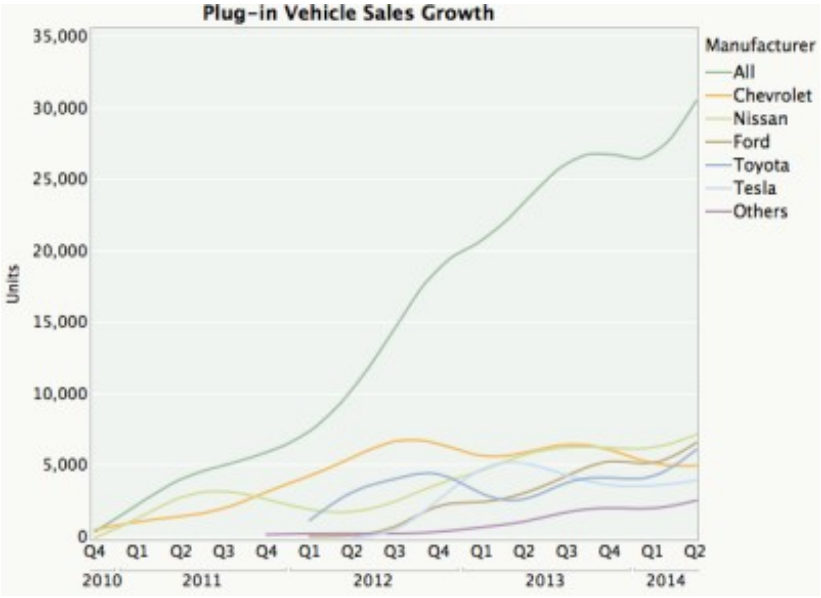


A reader an instantly compare the 5 variables. Since humans have a difficult time comparing angles, bar graphs or waffle diagrams are recommended. There are many other visualizations which are not recommended: spider charts, stacked bar charts, and many other junkcharts.

For example, this visualization is very complicated and difficult to understanding:



Often less is more: the plot redux as a simple line graph:



When dealing with multi-categorical data, avoid using stacked bar charts and follow the guidelines written by Solomon Messing (data scientist I met while working at Facebook).

Numeric Variables

Any value within a finite or infinite interval can be numeric or continuous variables. Examples include weight, height, and temperature. Intervals and ratios are two types of numeric variables. Interval variables have numerical ratios and the same definition over the entire scale, but have no absolute zero. For example, temperature may be meaningfully subtracted or added in Fahrenheit or Celcius (difference between 10 degrees and 20 degrees is the same difference as 40 to 50 degrees), but cannot be measured. For instance, a day that's twice as hot may not be twice as hot.

"The calculation proportion scale is the most insightful scale. This is a scale of intervals with the additional property that its zero position implies the absence of the measured quantity. You can think of a ratio scale as the three previous scales were rolled up into one. It provides a name or category for each object as a nominal scale (numbers are used as labels). The objects are ordered, like an ordinal scale (in terms of ordering the numbers). The same disparity at two positions on the scale has the same value as an interval scale. And, moreover, the same ratio in two places on the scale also has the same meaning. "A good example of a ratio scale is weight, because it has a true zero and can be added, subtracted, multiplied or divided.

Binning (Numeric to Categorical)

The process of transforming numerical variables into categorical is what is otherwise known as discretization. For example, age may be 0-12 (child), 13-19 (teenager), 20-65 (adult), 65+ (senior) categories. Binning is useful because it can be used as a filter to minimize noise or non-linearity and some algorithms need categorical data, such as decision trees. Binning also helps data scientists to easily determine numerical values for outliers, null or incomplete values. Binning strategies involve using equal width (based on the range), equal frequency in of bin, sorted rank, quantiles, or math (such as log) functions. Binning may be used based on entropy of information, or acquiring information.

Encoding

Encoding is the transformation of categorical variables into numeric (or binary) variables, otherwise known as continuation. Sex is a simple example of encoding: -1, 0, 1 may be used to identify males, females and others. Binary encoding is a special case of encoding where the value is set to a 0 or 1 indicating a category's absence or presence. One hot encoding is a special case where each binary is encoded in multiple categories. If we have k categories, this will generate k extra features (increasing the dimensionality of the data). Another type of encoding is using an encoding based on target and probability. The average value is the group, which contains a chance.

Iris DataSet

For exploratory data analysis, we are going be investigating Anscombe's_quartet and Fisher's Iris data set.

"Anscombe's quartet consists of four datasets with almost similar basic statistical features, but looking somewhat different when graphed. Every dataset is composed of eleven points (x, y). They were built by the statistician Francis Anscombe in 1973 in order to demonstrate both the importance of graphing data before analyzing it and the effect of outliers on statistical properties." The Iris data set is a multivariate set of data introduced by Ronald Fisher in his 1936 paper the use of multiple measures. This indicates the variability of three related species in the Iris bulbs.

Summary Statistics

Summary statistics are metrics intended to explain the details. There are several summary measures in the field of descriptive statistics but we will leave the description (and derivation) to textbooks. Examples of summary / descriptive statistics for a single number variable are median, median, mode, max, min, range, quartiles / percentiles, variance, standard deviation, determination coefficient, skewedness and kurtosis. List statistics is the number of distinct counts for categorical variables. The most basic overview metric for text data is the frequency of words and the frequency of reciprocal documents. The descriptive statistics for bivariate data are the linear correlation, the chi-squared or the p value

The Ansombe Excel worksheet with summary statistics can be downloaded from datascienceguide.github.io/datasets/anscombe.xls and the Iris data collection Excel worksheet with summary statistics for each component from datascienceguide.github.io/datasets/iris.xls Note how the mean, standard deviation and correlation between x and y are almost identical on the Ansombe dataset. If we know about linear regression we can also see the same linear regression coefficients.

Visualization

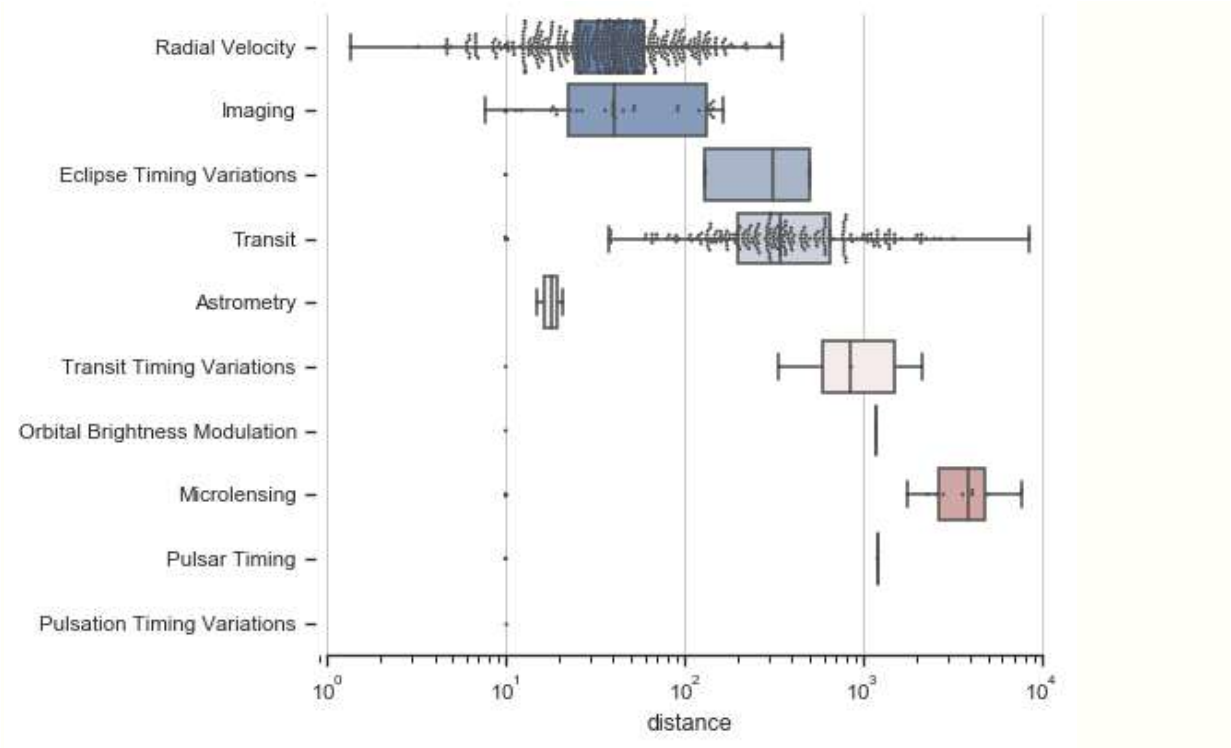
Visualizations can be used to analyze and explain data in addition to the summary statistics. We'll learn about the importance of visualizations in the tutorials, and that using simple statistical properties to represent data is not enough. This is shown by the quartet of Anscombe as outlined in this article Why Data Visualizations (is important) is.

Examples of quantitative data visualizations involve line charts with error bars, histograms, box and whisker graphs, scatter charts or mix charts for categorical data bar charts and waffle charts, and for bivariate results. Many of these visualisations go through the tutorial on exploratory data analysis.

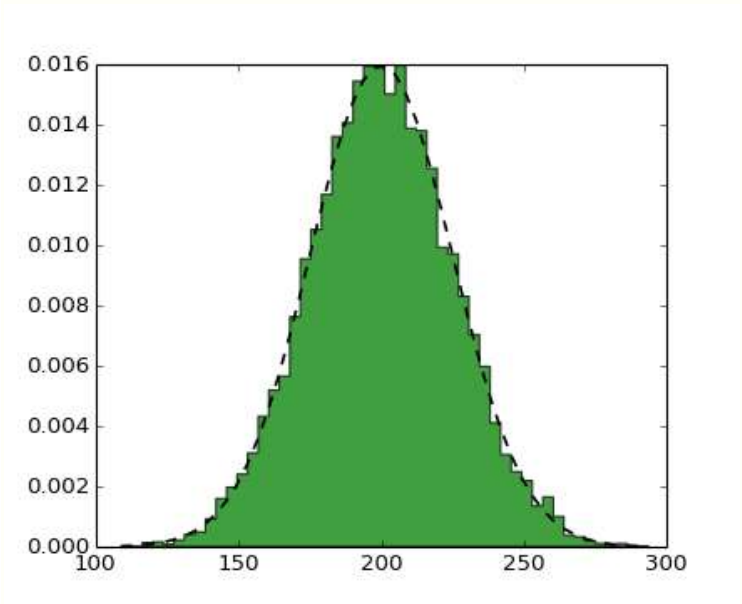
There are several tools and libraries that can be used to plot visualizations:-Excel/ Libre Office — Weka — matplotlib (python) — seaborn (python) — Grammer of graphics (ggplot2) — infovis — Rshiny — Data-driven documents (D3.js) — panoramic Additionally, TibcoSpotfire and Tableau are common yet commercial data visualization solutions.

Univariate data (One variable)

The box plot and the histogram are the two visualizations used to illustrate univariate (1 variable) data. The plot of boxes can be used to show the minimum, maximum, mean, median, quantile and range.

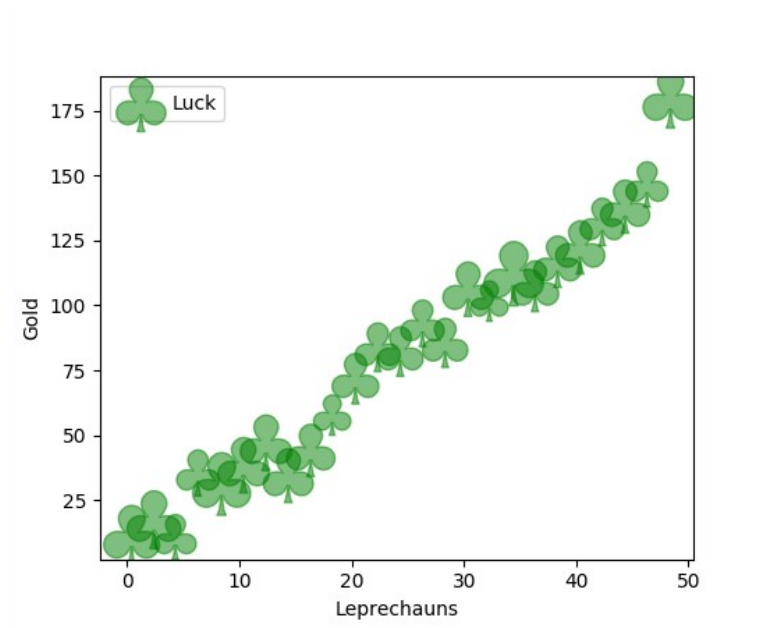


The histogram can be used to show the count, mode, variance, standard deviation, coefficient of deviation, skewness and kurtosis.

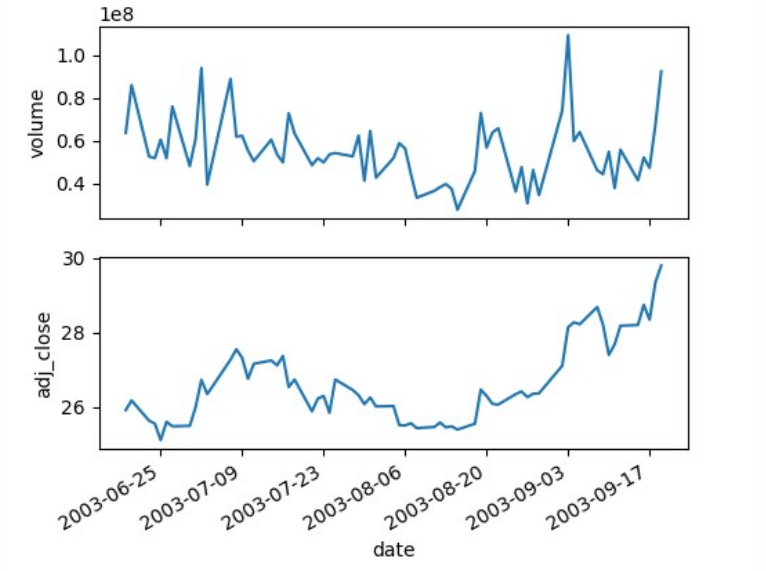


Bivariate data (Two Variables)

When plotting the relation between two variables, one can use a scatter plot.

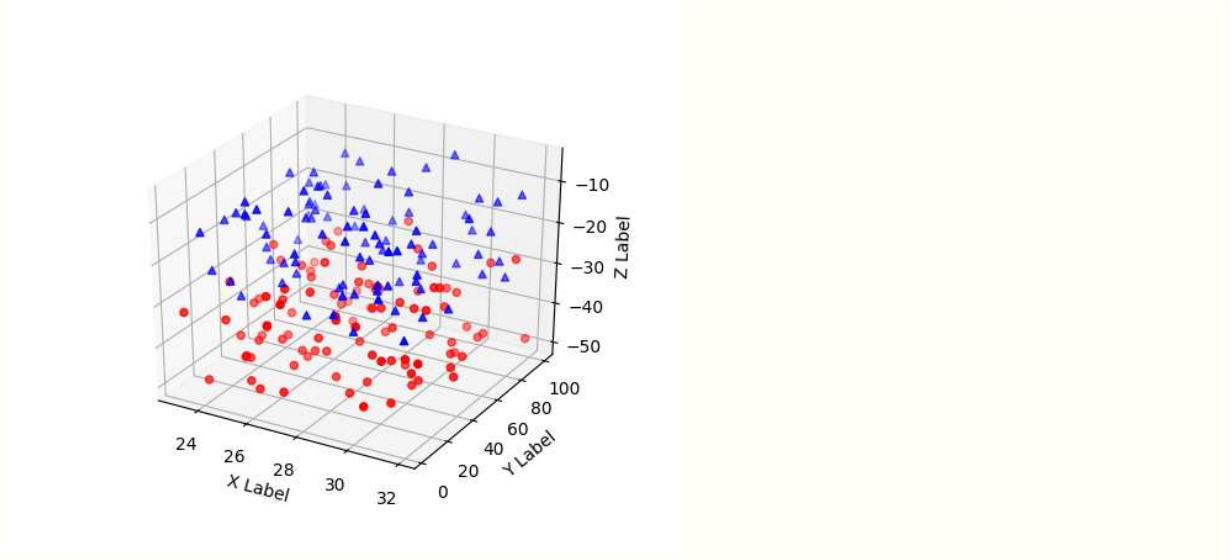


If the data is time series or has an order, a line chart can be used.

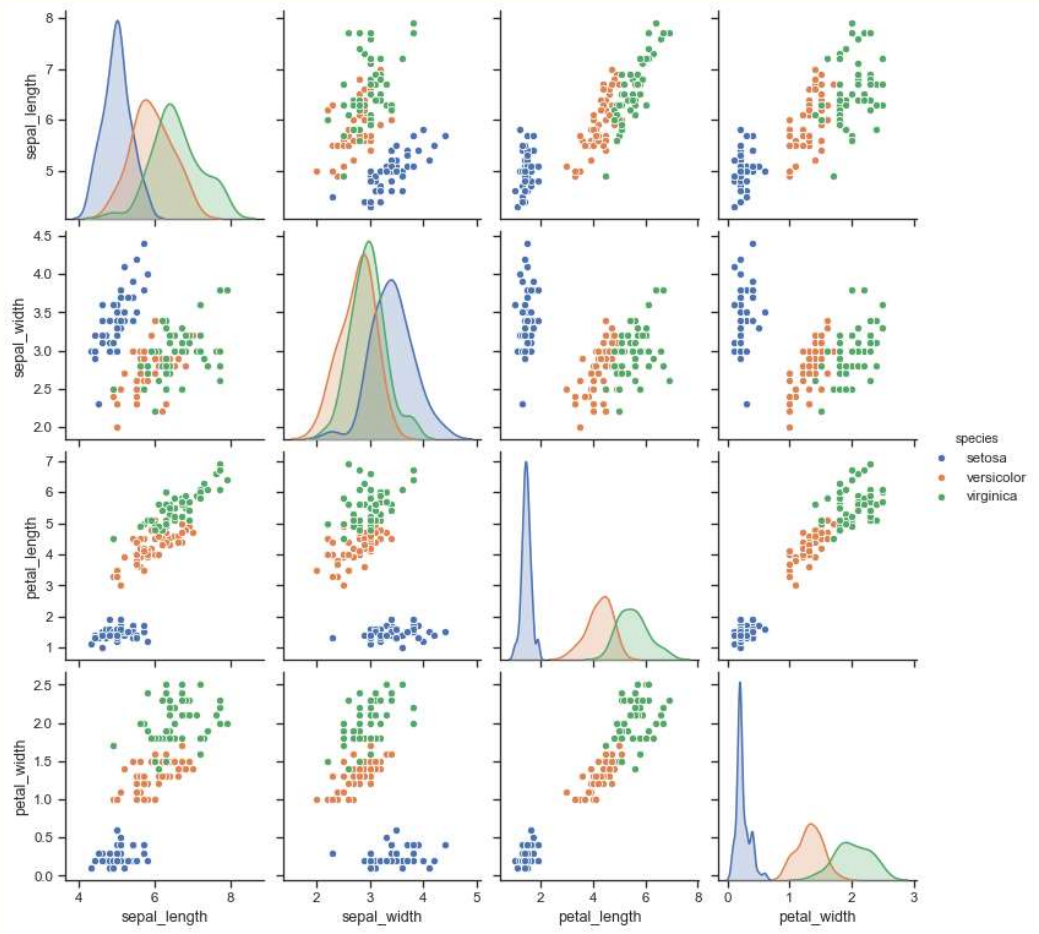


Multivariate Visualization

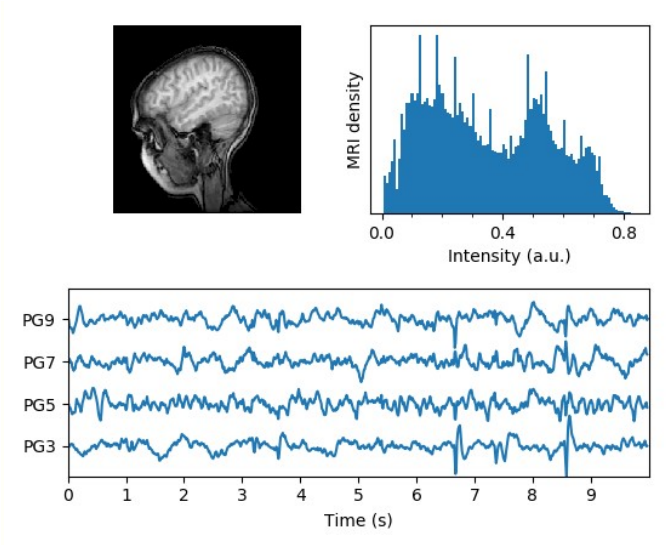
When dealing with multiple variables, it is tempting to make three dimensional plots, but as show below it can be difficult to understand the data:



Rather I recommend create a scatter plot of the relation between each variable:



Combined charts also are great ways to visualize data, since each chart is simple to understand on its own.



For very large dimensionality, you can reduce the dimensionality using principle component analysis, Latent Dirichlet allocation or other techniques and then make a plot of the reduced variables. This is particularly important for high dimensionality data and has applications in deep learning such as visualizing natural language or images.

Text Data

For example with Text data, one could create a world cloud, where the size of each word is the based on its frequency in the text. To remove the words which add noise to the dataset, the documents can be grouped using Topic modeling and only the important words can be displayed.

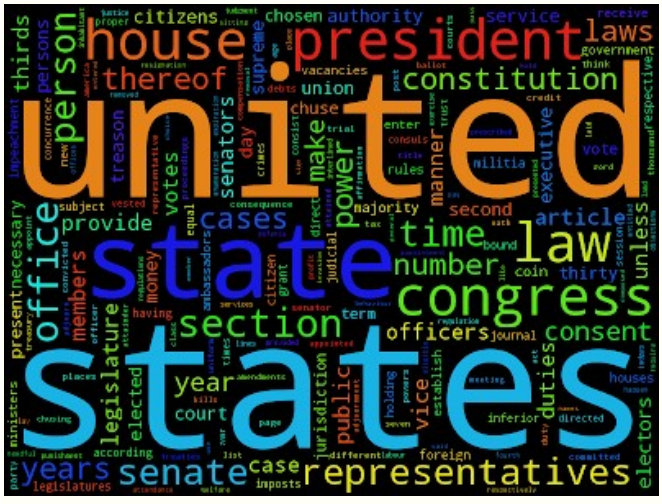
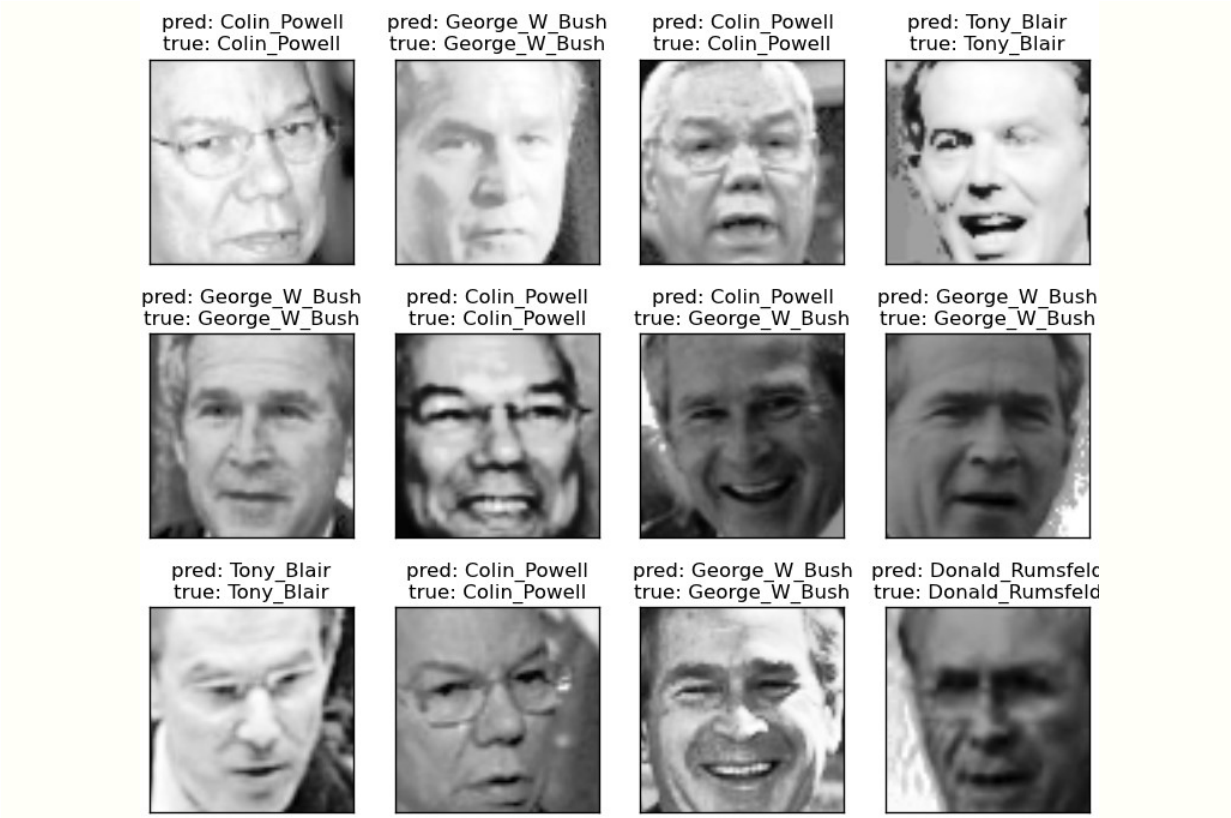


Image data

When doing image classification, it is common to use decomposition and remove the dimensionality of the data. For example, an image before decomposition looks like:



Instead of blindly using decomposition, a data scientist could plot the result:

By looking at the contrast (black and white) in the images, we can see there is an importance with the locations of the eyes, nose and mouth, along with the head shape.

9. THE DATA SCIENCE PROCESS - CASE STUDY, REAL DIRECT (ONLINE REAL ESTATE FIRM)

Data Science Case Studies

Here are the most popular data science case studies that will tell you how data science is used in various industries. Also, the importance of data science in a variety of industries.

1. Data Science in Pharmaceutical Industries

Through improved data processing and cloud-driven applications, it is now easier to access a wide variety of patient information datasets. In the pharmaceutical industry, artificial intelligence and data analytics have revolutionized oncology. With new pharmaceutical innovations emerging every day, it is difficult for physicians to keep up-to-date on treatment options. However, it is difficult to tap into a highly competitive market for more standardized medical care options. However, with the advances in technology and the development of parallel pipelined computational models, it is now easier for the pharmaceutical industry to have a competitive advantage over the market.

With various statistical models such as Markov Chains, it is now possible to predict the probability that doctors will prescribe medicines based on their experience with the brand. In the same way, improving learning is beginning to develop itself in the area of digital marketing. It is used to

understand the patterns of digital participation of physicians and their prescriptions. The main aim of this case study of data science is to discuss the problems facing it and how data science offers solutions to them.

2. Predictive Modeling for Maintaining Oil and Gas Supply

The crude oil and gas industries are faced with a major problem of equipment failures, typically due to the inefficiency of the oil wells and their output at a subpar stage. With the implementation of a effective strategy that advocates for predictive maintenance, well operators can be alerted, as well as informed of maintenance times, to the critical phases of shutdown. This would lead to an increase in oil production and avoid further losses.

Data Scientists can apply the Predictive Maintenance Strategy to the use of data to optimize high-value machinery for the production and refining of oil products. With telemetry data extracted through sensors, a steady stream of historical data can be used to train our machine learning model. This machine learning model will predict the failure of machine parts and will alert operators of timely maintenance in order to avoid oil losses. The Data Scientist assigned to the implementation of the PdM strategy should help prevent hazards and predict machine failure, encouraging operators to take precautions.

3. Data Science in BioTech

The human genome consists of four building blocks – A, T, C and G. Our appearance and characteristics are determined by the three billion permutations of these four building blocks. Although there are genetic defects and lifestyle defects, the results will lead to chronic diseases. Identifying these defects at an early stage will allow doctors and testing teams to take preventive action.

Helix is one of the companies for genome analysis that provides customers with their genomic data. Also, due to the emergence of new computational methodologies, many medicines adapted to particular genetic designs have become increasingly popular. Thanks to the data explosion, we can understand and analyze complex genomic sequences on a wide scale. Data Scientists can use modern computational resources to manage massive databases and understand patterns in genomic sequences to detect defects and provide information to physicians and researchers. In addition, with the use of wearable tools, data scientists may use the relationship between genetic characteristics and medical visits to build a predictive modeling framework.

4. Data Science in Education

Data Science has also changed the way students communicate with teachers and assess their success. Instructors may use data science to evaluate the input they obtain from students and use it to enhance their teaching. Data Science can be used to construct predictive modeling that can predict student drop-out levels based on their results and advise instructors to take the appropriate precautions.

UNIT- II (Basic Machine Learning Algorithms & Applications)

1. LINEAR REGRESSION FOR MACHINE LEARNING

Linear regression in statistics and machine learning is perhaps one of the most well-known and well-understood algorithms.

You will discover the linear regression algorithm in this article, how it operates and how you can best use it in your machine learning projects. In this article you'll learn: • Why linear regression is part of statistics as well as machine learning.

- The other titles known as linear regression.
- Algorithms of representation and inference used to construct a model of linear regression;
- How best to plan the data using linear regression modeling.

To grasp linear regression you don't need to learn any statistics or linear algebras. It is a gentle high-level introduction to the technique to give you enough experience to be able to make successful use of it for your own problems.

Discover how machine learning algorithms work in my latest book like kNN, decision trees, naive bayes, SVM, ensembles and much more, with 22 tutorials and examples in excel.

Let's kick off. Figure 1 shows Linear Regression for Machine Learning

Figure 1: Linear Regression for Machine Learning Photo by Nicolas Raymond.

Isn't linear statistical regression?

Before we immerse yourself in the specifics of linear regression, you might wonder why we are looking at this algorithm.

Isn't it mathematical technique?

Machine learning, more precisely the field of predictive modeling, is concerned primarily with reducing a model's error or making the most accurate predictions possible, at the cost of description. We can borrow, reuse and steal algorithms from many different fields like statistics in applied machine learning and use them for these purposes.

Linear regression has thus been developed in the field of statistics and is being studied as a model for understanding the relationship between them

As such, linear regression was developed in the field of statistics and is studied as a model for understanding the relationship between input and output numerical variables, but was borrowed from machine learning. It's both a statistical algorithm and a machine learning algorithm.

Next, let's look at some of the common names used to refer to a linear regression model. Figure 2 shows the sample of the handy machine learning algorithms mind map.

Get your FREE Algorithms Mind Map

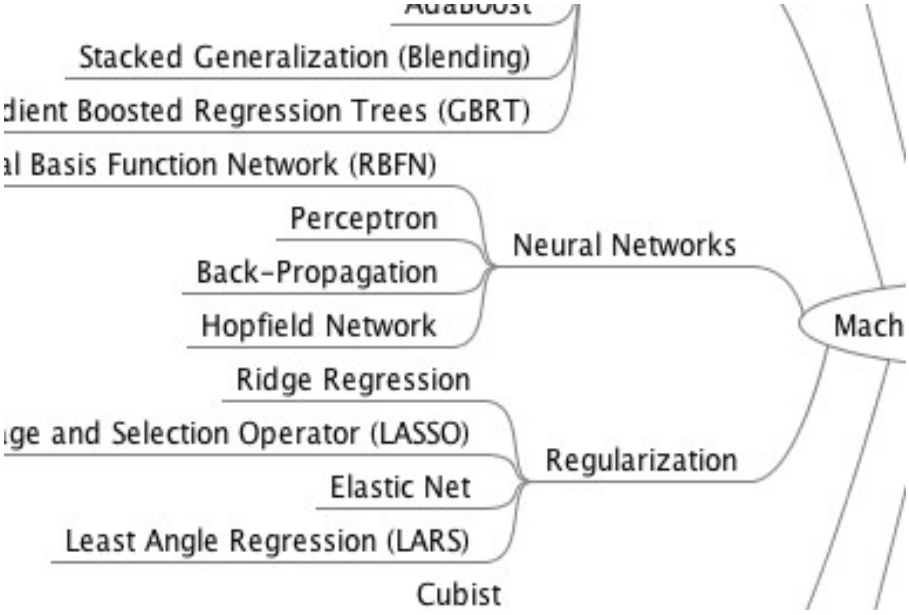


Figure 2: Sample of the handy machine learning algorithms mind map.

Many Linear Regression Names

Things can get really complicated when you start looking at linear regression. The explanation is that for so long (more than 200 years), linear regression has been around. From every possible angle, it has been studied and sometimes every angle has a new and different name. A linear model, such as a model that follows a linear relationship between the input variables (x) and the output variable (y), is a linear regression. More precisely, a linear combination of input variables (x) can be used to determine y.

The approach is referred to as simple linear regression when a single input variable (x) exists. Statistical literature also refers to the approach as multiple linear regression when multiple input variables are available. Different techniques, the most common of which is called Ordinary Lowest Squares, may be used to prepare or train linear regression equations from data. Therefore it is common to refer to the model built in this way as ordinary.

Linear Regression Model Representation

Linear regression, since it is so easy to depict, is a popular model. A representation is a linear equation that combines a particular set of input values (x) with the solution for which the output is expected for that set of input values (y). As such, both the values of input (x) and output are numeric.

For each input value or column, the linear equation assigns one scale factor, called the coefficient, represented by the Greek capital letter Beta (B). An additional coefficient is often introduced, which gives the line an extra degree of freedom (for example, going up and down on a two-dimensional plot) and is also referred to as the coefficient of intercept or bias.

For example, in a simple regression problem (a single x and a single y), the form of the model would be:

$$y = B0 + B1*x$$

In higher dimensions, when we have more than one input (x), the line is called a plane or a hyperplane. The representation is therefore the form of the equation and the specific values used for the coefficients (e.g. B0 and B1 in the example above).

It is common to talk about the complexity of a regression model such as linear regression. This refers to the number of coefficients used for the model.

When the coefficient is zero, the influence of the input variable on the model is effectively removed from the model prediction (0 * x = 0). This becomes relevant when you look at the regularization methods that change the learning algorithm to reduce the complexity of the regression models by putting pressure on the absolute size of the coefficients, driving some to zero.

Now that we understand the representation used for the linear regression model, let's look at some ways that we can learn this representation from the data.

Figure 2: What is Linear Regression?

Linear Regression Learning Model

Learning a linear regression model means estimating the values of the coefficients used in the representation with the data available to us.

In this section, we will take a brief look at four techniques for the preparation of a linear regression model. This is not enough information to implement it from scratch, but enough to get a taste of the computation and the trade-offs involved.

There are a lot more techniques because the model is so well studied. Take note of Ordinary Less Squares because it is the most common method used in general. Take note of Gradient Descent as it is the most common technique taught in machine learning classes.

1. Simple Linear Regression

With simple linear regression, if we have a single input, we can use statistics to estimate the coefficients. This requires you to calculate statistical properties from data such as mean, standard deviations, correlations and covariance. All data must be made available for the purpose of crossing and calculating statistics.

2. Ordinary Least Squares

If we have more than one input, we can use the Ordinary Lowest Squares to estimate the coefficient values.

The Ordinary Least Squares procedure seeks to minimize the sum of the squared residues. This means that given the regression curve through the info , we calculate the space from each datum to the regression curve , square it, and sum all the squared errors together. This is the quantity that the least common squares are trying to minimize.

This approach treats the data as a matrix and uses linear algebra operations to estimate the optimum coefficient values. This means that all the data must be available and you must have enough memory to fit the data and perform the matrix operation.

It is unusual to perform the Ordinary Least Squares procedure yourself, unless it is done as a linear algebra exercise. It's more likely you're going to call a procedure in a linear algebra library. This procedure is very quick to calculate.

3. Gradient Descent

When one or more inputs are available, you can use the process of optimizing coefficient values by iteratively minimizing the model error on your training data.

This operation is called Gradient Descent, starting with random values for each coefficient. The sum of squared errors is calculated for each pair of input and output values. The learning rate is used as a scale factor and the coefficients are updated in order to minimize the error. The process is repeated until a minimum amount of squared error is achieved or no further improvement is possible.

When using this method, you must select the learning rate (alpha) parameter that will determine the size of the improvement step to be taken for each iteration of the procedure.

Gradient descent is often taught using a linear regression model because it is relatively easy to understand. In practice, it is useful when you have a very large dataset in either the number of rows or the number of columns that may not fit into your memory.

4. Regularization

There are extensions to the training of a linear model called regularization methods. Both aim to minimize the sum of the squared error of the model on the training data (using ordinary least squares) but also to reduce the complexity of the model (such as the number or absolute size of the sum of all coefficients in the model).

Two common examples of regularization procedures for linear regression are:

1. Lasso regression: where ordinary least squares are modified to minimize the absolute sum of the coefficients (called L1 regularization) as well.

2. Ridge Regression: where the ordinary least squares are modified to minimize the squared absolute sum of the coefficients (called L2 regularization)

These methods are effective to use when there is collinearity in your input values, and the ordinary least squares would override the training data.

Now that you know some techniques to learn the coefficients in a linear regression model, let's look at how we can use a model to make new data predictions.

Making Linear Regression Predictions Since representation is a linear equation, making predictions is as simple as solving an equation for a specific set of inputs.

Let's use an example to make this concrete. Imagine that we predict weight (y) from height (x). Our linear regression model for this problem would be:

$$y = B0 + B1 * x1$$

or

$$\text{weight} = B0 + B1 * \text{height}$$

Where B0 is that the bias coefficient and B1 is that the coefficient for the height column. We use a learning technique to seek out an honest set of coefficient values. Once found, we will connect different height values to predict the load .

For example, let’s use B0 = 0.1 and B1 = 0.5. Let’s plug them in and calculate the weight (in kilograms) for an individual with the height of 182 centimeters.

$$\text{weight} = 0.1 + 0.5 * 182$$

$$\text{weight} = 91.1$$

You can see that the above equation could be plotted as a line in two-dimensions. The B0 is our starting point regardless of what height we have. We can run through a bunch of heights from 100 to 250 centimeters and plug them to the equation and get weight values, creating our line.

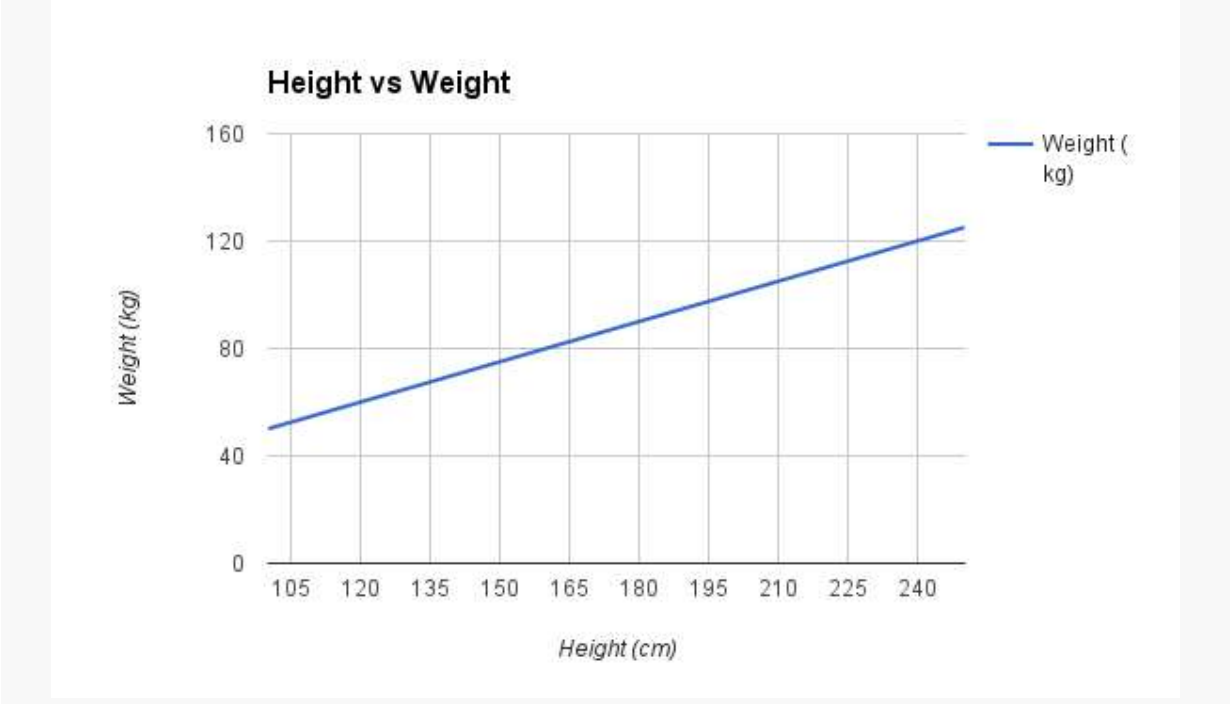


Figure 3: Sample Height vs Weight Linear Regression

Now that we know how to make predictions given a learned linear regression model, let's look at some rules of thumb for preparing our data to make the most of this type of model.

Preparing Data for Linear Regression

Linear regression has been studied extensively, and there is a lot of literature on how your data needs to be structured to make the most of the model.

As such, there is a lot of sophistication in talking about these requirements and expectations that can be intimidating. In practice, these rules can be used more as thumb rules when using Ordinary Less Squares Regression, the most common linear regression implementation.

Try using these heuristics to prepare your data differently and see what works best for your problem.

1. Linear Assumption. Linear regression assumes that the relationship between input and output is linear. It doesn't support anything else. This may be obvious, but it's a good thing to remember when you have a lot of attributes. You may need to transform the data to make the relationship linear (e.g. transform log for an exponential relationship).

2. Remove your noise. Linear regression assumes that the input and output variables are not noisy. Consider using data cleaning operations that will make it easier for you to expose and clarify the signal in your data. This is most important for the output variable and, if possible, you want to remove outliers in the output variable (y).

3. Remove Collinearity from me. Linear regression is over-fitting your data when you have highly correlated input variables. Consider calculating pairwise correlations for your input data and removing the most correlated data.

4. The Gaussian Distribution. Linear regression makes more reliable predictions if your input and output variables have a Gaussian distribution. You may get some benefit from transforms (e.g. log or BoxCox) on your variables to make their distribution look more Gaussian.

5. Rescale Inputs: Linear regression will often make more reliable predictions if you rescale input variables using standardization or normalization.

2. K-NEAREST NEIGHBORS (K-NN)

K-nearest neighbors (KNN) algorithm may be a sort of controlled ML algorithm which will be used for both classification and regression predictive problems. However, it's mainly used for the classification of predictive problems within the industry.

The subsequent two properties would define KNN well –

• Lazy learning algorithm − KNN may be a lazy learning algorithm because it doesn't have a specialized training phase and uses all data for training while classifying.

• Non-parametric learning algorithm − KNN is additionally a non-parametric learning algorithm because it assumes nothing about the underlying data.

2.1 Working of KNN Algorithm

K-nearest neighbors (KNN) algorithm uses 'feature similarity' to predict the values of new data points, which means that a value will be assigned to the new data point based on how closely the points in the training set match. We can understand how it works by following steps − Step 1 − We need a data set to implement any algorithm. So we have to load the training as well as the test data during the first step of KNN.

Step 2 − Next, we'd like to settle on the value of K, i.e. the nearest data point. K could be any integer.

Step 3 − For each point of the test data, do the following −

3.1 − Calculate the distance between the test data and each row of training data using any method, namely: Euclidean, Manhattan or Hamming distance. Euclidean is the most common method used to calculate distance.

3.2 − Now, based on the distance value, sort it in ascending order.

3.3 − Next, the top rows of K will be selected from the sorted array.

3.4 − Now assign a class to the test point based on the most frequent class of these rows.

Step 4 − End Example The following is an example to understand the concept of K and the working of the KNN algorithm − Suppose we have a dataset that can be plotted as follows shown in Figure 4−

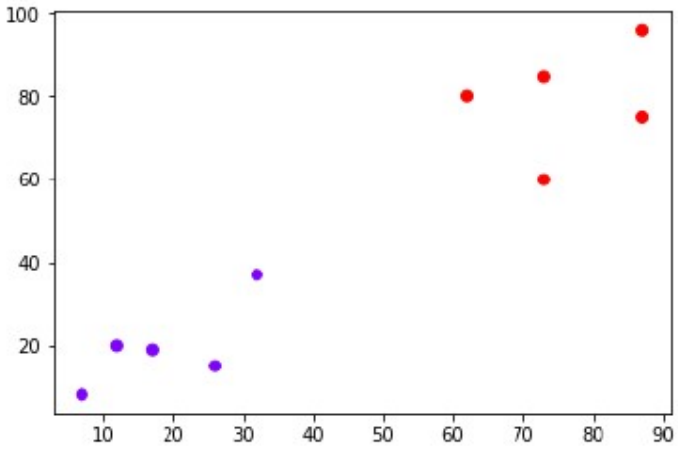


Figure 4: Dataset

Now, we'd like to classify new data point with black dot (at point 60, 60) into blue or red class. We are assuming K = 3 i.e. it would find three nearest data points. It is shown in Figure 5 −

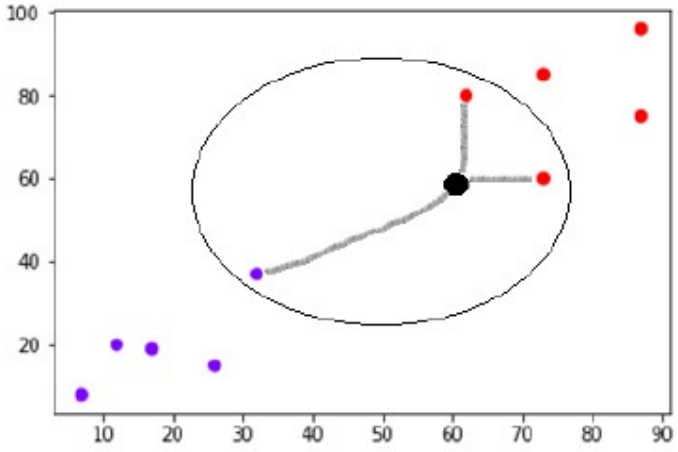


Figure 5: Finding three nearest neighbors

We can see in Figure 5 the three nearest neighbors of the data point with black dot. Among those three, two of them lie in Red class hence the black dot will also be assigned in red class.

Implementation in Python

As we all know K-nearest neighbors (KNN) algorithm are often used for both classification also as regression. The following are the recipes in Python to use KNN as classifier also as regressor – KNN as Classifier

First, start with importing necessary python packages −

```
importnumpy as np
importmatplotlib.pyplot as plt
import pandas as pd
```

Next, download the iris dataset from its weblink as follows −

```
path = "https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data"
```

Next, we need to assign column names to the dataset as follows −

```
headernames = ['sepal-length', 'sepal-width', 'petal-length', 'petal-width', 'Class']
```

Now, we need to read dataset to pandasdataframe as follows −

```
dataset = pd.read_csv(path, names = headernames)
dataset.head()
```

	sepal-length	sepal-width	petal-length	petal-width	Class
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

Data Preprocessing will be done with the help of following script lines.

```
X = dataset.iloc[:, :-1].values
y = dataset.iloc[:, 4].values
```

Next, we will divide the data into train and test split. Following code will split the dataset into 60% training data and 40% of testing data −

```
fromsklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.40)
```

Next, data scaling will be done as follows −

```
fromsklearn.preprocessing import StandardScaler
scaler = StandardScaler()
scaler.fit(X_train)
```

```
X_train = scaler.transform(X_train)
X_test = scaler.transform(X_test)
```

Next, train the model with the help of KNeighborsClassifier class of sklearn as follows –

```
fromsklearn.neighbors import KNeighborsClassifier
classifier = KNeighborsClassifier(n_neighbors = 8)
classifier.fit(X_train, y_train)
```

At last we need to make prediction. It can be done with the help of following script −

```
y_pred = classifier.predict(X_test)
```

Next, print the results as follows −

```
fromsklearn.metricsimportclassification_report,confusion_matrix,accuracy_score
result=confusion_matrix(y_test,y_pred)
print("Confusion Matrix:")
print(result)
result1 =classification_report(y_test,y_pred)
print("Classification Report:",)
print(result1)
result2 =accuracy_score(y_test,y_pred)
print("Accuracy:",result2)
```

Output

```
Confusion Matrix:

[[21 0 0]
[ 0 16 0]
[ 0 7 16]]

Classification Report:

              precision recall    f1-score  support
Iris-setosa       1.00              1.00      1.00      21
Iris-versicolor   0.70              1.00      0.82      16
Iris-virginica    1.00              0.70      0.82      23
microavg0.88              0.88      0.88      60
macroavg          0.90              0.90      0.88      60
weightedavg       0.92              0.88      0.88      60

Accuracy: 0.8833333333333333
```

KNN as Regressor

First, start with importing necessary Python packages −

```
importnumpy as np
import pandas as pd
```

Next, download the iris dataset from its weblink as follows −

```
path = "https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data"
```

Next, we need to assign column names to the dataset as follows −

```
headernames = ['sepal-length', 'sepal-width', 'petal-length', 'petal-width', 'Class']
```

Now, we need to read dataset to pandasdataframe as follows −

```
data=pd.read_csv(url, names =headernames)
array=data.values
X =array[:,:2]
Y =array[:,2]
data.shape
output:(150,5)
```

Next, import *KNeighborsRegressor* from *sklearn* to fit the model −

```
fromsklearn.neighbors import KNeighborsRegressor
knnr = KNeighborsRegressor(n_neighbors = 10)
knnr.fit(X, y)
```

At last, we can find the MSE as follows −

```
print ("The MSE is:",format(np.power(y-knnr.predict(X),2).mean()))

Output
The MSE is: 0.12226666666666669
```

2.3 Pros and Cons of KNN

Pros

• It is a very simple algorithm to understand and interpret.

• It's very useful for non-linear data because there's no assumption about the information in this algorithm.

- It may be a versatile algorithm which will be used for both classification and regression.
- It is comparatively accurate, but there are far better supervised learning models than KNN.

Cons

- It's a bit expensive algorithm, because it stores all the training data.

- High memory storage required compared to other supervised learning algorithms.
- The prediction is slow within the case of a large N.
- It is extremely sensitive to the size of the information also on irrelevant features.

2.4 Applications of KNN

The following are some of the areas in which KNN can be successfully applied –

1. The KNN Banking System can be used in the banking system to predict the weather and the individual is fit for loan approval? Does that individual have the same characteristics as one of the defaulters?
2. Calculation of credit ratings KNN algorithms can be used to find an individual's credit rating by comparing it to persons with similar characteristics.
3. Politics With the assistance of KNN algorithms, we will classify potential voters into different classes like "Will vote," "Will not vote," "Will vote to the Congress Party," "Will vote to the BJP Party."
4. Other areas where the KNN algorithm is often used are Speech Recognition, Handwriting Detection, Image Recognition and Video Recognition.

3. K-MEANS

K-means algorithm is an iterative algorithm that attempts to divide the dataset into Kpre-defined separate non-overlapping subgroups (clusters) where each data point belongs to only one group. It tries to make inter-cluster data points as similar as possible while keeping clusters as different (far) as possible. It assigns data points to a cluster in such a way that the sum of the squared distance between the data points and the centroid cluster (arithmetic mean of all data points belonging to that cluster) is at a minimum. The less variation we have within clusters, the more homogenous (similar) the data points are within the same cluster.

The way k-means algorithm works is as follows:

1. Specify the number of K clusters.

2. Initialize the centroids by first shuffling the dataset and then randomly selecting K data points for the centroids without substitution.

3. Keep iterating until there is no change in the centroids. i.e. the assignment of data points to clusters does not change.

4. Calculate the sum of the squared distance between the data points and all the centroids.

5. Assign each data point to the nearest cluster (centroid).

6. Calculate the centroids for clusters by taking the average of all data points belonging to each cluster.

The following approach k-means to solve the problem is called Expectation-Maximization. The E-step assigns the data points to the nearest cluster. The M— step is computing the centroid of each cluster. Below is a breakdown of how we can solve it mathematically (feel free to skip it).

The objective function is:

$$J = \sum_{i=1}^{m} \sum_{k=1}^{K} w_{ik} \|x^i - \mu_k\|^2 \qquad (1)$$

Where wik=1 for data point xi if it belongs to cluster *k*; otherwise, wik=0. Also, μk is the centroid of xi's cluster.

It's a minimization problem of two parts. We first minimize J w.r.t. wik and treat μk fixed. Then we minimize J w.r.t. μk and treat wik fixed. Technically speaking, we differentiate J w.r.t. wik first and update cluster assignments (*E-step*). Then we differentiate J w.r.t. μk and recompute the centroids after the cluster assignments from previous step (*M-step*). Therefore, E-step is:

$$\frac{\partial J}{\partial w_{ik}} = \sum_{i=1}^{m} \sum_{k=1}^{K} \|x^i - \mu_k\|^2$$

$$\Rightarrow w_{ik} = \begin{cases} 1 & \text{if } k = argmin_j \|x^i - \mu_j\|^2 \\ 0 & \text{otherwise.} \end{cases} \qquad (2)$$

In other words, assign the data point xi to the closest cluster judged by its sum of squared distance from cluster's centroid.

And M-step is:

$$\frac{\partial J}{\partial \mu_k} = 2 \sum_{i=1}^{m} w_{ik}(x^i - \mu_k) = 0$$

$$\Rightarrow \mu_k = \frac{\sum_{i=1}^{m} w_{ik} x^i}{\sum_{i=1}^{m} w_{ik}} \qquad (3)$$

Which translates to recomputing the centroid of each cluster to reflect the new assignments.

Few things to note here:

• Since cluster algorithms, together with k-means, use distance-based measurements to work out the similarity between data points, it's counseled that data ought to be standardized to own a mean zero and a typical deviation of 1, since nearly always the characteristics in any knowledge set would have totally different units of measuring like age vs financial gain.

• In sight of the k-mean repetitious nature and therefore the random formatting of the centroids at the start of the algorithmic program, {different|totally totally different|completely different} initializations

could result in different clusters, because the k-mean algorithmic program could also be stuck to the native optimum and should not converge to the world optimum.

It is so counseled that the algorithmic program be run mistreatment totally different center of mass initializations which the results of the run be chosen that yielded a lower total of square distance.

• The assignment of examples doesn't amendment is that the same issue as no amendment in within-cluster variation:

$$\frac{1}{m_k}\sum_{i=1}^{m_k}\|x^i - \mu_{c^k}\|^2 \qquad (4)$$

Implementation

We will use simple implementation of k-means here to illustrate some of the concepts. Then we will use sklearn implementation that makes it more efficient to take care of a lot of things for us.

Applications

K-means algorithm is very popular and is used in a variety of applications such as market segmentation, document clustering, image segmentation and image compression, etc. The goal usually when we're undergoing a cluster analysis is either:

1. Get a meaningful insight into the structure of the data we're dealing with.

2. Cluster-then-predicts where different models will be built for different subgroups if we believe there is a wide variation in the behavior of different subgroups. An example of this is the clustering of patients into different subgroups and the development of a model for each subgroup to predict the risk of heart attack.

4. FILTERING SPAM

4.1 Spam

- Spam additionally referred to as unsought industrial Email (UCE)
- Involves causation messages by email to varied recipients at constant time (Mass Emailing).
- Grew exponentially since 1990 however has leveled off recently and is not any longer growing exponentially
- 80% of all spam is sent by but two hundred spammers

4.2 Purpose of Spam

- Advertisements
- Pyramid schemes (Multi-Level Marketing)

- Giveaways
- Chain letters
- Political email
- Stock market recommendation

Spam as a retardant

- Consumes computing resources and time
- Reduces the effectiveness of legitimate advertising
- Cost Shifting
- Fraud
- Identity thievery
- Consumer Perception
- Global Implications

John Borgan [ReplyNet] – "Junk email isn't simply annoying any longer. It's feeding into productivity. It's feeding into time".

Some Statistics

- **Cost of Spam 2009:**
  - $130 billion worldwide
  - ϒ $42 billion in US alone ϒ 30% increase from 2007 estimates
  - 100% increase in 2007 from 2005
- **Main components of cost:**
  - Productivity loss from inspecting and deleting spam missed by spam control products (False Negatives)
  - Productivity loss from searching for legitimate mails deleted in error by spam control products (False Positives)
  - Operations and helpdesk costs (Filters and Firewalls – installment and maintenance)

Email Address Harvesting - Process of obtaining email addresses through various methods:

- Purchasing /Trading lists with other spammers
- Bots
- Directory harvest attack
- Free Product or Services requiring valid email address
- News bulletins /Forums

- **Cost of Spam 2009:**

  - $130 billion worldwide
  - $42 billion in u. s. u. s. unit of time increase from 2007 estimates
  - 100% increase in 2007 from 2005

- **Main components of cost:**

  - Productivity loss from inspecting and deleting spam lost by spam management merchandise (False Negatives)
  - Productivity loss from sorting out legitimate mails deleted in error by spam management merchandise (False Positives)
  - Operations and repair costs (Filters and Firewalls – installment and maintenance)

Email Address gather - methodology of obtaining email addresses through various methods:

- Purchasing /Trading lists with completely different spammers
- Bots
- Directory harvest attack
- Free Product or Services requiring valid email address
- News bulletins /Forums

4.3 Spam Life Cycle

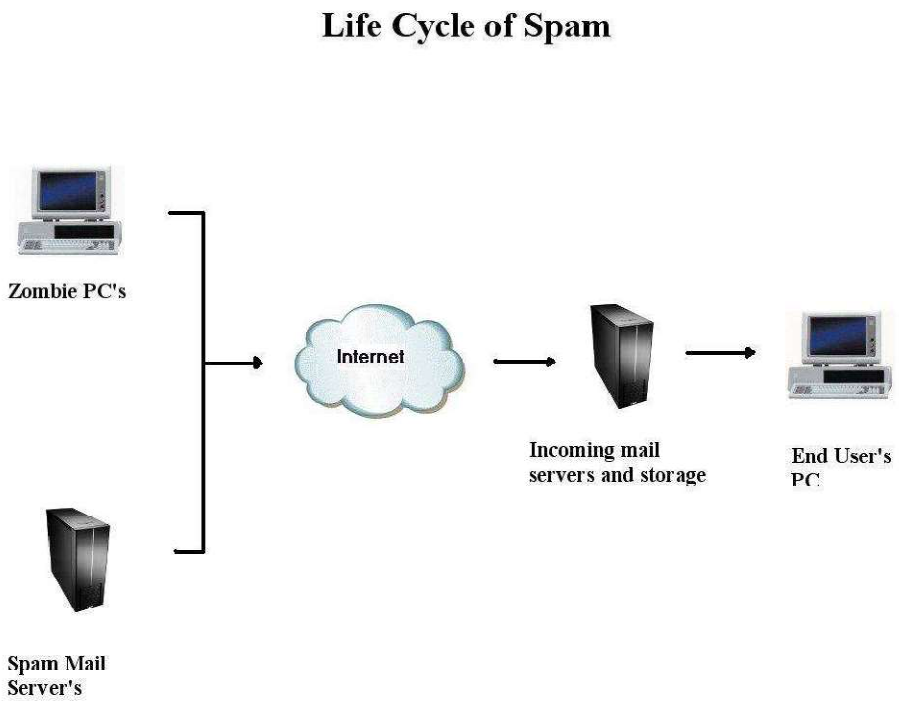


Figure 6: Life cycle of spam

Types of Spam Filters

1. Header Filters
   a. Look at email headers to judge if forged or not

b. Contain more information in addition to recipient , sender and subject fields

2. **Language Filters**
    a. filters based on email body language
    b. Can be used to filter out spam written in foreign languages

3. **Content Filters**
    a. Scan the text content of emails
    b. Use fuzzy logic
4. **Permission Filters**
    a. Based on Challenge /Response system
5. White list/blacklist Filters
    a. Will only accept emails from list of "good email addresses"
    b. Will block emails from "bad email addresses"
6. **Community Filters**
    a. Work on the principal of "communal knowledge" of spam
    b. These types of filters communicate with a central server.
7. **Bayesian Filters**
    a. Statistical email filtering
    b. Uses Naïve Bayes classifier

Spam Filters Properties

1. Filter must prevent spam from entering inboxes
2. Able to detect the spam without blocking the ham
    a. Maximize efficiency of the filter
3. Do not require any modification to existing e-mail protocols
4. Easily incremental
    a. Spam evolve continuously
    b. Need to adapt to each user

Data Mining and Spam Filtering

- Spam Filtering can be seen as a specific text categorization (Classification)
- History
    - Jason Rennie's**iFile** (1996); first know program to use Bayesian Classification for spam filtering

Bayesian Classification

1. Specific words are likely to occur in spam emails and legitimate emails For example, most email users often encounter the word "Viagra" in spam emails, but rarely see it in other emails
2. The filter does not know these probabilities beforehand, and must be trained first so that it can build them up to
3. To train the filter, the user must manually indicate whether or not the new email is spam
4. For all of the words in each training email, the filter will adjust the probability that each word will appear in the spam or legitimate email in its database.

For example, Bayesian spam filters will typically have learned a very high spam probability for the words "Viagra" and "refinance," but a very low spam probability for words that are seen only in legitimate emails, such as the names of friends and family members

5. After training, the word probabilities are used to calculate the probability that an email containing a specific set of words belongs to either category
6. Each word in the email contributes to the spam probability of the email, or only the most interesting words
7. This contribution is calculated using the Bayes Theorem
8. Then, the spam probabil (false positive or false negative) which allows the software to dynamically adapt to the ever-evolving nature of spam
10. Some spam filters combine the results of both Bayesian spam filtering and other heuristics (predefined content rules, envelope viewing, etc.) resulting in even higher filtering accuracy.

Computing the Probability

1. Calculation of the probability that a message containing a given word is spam:

2. Suppose the suspected message contains the word "replica" Most people who are used to receive e-mails know that this message is likely to be spam

3. The formula used by the software for computing.

$$\Pr(S|W) = \frac{\Pr(W|S) \cdot \Pr(S)}{\Pr(W|S) \cdot \Pr(S) + \Pr(W|H) \cdot \Pr(H)}$$

where:

- $\Pr(S|W)$ is the probability that a message is a spam, knowing that the word "replica" is in it;
- $\Pr(S)$ is the overall probability that any given message is spam;
- $\Pr(W|S)$ is the probability that the word "replica" appears in spam messages;
- $\Pr(H)$ is the overall probability that any given message is not spam (is "ham");
- $\Pr(W|H)$ is the probability that the word "replica" appears in ham messages.

2. Spam or ham:

- Recent statistics show that current probability of any message to be spam is 80%, at the very least:

$$\Pr(S) = 0.8; \Pr(H) = 0.2$$

- However, most spam detection software are "not biased", meaning that they have no prejudice regarding the incoming email

$$\Pr(S) = 0.5; \Pr(H) = 0.5$$

  - It is advisable that the datasets of spam and ham are of **same size**

3. Combining individual probabilities:

- The Bayesian spam filtering software makes the "naïve" assumption that the words present in the message are independent events

- With that assumption,

$$p = \frac{p_1 p_2 \cdots p_N}{p_1 p_2 \cdots p_N + (1 - p_1)(1 - p_2) \cdots (1 - p_N)}$$

where:

- p is the probability that the suspect message is spam
- p_i is the probability $p(S|W_i)$

Pros & Cons

Advantages

- It can be trained on a per-user basis
  - The spam that a user receives is often related to the online user's activities

Disadvantages

- Bayesian spam filtering is susceptible to Bayesian poisoning
  - Insertion of random innocuous words that are not normally associated with spam
  - Replace text with pictures
  - Google, by its Gmail email system, performing an OCR to every mid to large size image, analyzing the text inside
- Spam emails not only consume computing resources, but can also be frustrating
- Numerous detection techniques exist, but none is a "good for all scenarios" technique
- Data Mining approaches for content based spam filtering seem promising

4.8 How to Design a Spam Filtering System with Machine Learning Algorithm

Exploratory Data Analysis (EDA)

Exploratory Data Analysis is a very important data science process. It helps the data scientist to understand the data at hand and relates it to the business context.

The open source tools that I will use to visualize and analyze my data are Word Cloud.

Word Cloud is a data visualization tool used to represent text data. The size of the text in the image represents the frequency or importance of the words in the training data.

Steps to take in this section:

1. Get the email data
2. Explore and analyze the data
3. Visualize the training data with Word Cloud & Bar Chart

Get the spam data

Data is the essential ingredients before we can develop any meaningful algorithm. Knowing where to get your data can be a very handy tool especially when you are just a beginner.

Below are a few of the famous repositories where you can easily get thousand kind of data set for **free:**

1. UC Irvine Machine Learning Repository
2. Kaggle datasets
3. AWS datasets

You can go to this link (https://spamassassin.apache.org/old/publiccorpus/) to go to the data set for this email spamming data set, which is distributed by Spam Assassin. There are a few categories of data that you can read from readme.html to get more background information about the data.

In short, there are two types of data present in this repository, namely ham (non-spam) and spam data. In addition, the ham data is easy and hard, which means that there are some non-spam data that are very similar to spam data. This could make our system difficult to make a decision.

If you are using Linux or Mac, simply do this on the terminal, wget is simply a command that will help you download the url file:

For this email spamming data set, which is distributed by Spam Assassin, you can go to this link (https://spamassassin.apache.org/old/publiccorpus/) to go to the data set. There are a few categories of data that you can read from readme.html to get more background information on the data.

In short, there are two types of data in this repository that are ham (non-spam) and spam data. In addition, ham data is easy and hard, which means that there are some non-spam data that are very similar to spam data. This may pose a difficulty for our system to make a decision.

If you are using Linux or Mac, simply do this on the terminal, wget is simply a command that helps you download a file with a url:

```
wget
https://spamassassin.apache.org/old/publiccorpus/20030228_easy_ham.tar.bz2

wget
https://spamassassin.apache.org/old/publiccorpus/20030228_easy_ham_2.tar.bz2

wget
https://spamassassin.apache.org/old/publiccorpus/20030228_spam.tar.bz2

wget
https://spamassassin.apache.org/old/publiccorpus/20050311_spam_2.tar.bz2

wget
https://spamassassin.apache.org/old/publiccorpus/20030228_hard_ham.tar.bz2
```

Let's run some code and see what content is inside all these emails!

Explore and Analyze the Data

Let's take a look at the email message content and have a basic understanding of the data

Ham

This looks like a normal email reply to another person, which is not difficult to classified as a ham:

```
This is a bit of a messy solution but might be useful -

If you have an internal zip drive (not sure about external) and

you bios supports using a zip as floppy drive, you could
use a bootable zip disk with all the relevant dos utils.
```

Hard Ham (Ham email that is trickier)

Hard Ham is indeed more difficult to differentiate from the spam data, as they contain some key words such as *limited time order, Special "Back to School" Offer*, this make it very suspicious !

Hello Friends!

We hope you had a pleasant week. Last weeks trivia questions was:

What do these 3 films have in common: One Crazy Summer, Whispers in the =
Dark, Moby Dick?=20

Answer: Nantucket Island

Congratulations to our Winners:

Caitlin O. of New Bedford, Massachusetts

Brigid M. of Marblehead, Massachusetts

Special "Back to School" Offer!

For a limited time order our "Back to School" Snack Basket and receive =
20% Off& FREE SHIPPING!

Spam

One of the spam training data does look like one of those spam advertisement email in our junk folder:

IMPORTANT INFORMATION:

The new domain names are finally available to the general public at discount prices. Now you can register one of the exciting new .BIZ or .INFO domain names, as well as the original .COM and .NET names for just $14.95. These brand new domain extensions were recently approved by ICANN and have the same rights as the original .COM and .NET domain names. The biggest benefit is of-course that the .BIZ and .INFO domain names are currently more available. i.e. it will be much easier to register an attractive and easy-to-remember domain name for the same price. Visit: http://www.affordable-domains.com today for more info.

Visualization

Wordcloud

Wordcloud is a useful visualization tool for you to have a rough estimate of the words that has the highest frequency in the data that you have.

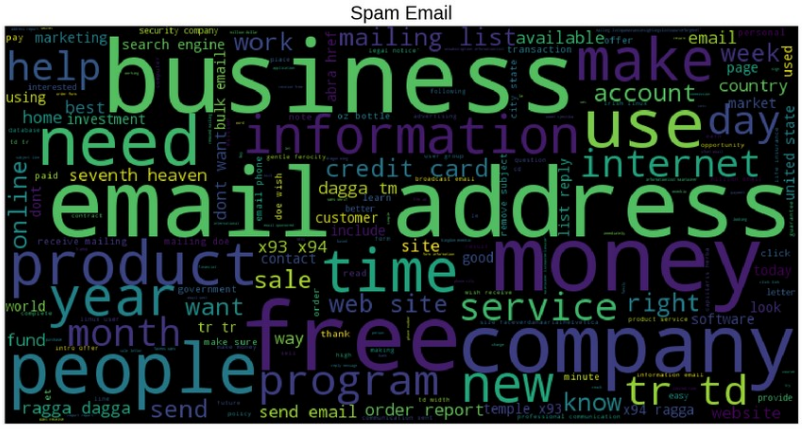


Figure 7: Visualization for spam email

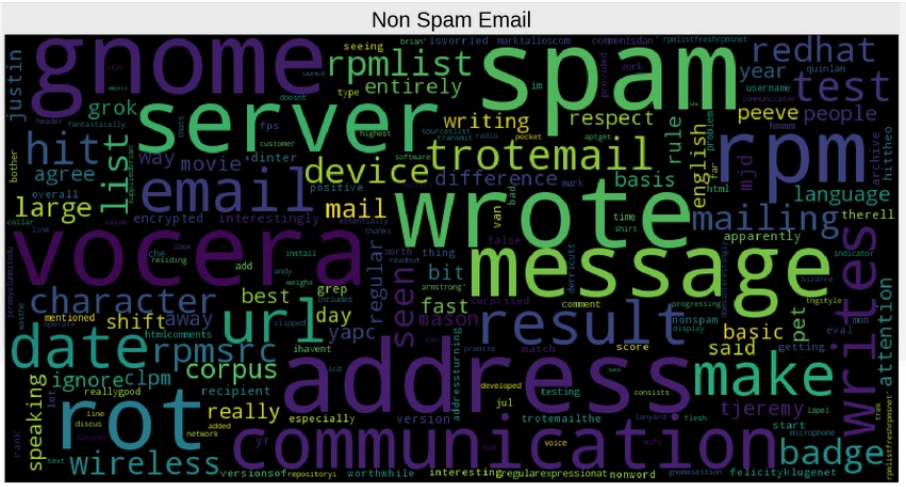


Figure 8: Visualization for non-spam email

From this view, you can see something interesting about the spam email. Many of them have a high number of "spam" words, such as: free, money, product, etc. Having this awareness could help us make a better decision when it comes to designing a spam detection system.

One important thing to note is that the word cloud displays only the frequency of words, not necessarily the importance of words. It is therefore necessary to do some data cleaning, such as removing stop words, punctuation and so on, from the data before visualizing it.

N-grams model visualization

Another technique of visualization is to use the bar chart and display the frequency of the most visible words. N-gram means how many words you consider to be a single unit when you calculate the frequency of words.

I have shown an example of 1-gram and 2-gram in Figure 9. You can definitely experiment with a larger n-gram model.

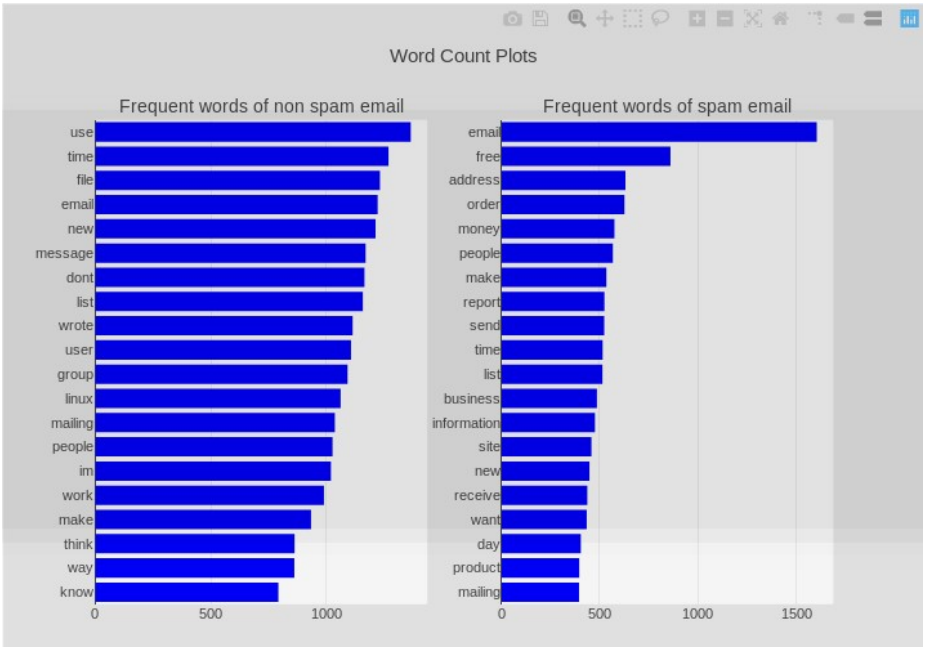


Figure 9: Bar chart visualization of 1-gram model

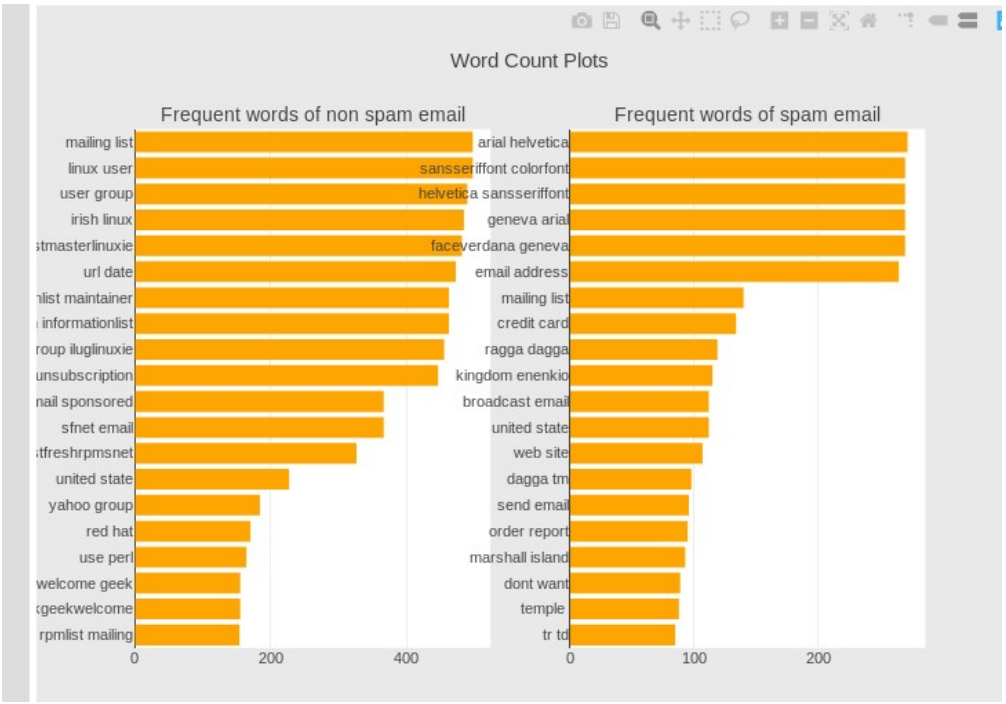


Figure 10: Bar chart visualization of 2-gram model

Train Test Split

It is important to divide your data set into a training set and test set, so that you can evaluate the performance of your model using the test set before deploying it in a production environment.

One important thing to note when splitting the train test is to ensure that the data distribution between the training set and the test set is similar.

What this means in this context is that the percentage of spam emails in the training set and the test set should be similar.

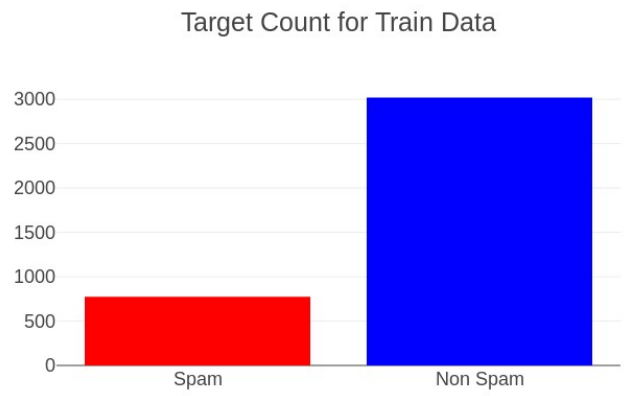


Figure 11: Target Count For Train Data

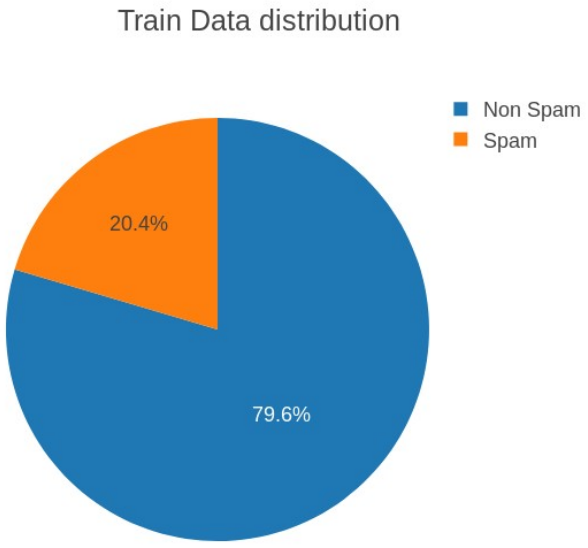


Figure 12: Train Data Distribution

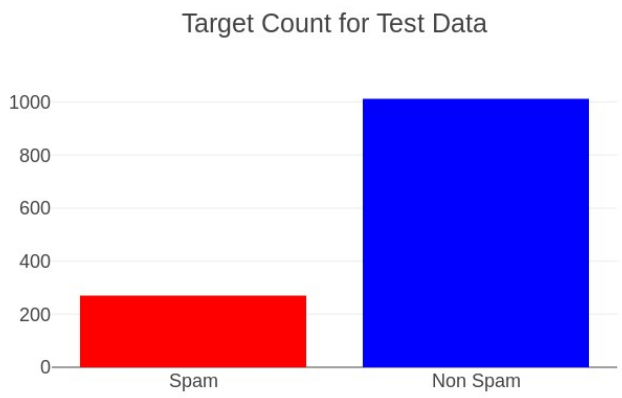


Figure 13: Target Count For Test Data

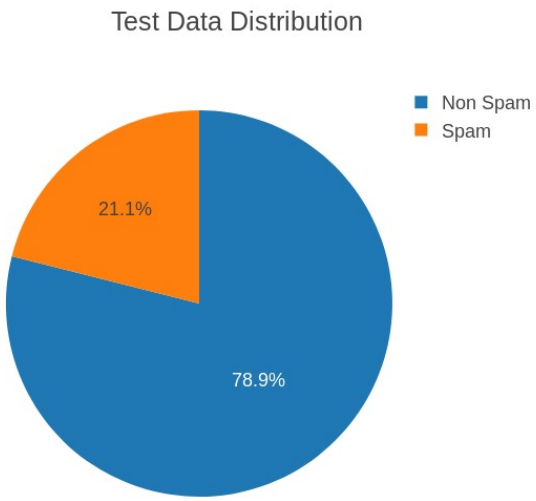


Figure 14: Train Data Distribution

Test Data Distribution

The distribution between train data and test data are quite similar which is around 20–21%, so we are good to go and start to process our data !

Data Preprocessing

Text Cleaning

Text Cleaning is a very important step in machine learning because your data may contains a lot of noise and unwanted character such as punctuation, white space, numbers, hyperlink and etc.

Some standard procedures that people generally use are:

- convert all letters to lower/upper case
- removing numbers
- removing punctuation
- removing white spaces
- removing hyperlink
- removing stop words such as *a, about, above, down, doing* and the list goes on…
- Word Stemming
- Word lemmatization

The two techniques that may seem foreign to most people are word stemming and word lemmatization. Both of these techniques try to reduce words to their most basic form, but they do so with different approaches.

- Word stemming — Stemming algorithms work by removing the word end or beginning, using a list of common prefixes and suffixes that can be found in that language. Examples of Word Stemming for English Words are as follows:

Form	**Suffix**	**Stem**
running	-ing	run
runs	-s	run
consolidate	-ate	consolid
consolidated	-ated	consolid

- Word Lemmatization — Lemmatization is utilizing the dictionary of a particular language and tried to convert the words back to its base form. It will try to take into account of the meaning of the verbs and convert it back to the most suitable base form.

Form	**Suffix**	**Stem**
running	-ing	run
runs	-s	run
consolidate	-ate	consolid

Implementing these two algorithms might be tricky and requires a lot of thinking and design to deal with different edge cases.

Luckily **NLTK** library has provided the implementation of these two algorithms, so we can use it out of the box from the library!

Import the library and start designing some functions to help us understand the basic working of these two algorithms.

```
# Just import them and use it

fromnltk.stem import PorterStemmer
from nltk.stem import WordNetLemmatizer

stemmer = PorterStemmer()
lemmatizer = WordNetLemmatizer()

dirty_text = "He studies in the house yesterday, unluckily,
the fans breaks down"

defword_stemmer(words):
stem_words = [stemmer.stem(o) for o in words]
   return " ".join(stem_words)

defword_lemmatizer(words):
lemma_words = [lemmatizer.lemmatize(o) for o in words]
  return " ".join(lemma_words)
```

The output of word stemmer is very obvious, some of the endings of the words have been chopped off

```
clean_text = word_stemmer(dirty_text.split(" "))
clean_text

#Output
'He studi in the hous yesterday, unluckily, the fan break down'
```

The lemmatization has converted studies -> study, breaks -> break

```
clean_text = word_lemmatizer(dirty_text.split(" "))
clean_text

#Output
'I study in the house yesterday, unluckily, the fan break down'
```

Feature Extraction

Our algorithm always expect the input to be integers/floats, so we need to have some feature extraction layer in the middle to convert the words to integers/floats.

There are a couples ways of doing this, and today I am going to introduce:

1. CountVectorizer
2. TfidfVectorizer
3. Word Embedding

CountVectorizer

First we need to input all the training data into CountVectorizer and the CountVectorizer will keep a dictionary of every word and its respective id and this id will relate to the word count of this word inside this whole training set.

For example, a sentence like *'I like to eat apple and drink apple juice'*

```
fromsklearn.feature_extraction.text import CountVectorizer

# list of text documents

text = ["I like to eat apple and drink apple juice"]

# create the transform

vectorizer = CountVectorizer()

# tokenize and build vocab

vectorizer.fit(text)

# summarize

print(vectorizer.vocabulary_)

# encode document

tvector = vectorizer.transform(text)

# summarize encoded vector

print(vector.shape)

print(type(vector))

print(vector.toarray())

# Output
```

```
# The number follow by the word are the index of the word

{'like': 5, 'to': 6, 'eat': 3, 'apple': 1, 'and': 0, 'drink': 2, 'juice': 4}

# The index relates to the position of the word count array

below

# "I like to eat apple and drink apple juice" -> [1 2 1 1 1 1 1]

# apple which has the index 1 correspond to the word count of 2 in the array
```

TfidfVectorizer

Word counts are good, but can we do better than that? One issue with a simple word count is that some words like 'the,' 'and' will appear many times and really don't add too much meaningful information.

TfidfVectorizer is another popular alternative. Besides taking the word count of every word, words that often appear across multiple documents or sentences, the vector will try to reduce them.

For more information about CountVectorizer and TfidfVectorizer, please read from this great article, which is also where I get the most of my understanding.

Word Embedding

There are a lot of great articles online that explain the details of word embedding and the algorithm to generate them. So here, I'm going to skip most of them and try to give you a rough idea of what it is.

Word embedding is trying to convert a word to a vector format, and this vector represents the position of this word in a higher dimensional space.

For words that have a similar meaning, the cosine distance of the two word vectors will be shorter and closer to each other.

And in fact, these words are vectors, so that you can even perform math operations on them! The end result of this operation is another vector that maps a word. Unexpectedly, some of these operations are producing some amazing result !

Example 1 : King- Man + Woman = Queen

Example 2: Madrid-Spain+France = Paris

Example 3: Walking-Swimming+Swam= Walked

Simply put, word embedding is a very powerful representation of the words and one of the well-known techniques in generating this embedding is *Word2Vec.*

Hooh ! After converting all the sentences into some form of vectors, it comes to the most exciting part of our articles → **Algorithm Implementation !**

Algorithm Implementation

TfidfVectorizer + Naive Bayes Algorithm

The first approach I took was to use the TfidfVectorizer as a feature extraction tool and the Naive Bayes algorithm to make the prediction. Naive Bayes is a simple, probabilistic, traditional machine learning algorithm.

It is very popular even in the past to solve problems such as spam detection. Devi Soni can check out the details of Naive Bayes in this article, which is a concise and clear explanation of the Naive Bayes algorithm theory.

Using the Naive Bayes library provided by the sklearn library saves us a lot of trouble implementing this algorithm on our own. This can be done easily in a few lines of codes

```
fromsklearn.naive_bayes import GaussianNB

clf.fit(x_train_features.toarray(),y_train)

# Output of the score is the accuracy of the prediction

# Accuracy: 0.995

clf.score(x_train_features.toarray(),y_train)

# Accuracy: 0.932

clf.score(x_test_features.toarray(),y_test)
```

We achieve an accuracy of **93.2%.** But accuracy is not solely the metrics to evaluate the performance of an algorithm. Let's try out other scoring metrics and that may help us to understand thoroughly how well this model is doing.

Scoring & Metrics

Disadvantages of accuracy

When it comes to evaluating the performance of a data science model, sometimes accuracy may not be the best indicator.

Some of the problems that we solve in real life may have a very unbalanced class, and using accuracy might not give us enough confidence to understand the performance of the algorithm.

In the email spamming issue that we're trying to solve, the spam data is about 20% of our data. If our algorithm predicts all emails as non-spam, it will achieve 80 percent accuracy.

And for some problem that has only 1% of positive data, predicting all the samples as negative will give them 99 percent accuracy, but we all know that this kind of model is useless in a real-life scenario.

Precision & Recall

Precision & Recall is the common evaluation metrics that people use when they are evaluating class-imbalanced classification model.

Let's try to understand what questions Precision & Recall is trying to answer,

Precision: What proportion of positive identifications was actually correct ?

Recall: What actual proportion of actual positives was identified correctly?

So, **precision** is evaluating, when a model predict something as positive, how accurate the model is. On the other hand, **recall** is evaluating how well a model in finding all the positive samples.

The mathematical equation for precision & recall are as respective

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

TP: True Positive

FP : False Positive

TN: True Negative

FN: False Negative

Confusion Matrix

Confusion Matrix is a very good way to understand the results as true positive, false positive, true negative, and so on is shown in Figure 15.

Sklearn documentation provided a sample code of how to plot a nice confusion matrix to visualize your result. You can check it out here, or you can find the code that

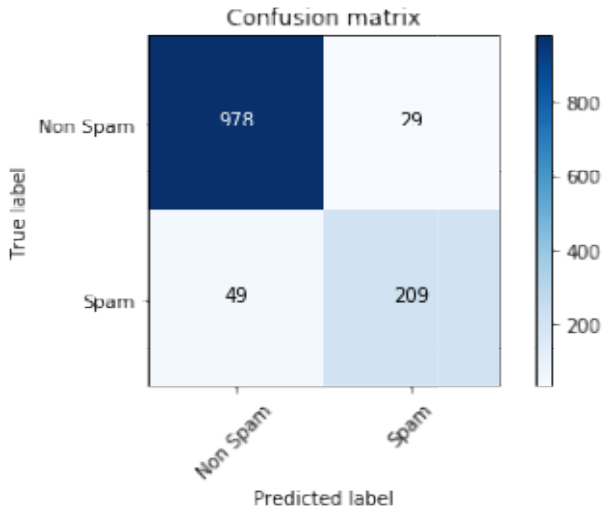


Figure 15: Confusion Matrix

Confusion Matrix of the result

```
Precision: 87.82%
Recall: 81.01%
```

The recall of this model is rather low, it might not be doing a good enough job in discovering the spam email. **How do we do better than this ?**

Summary

In this article, I have showed you all the necessary steps needed in designing a spam detection algorithm. Just a brief recap:

1. Explore and understand your data
2. Visualize the data at hand to gain a better intuition — Wordcloud, N-gram Bar Chart
3. Text Cleaning — Word Stemmer and Word Lemmatization
4. Feature Extraction — Count Vectorizer, TfidfVectorizer, Word Embedding
5. Algorithm — Naive Bayes

6. Scoring & Metrics — Accuracy, Precision, Recall

5. LINEAR REGRESSION AND K-NN FOR FILTERING SPAM

Spamming has become a long and expensive issue that a range of latest directions have recently been explored. a brand new approach to the spam detection filter has been developed. associateswer} developed is an offline program that uses the k-Nearest Neighbor (kNN) algorithmic program and a pre-classified e-mail knowledge set for the training method.

Present spam filtering techniques investigate the utilization of the k-Nearest-Neighbor algorithmic program because the basis for custom spam filters. Many alternative classifiers, like Naive Bayesian Classifier, Random Forest Tree, and Heuristic Rules, are combined to make hybrid spam filtering systems to improve classification efficiency if possible. At a similar time, significant experiments are performed in preprocessing steps to check their impacts on a similar algorithmic program and also the results are reported.

Why Won't Linear Regression Work for Filtering Spam?

Since we're already familiar with linear regression, and that's a resource in our toolbelt, let's start by thinking about what we'd need to do to continue and use linear regression. We already know that's not what we're going to use, so let's talk to you about why. Imagine a dataset or matrix where each row represents a particular email message (it could be labelled by email Id). Now let's make a function of each word in the email — this means that we build a column called "Viagra," for example, and then, for every message that has the word Viagra in it at least once, we put a 1 in; otherwise, we assign a 0. Alternatively, we could put the number of times the word appears. Then each column represents the appearance of a different word.

In order to use linear regression, we need a training set of emails where messages have already been labelled with some output variable. In this case, the tests are either spam or not. We could do this by making spam messages labelled by human evaluators, which is a logical, but time-consuming, solution. Another way to do this will be to take an existing spam filter, such as Gmail's spam filter, and use such names. (Now, of course, if you already have a Gmail spam filter, it's hard to understand why you would want to create another spam filter in the first place, but let's just presume you do it.) If you create a setup, email messages will come through without a mark, so you'd use your model to predict the labels.

The first thing to remember is that your goal is binary (0 if not spam, 1 if spam)—you will not get 0 or 1 using linear regression; you would get a number. Strictly speaking, this choice is not ideal; linear regression is built to model a continuous output, and this is binary.

Basically, this question is a non-starter. We should use a model that is suitable for the data. But if we were to fit it into R, it might still function in theory. R does not test for us whether or not the model is correct. We should go for it, fit a linear model, and then use it to forecast and then select a critical value so that above the predicted value we call it "1" and below it we call it "0."

Yet if we went ahead and tried, it wouldn't work because there are so many variables compared to the observations! We have an order of 10,000 emails with an order of 100,000 pages. This isn't going to work. Technically, this refers to the fact that the matrix in the linear regression equation is not invertible — in truth, it's not even real. Plus, maybe we can't even store it because it's too heavy.

We might be able to restrict it to the top 10,000 words? Or at least we will have an invertible matrix. Even then, there are too many variables and assumptions to make you feel confident about it. With carefully chosen feature selection and domain knowledge, we could restrict it to 100 characters, and that could be enough! But again, we'd still have the problem that linear regression is not the best model for a binary outcome.

How About k-nearest Neighbors?

We're going to get to Naive Bayes early, we say, but let's take a minute to think about trying to use k-nearest neighbors (k-NN) to build a spam filter. We would still have to select features, probably corresponding to terms, and we would probably have specified the value of those features as 0 or 1, depending on whether the word is present or not. Then we'd have to describe when two emails are "next" to each other, based on the terms they both contain.

Again, with 10,000 emails and 100,000 words, we're going to experience a problem different from the non-invertible matrix problem. Namely, the room in which we'd be operating has too many dimensions. Yeah, calculating distances in 100,000-dimensional space requires a lot of computational work. But this isn't the real problem.

The real problem is even more basic: even our closest neighbors are far away. This is called "the curse of dimensionality" and, in this case, makes k-NN a bad algorithm.

6. NAIVE BAYES

Naïve Bayes algorithm is a classification technique based on the application of Bayes' theorem, with the clear assumption that all predictors are independent of each other. Simply put, the presumption is that the existence of a feature in a class is independent of the existence of some other feature in the same class.

For example, a phone can be considered smart because it has a touch screen, an internet facility, a good camera, etc. Since all these features are mutually related, they contribute independently to the likelihood that the phone is a smart phone.

The main interest in the Bayesian classification is to find the posterior probabilities, i.e. the likelihood of a mark given some of the features observed (L features). With the aid of Bayes Theorem, this can be expressed in quantitative form as follows −

```
$$P(L|features)\:=\:\frac{P(L)P(features|L)}{P(features)}$$
```

Here, $(L \mid features)$ is the posterior probability of class.

(L) is the prior probability of class.

$(features|L)$ is the likelihood which is the probability of predictor given class.

$(features)$ is the prior probability of predictor.

Building the Naïve Bayes model in Python

Python Library, Scikit Learn is the most useful library to help us construct a Naïve Bayes model in Python. We have the following three types of Naïve Bayes model under Scikit Learn Python Library-

Gaussian Naïve Bayes

It is the easiest Naïve Bayes classifier to presume that the data from each mark is extracted from a simple Gaussian distribution.

Multinomial of Naïve Bayes

Another useful classifier for Naïve Bayes is Multinomial Naïve Bayes, in which the features are considered to be extracted from a simple Multinomial distribution. This kind of Naïve Bayes is better suited to features that reflect distinct counts.

Bernoulli of the Naïve Bayes

Another essential model is Bernoulli Naïve Bayes, whose features are believed to be binary (0s and 1s). The text classification with the 'bag of words' model can be implemented by Bernoulli Naïve Bayes.

This is an example

Depending on our data collection, we can select either of the Naïve Bayes models mentioned above. Here, we are implementing the Gaussian Naïve Bayes model in Python − we will begin with the necessary imports as follows −

```
importnumpy as np
importmatplotlib.pyplot as plt
importseaborn as sns; sns.set()
```

Now, by using *make_blobs()* function of *Scikit learn*, we can generate blobs of points with Gaussian distribution as follows −

```
fromsklearn.datasets import make_blobs
X, y = make_blobs(300, 2, centers = 2, random_state = 2, cluster_std = 1.5)
plt.scatter(X[:, 0], X[:, 1], c = y, s = 50, cmap = 'summer');
```

Next, for using *GaussianNB* model, we need to import and make its object as follows −

```
fromsklearn.naive_bayes import GaussianNB
model_GBN = GaussianNB()
model_GNB.fit(X, y);
```

Now, we have to do prediction. It can be done after generating some new data as follows −

```
rng = np.random.RandomState(0)
Xnew = [-6, -14] + [14, 18] * rng.rand(2000, 2)
ynew = model_GNB.predict(Xnew)
```

Next, we are plotting new data to find its boundaries −

```
plt.scatter(X[:, 0], X[:, 1], c = y, s = 50, cmap = 'summer')
lim = plt.axis()
plt.scatter(Xnew[:, 0], Xnew[:, 1], c = ynew, s = 20, cmap = 'summer', alpha = 0.1)
plt.axis(lim);
```

Now, with the help of following line of codes, we can find the posterior probabilities of first and second label −

```
yprob = model_GNB.predict_proba(Xnew)
yprob[-10:].round(3)
```

Output

```
array([[0.998, 0.002],
  [1. , 0. ],
  [0.987, 0.013],
  [1. , 0. ],
  [1. , 0. ],
  [1. , 0. ],
  [1. , 0. ],
  [1. , 0. ],
  [0. , 1. ],
  [0.986, 0.014]])
```

Pros & Cons

Pros

The followings are some pros of using Naïve Bayes classifiers −

- Naïve Bayes classification is easy to implement and fast.
- It will converge faster than discriminative models like logistic regression.
- It requires less training data.
- It is highly scalable in nature, or they scale linearly with the number of predictors and data points.
- It can make probabilistic predictions and can handle continuous as well as discrete data.

- Naïve Bayes classification algorithm can be used for binary as well as multi-class classification problems both.

Cons

The following are some cons of using Naïve Bayes classifiers –

- Some of the most important cons of the Naïve Bayes classification is its clear feature independence, since in real life it is almost impossible to have a collection of features that are totally independent of each other.

- Another problem with the Naïve Bayes classification is its 'zero frequency' which means that if a categorical variable has a category but is not observed in the training data set, the Naïve Bayes model will assign a zero probability to it and will not be able to make a prediction.

Applications of Naïve Bayes classification

The following are some common applications of Naïve Bayes classification −

- Real-time prediction − Due to its ease of implementation and quick computation, it can be used to make real-time predictions.
- Multi-Class Prediction − Naïve Bayes Classification Algorithm can be used to estimate the posterior likelihood of several classes of target variables.
- Text classification − Due to the multi-class prediction feature, Naïve Bayes classification algorithms are well suited for text classification. That is why it is also used to address issues such as spam filtering and sentiment analysis.
- Recommendation system − In addition to algorithms such as collaborative filtering, Naïve Bayes creates a recommendation framework that can be used to process undetected information and predict it.

7. DATA WRANGLING

Data Wrangling (also known as Data Munging) is the process of converting data from its original "raw" form into a more digestible format and assembling sets from different sources into a single cohesive whole for further processing. This is shown in Figure 16.

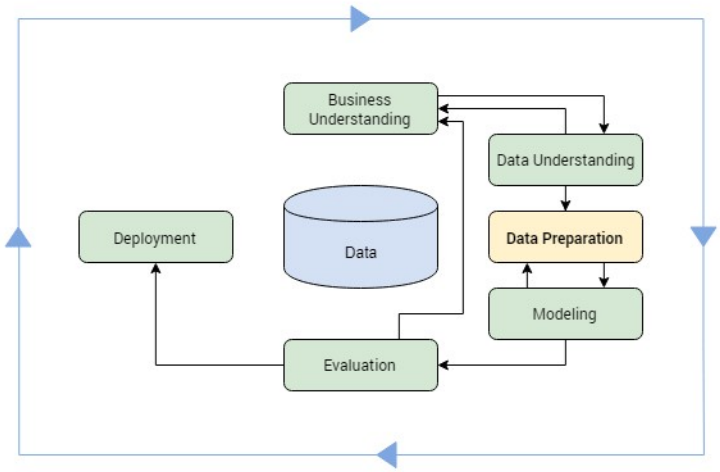


Figure 16: Data Munging

What is “raw data”?

Any repository data (texts, images, database records) that is documented but still to be processed and fully integrated into the system.
The wrangling process can be described as "digesting" data (often referred to as "munging" thus the alternative term "data munging") and making it useful (also usable) for the system. It can be described as a stage of preparation for any other data-related operation.
Data wrangling is generally accompanied by mapping. The term "Data Mapping" refers to a wrangling process element that involves identifying the source data fields in their respective target data fields. While Wrangling is dedicated to the transformation of data, mapping is about connecting dots between different elements.

What is the Purpose of Data Wrangling?

The primary purpose of data wrangling can be described as data collection in a coherent form. In other words, it makes raw data usable. It sets out the substance for further proceedings.

As such, Data Wrangling acts as a stage of preparation for the data mining operation. Process-wise, these two operations are coupled together as you can't do without each other.

You have a basic idea of what Data Wrangling is now let's look at key steps in the Data Wrangling process with basic examples to get you started.

1— Acquiring Data

The first and most critical stage is, of course, data collection and sorting. Or we may suggest that finding your data to further explore this could be the most important step towards achieving your target of answering your questions. However, before finding the data, you need to know the following properties, and you need to be fine with that, because this is only the beginning of a tiring phase.

Not all of the data is generated equal.

While we would like to believe in the truthfulness and accuracy of the data we see, not all the data would meet our standards. When you first explore data, you need to ask yourself a small set of questions:

• Is the source author available if I have any questions or concerns?

• Does the data tend to be updated regularly?

• Does it include details as to how it was obtained and what kind of samples were used for its acquisition?

• Is there any other source where the data can be verified?

If the answer to three or more questions is yes, then you are on the right track, but if the answer to one or two questions is no, you have to dig a little deeper into it.

Fact-Checking

Fact-checking your results, even if it is irritating most of the time, is crucial to the quality of your study. If you have access to any of the resources, such as LexisNexis, Cornell University's arXiv Project, Google's Scholar Quest, and Google's newly released Data Quest, you can learn what others have learned and used in a project or research project. Once you have authenticated and checked your data, it will be easier to assess its validity in the future.

Where to find data

It's obvious that you're not going to ring everyone's phone to collect data. Just like there are numerous sources to verify your data, there are a large number of sources from which you can obtain your data. That includes government data, NGO data, educational or university data, medical or research data, crowdsourced data and so on. Know the best areas to locate datasets for data science ventures.

Now let's jump to our main step, which is Data Cleaning.

2— Data Cleaning

Cleaning up data is no longer a glamorous job, but it is an important part of Data Wrangling. In order to become a Data Cleaning specialist, you must have accuracy, knowledge of the particular field and, above all, patience. Yeah, Patience, sir.

Moving to the technical side, Python can help you clean your data quickly. Assuming you have basic knowledge of Python, we're going to look at some data wrangling with Python in this chapter.

Data Cleanup basics

We need data to perform operations. Here, we will use UNICEF data collection relevant to child labour. Let me give you some insight into the results. Early data sets include Multiple Indicator Cluster Surveys (MICS). Such surveys are household surveys conducted by UNICEF staff and volunteers to help study the living conditions of women and children around the world. Going at the most recent surveys, we extracted some data from Zimbabwe's new MICS for study. You're going to find the updated.csv here.

Identifying Software Cleanup Principles.

Let's look at the mn.csv file from the provided repository connection. The file has raw data and uses codes as headers that look like something like this,

```
"","HH1","HH2","LN","MWM1","MWM2", ...
```

Each of these is a data or problem in the survey. It's not that easy to read, though. With some web scraping expertise, and a little data wrangling with R, you can have another csv with the English

version of the headers. You will find this file in the same repository (mn-headers.csv). Below is the code to remove the headers and the output we need to move forward.

```
from csv import DictReader

data_rdr = DictReader(open('data/unicef/mn.csv', 'rb'))
header_rdr = DictReader(open('data/unicef/mn_headers.csv', 'rb'))

data_rows = [d for d in data_rdr] ❶
header_rows = [h for h in header_rdr]

print data_rows[:5] ❷
print header_rows[:5]
```

❶ This code writes the iterable `DictReader` object into a new list so we can preserve the data and reuse it. We're using the list generator format so we can do it in one simple line of code that's readable and clear.

❷ This prints just a slice of the data, by using the Python list's `slice` method to show the first five elements of our new lists and get an idea of the content.

```
new_rows = [] ❶

for data_dict in data_rows:
    new_row = {} ❷
    for dkey, dval in data_dict.items():
        for header_dict in header_rows:
            if dkey in header_dict.values(): ❸
                new_row[header_dict.get('Label')] = dval ❹
    new_rows.append(new_row) ❺
```

❶ Creates a new list to populate with cleaned rows.

❷ Creates a new dictionary for each row.

❸ Here, we use the dictionary's `values` method instead of iterating over every key and value of the header rows. This method returns a list of only the values in that dictionary. We are also using Python's `in` method, which tests whether an object is a member of a list. For this line of code, the object is our key, or the abbreviated string, and the list is the values of the header dictionary (which contains the abbreviated headers). When this line is true, we know we have found the matching row.

❹ Adds to our `new_row` dictionary every time we find a match. This sets the dictionary key equal to the `Label` value in the header row, replacing those short `Name` values with the longer, more readable `Label` values, and keeps the values set to the data row values.

❺ Appends the new cleaned dictionary we created to our new array. This is indented to ensure we have all the matches before going to the next row.

```
In [8]: new_rows[0]
Out[8]: {
    'AIDS virus from mother to child during delivery': 'Yes',
    'AIDS virus from mother to child during pregnancy': 'DK',
    'AIDS virus from mother to child through breastfeeding': 'DK',
    'Age': '25-29',
    'Age at first marriage/union': '29',...
```

Formatting Data

The most popular aim of data cleanup is to get the unreadable or hard-to-read data to be found in the correct readable format. Python gives us a lot of ways to format strings and numbers. We used the percent r that shows the Python representation of an object in a string or Unicode to test and display our results.

Python also has percent s and percent d string formatter, which represent strings and digits, respectively. These are also used with the print instruction. The format method of Python is advanced, which, according to the official documentation, helps one to define a string and transfer the data as arguments or keyword arguments to the string. Take a closer look at the format.

```
for x in zipped_data[0]:
    print 'Question: {}\nAnswer: {}'.format( ❶
        x[0], x[1]) ❷
```

❶ `format` uses {} to represent where to put the data and the \n newline character to create breaks between the lines.

❷ Here, we pass the first and second values of the question and answer tuple.

You should see something like this:

```
Question: ['MMT9', 'Ever used Internet', 'Have you ever used the Internet?']
Answer: Yes
Question: ['MMT10', 'Internet usage in the last 12 months',
'In the last 12 months, have you used the Internet?']
Answer: Yes
```

I admit that this is pretty hard to read. Want to make it easier to read by cleaning up a bit? Let's do it. There is an abbreviation in the 0-index and a summary of the problem in the 1-index. We just want the second part of it, so here it is,

```
for x in zipped_data[0]:
    print 'Question: {[1]}\nAnswer: {}'.format(
        x[0], x[1]) ❶
```

❶ This time we use the ability to single out the index in the format syntax 1, making the output more readable.

Let's see what output we get:

```
Question: Frequency of reading newspaper or magazine
Answer: Almost every day
Question: Frequency of listening to the radio
Answer: At least once a week
```

Now the output is fairly readable and easy to understand. Hooray.

Finding Outliers

Seeking poor data or strangers is potentially one of the most challenging jobs. You do have to bear in mind that you need to clean up the data and not exploit it. For example, our UNICEF survey dataset follows a standard question format.

This is a good sign that the data is a good sample. But what if we find that volunteers only interviewed families in urban areas and left rural areas, this could result in an error of selection or sampling. Based on your sources, you can decide what bias your dataset may have.

Apart from discovering which data bias is being used, you can find outliers by simply if-not statements. But they struggle most of the time in large data sets. For example, if we search our entire data set for missing data by if-not statements, it would look like this. But you're not going to find any significant missing data points.

```
for row in zipped_data: ❶
    for answer in row: ❷
        if answer[1] is None: ❸
            print answer
```

❶ This time, we loop over every row in our dataset instead of just the first entry.

❷ We remove the `[0]` from our previous example, as we have each row as its own loop.

❸ For example's sake, here we test if we see any `None` types. This will tell us if there are null data points, but won't tell us if we have zeros or empty strings.

Instead, let's try to find our data bias *NA,* which stands for Not Applicable.

Let's see if there is a preponderance of *NA* answers for any specific questions:

```
na_count = {} ❶

for row in zipped_data:
    for resp in row:
        question = resp[0][1] ❷
        answer = resp[1]
        if answer == 'NA': ❸
            if question in na_count.keys(): ❹
                na_count[question] += 1 ❺
            else:
                na_count[question] = 1 ❻

print na_count
```

❶ Defines a dictionary to keep track of questions with *NA* responses. Keeping the data in a hashed object (like a dictionary) allows Python to quickly and easily query the members. The questions will be the keys and the values will hold the count:

❷ Stores the second entry from the first part of the tuple (the description of the question) in `question`. The first entry (`[0]`) is the shorthand title and the last entry (`[2]`) is the question the surveyors asked, which is not always available.

❸ Uses Python's equivalency test to find *NA* responses. If we cared about more than one way to write *NA*, we might use something like `if answer in ["NA", "na", "n/a"]:` to accept a variety of written responses with the same meaning.

❹ Tests if this question is already in the dictionary by testing if it is in the keys of the dictionary.

❺ If the question is already in the keys, this code adds 1 to the value using Python's += method.

❻ If it is not a member of the dictionary yet, this code adds it to the dictionary and sets its count value to 1.

If you're going along with the post, you'll see that there are a lot of NA responses in the results. Have you got an exact number? Tell us about that in the comments. So now that you've found out about the outliers, you know what to do. Kick them out of here. If you know your way around Python, you know that it only takes one line of code to replace the entire NAS.

Advanced Option

APIs

However fancy it might sound, don't trust me. The API is a structured way to exchange data on the Internet. Many websites share data through the endpoints of the API. Many, but not limited to, are Facebook, LinkedIn, the World Bank, the US Census.

An API may be as simple as a data response to a request, but it is uncommon to find APIs with just that feature. Most of the APIs have other useful features. These features can include several API request (REST or streaming) methods. Let's understand, for example, that the twitter API comes in two forms: REST and Streaming. REST stands for Representational State Transfer and is intended to establish consistency in the architecture of the API, while some real-time applications provide streaming APIs.

Data Wrangling Machine Learning Algorithms

- Overall, the following types of machine learning algorithms are at play: • Supervised ML algorithms are used to standardize and integrate disparate data sources: ARY Classification is used to classify established patterns; ARY Normalization is used to flatten independent data set variables and to restructure data in a more coherent manner.
- Unsupervised ML algorithms are used for exploring unmarked data: Clustering is used to detect distinct patterns.

How Data Wrangling solves major Big Data / Machine Learning challenges**?**

Data Exploration

The most important consequence of data mapping in the data processing project is exploratory. This helps you to understand what kind of data you have and what you should do with it.

While it seems rather obvious — more often than not, this stage is skewed for the sake of seemingly more efficient manual approaches.

Unfortunately, these methods also leave out and skip a lot of useful insights into the essence and structure of the data. In the end, you will be required to do it correctly to make more data processing operations feasible.

Automated Data Wrangling goes through data in a variety of ways and provides many more information that can be of interest to business operations.

Unified and Structured Data

It's safe to say that data still comes in as a beautiful mess of various sizes and shapes. Although you might have a sense of knowing "what it is" and "what it is for "— data, as it is in its original form, raw data is largely worthless if it is not structured correctly beforehand.

Data wrangling and subsequent visualization of data segments and frames in a manner that would better serve its purpose. This makes datasets freely accessible for the purpose of gaining some insight into any developing mission.

On the other hand, clearly structured data makes it possible to merge several data sets and slowly make the system more efficient.

Data Clean-up from Noise / Errors / Missing Information

Noise, errors and missing values are normal in any data set. There are a variety of explanations for this:
• Human error (so-called soapy eye);
• Unintended mislabeling;
• Technological glitches;
its effect on the efficiency of data processing operations is well known — leads to low efficiency of results and, consequently, to less productive business operations. Noise, noisy data is much worse for machine learning algorithms. If these datasets are educated in an algorithm — they can be made useless for its purposes. That's why the data wrangling is there to the right of the wrongs and to make it the way it was meant to be. In the form of data cleaning, wrangling is performed as follows.

- Data audit

  — Anomaly and error / contradiction identification by statistical and database approaches.

- Process specification and execution

  — The causes of irregularities and errors are analyzed. After defining their origin and effect in the sense of a particular workflow

  — The element is corrected or removed from the data collection.

- Post-processing control

  — After the clean-up is carried out

  — Results of the clean workflow are reassessed. In the event of further complications

  — A new cleaning cycle can occur.

Minimized Data Leakage

Data Leakage is also considered to be one of the greatest problems in Machine Learning. And since ML algorithms are used for data processing — the threat is increasing exponentially. The thing is — prediction relies on the accuracy of the data. And if the measured prediction is based on unknown data — this prediction is as good as a wild guess.

What's the Data Leakage? The term applies to situations where the training of the predictive model uses data outside the training data set. So-called "outside evidence" may be something unverified or unmarked for model training.

The direct consequence of this is an inaccurate algorithm that makes you inaccurate results that may have a significant impact on you.

Why is this happening? The typical trigger is a chaotic data structure with no simple boundary signals where what is and what is for what. The most popular form of data leakage is when data from the test set is bleeded into the training data set.

Extended data wrangling and data mapping techniques will help to mitigate its capacity and, ultimately, to neutralize its effect.

Data Wrangling Tools

Basic Data Munging Tools

- Excel Power Query / Spreadsheets — the most basic structuring tool for manual wrangling.
- OpenRefine — more sophisticated solutions, requires programming skills
- Google DataPrep - for exploration, cleaning, and preparation.
- Tabula — swiss army knife solutions — suitable for all types of data
- DataWrangler — for data cleaning and transformation.
- CSVKit — for data converting

Data Wrangling in Python

1. Numpy (aka Numerical Python)—the most simple package. Lots of functionality for n-array and matrice operations in Python. The library provides vectorization of mathematical operations of the NumPy array type, which increases performance and thus speeds up execution.

2. Pandas — designed for quick and simple data analysis. Useful for data structures with axes numbered. Explicit data alignment avoids common errors arising from misaligned data coming from different sources.

3. Matplotlib — Module of Python Visualisation. Perfect for line graphs, pie graphs, histograms, and other technical grade numbers.

4. Plotly — for interactive, publication-quality graphics. Excellent for line plots, scatter plots, area charts, bar charts, error bars, box plots, histograms, heat maps, subplots, multi-axis, polar graphs, and bubble charts.

5. Theano — a library for numerical computation close to Numpy. This library is designed to efficiently describe, optimize and evaluate mathematical expressions involving multi-dimensional arrays.

Data Wrangling in R

1. Dplyr-fundamental data-munging R bundle. Tool for Supreme Data Framing.Particularly useful for the operation of categories of data.

2. Purrr-good for listing and error-checking features.

3. Splitstackshape-an oldie, but a goldie. Great for the shaping of full

4. Splitstackshape-an oldie, but a goldie. Perfect for shaping complex data sets and making visualization easier.
5. JSOnline-a simple and quick scanning device.
6. Magrittr-good for wrangling scattered sets and putting them in a more cohesive shape.

8. FEATURE GENERATION

8.1 INTRODUCTION

The 2004 Text Retrieval Conference (TREC) Genomics Track was divided into two main tasks: categorization and ad hoc retrieval. The categorization task consisted of a document triage subtask and an annotation subtask to detect the presence of evidence in the document for each of the three main Gene Ontology (GO) code hierarchies. Our work focused on the document triage subtask. We also participated in the ad hoc retrieval task.

8.2 BACKGROUND

The classification of documents is a common problem in biomedicine. Training a support vector machine (SVM) on vectors generated from stemmed and/or stopped document word counts has proven to be a simple and generally efficient method (Yeh et al., 2003).

However, we agreed that the triage issue posed here had some distinctive features that would entail a modification of the standard approach. First, it was understood that the number of true positive results in both the training and the test set was low, about 6-7%. Second, the utility function chosen as the record metric was heavily weighted to reward recall and not precision.

This was based on an overview of the existing working procedures of the annotators at the Mouse Genome Institute (MGI) and an estimate of how they actually view false negative and false positive classifications. The official utility feature weights a false negative as 20 times more extreme than a false positive. Using this metric, the existing work procedure of MGI, which reads all the documents in the test set, has a value of 0.25. In fact, the training and evaluation samples were not randomly drawn from the same survey, but rather obtained from documents released in two consecutive years. While this is a more practical simulation of the framework as it would be applied to MGI, it poses the question of how well the features derived from one year of literature reflect the literature of subsequent years. As a result of these problems, our approach included a rich collection of features, statistically dependent selection of features, multiple classifiers, and an analysis of how well the features extracted from the 2002 corpus reflected the documents in the 2003 corpus.

8.3 SYSTEM AND METHODS

We have tackled the question of triage in four stages: the generation of features, the selection of features, the collection and training of classifiers and, finally, the classification of test documents. Only the training package was used to complete the first three measures. The final step in collecting the test was taken to produce the results submitted. During the development of the program, we used ten-fold cross-validation on the training set to compare approaches and set program parameters. It included the execution of the first two phases of the entire training series. Then 90% of the training data was used to train the classifiers, which were then applied to the remaining 10% of the training data. This has been repeated nine times, so that all the training data has been classified once. The findings were then aggregated to compute cross-validation metrics for the training corpus. Figure 17 displays this phase diagrammatically.

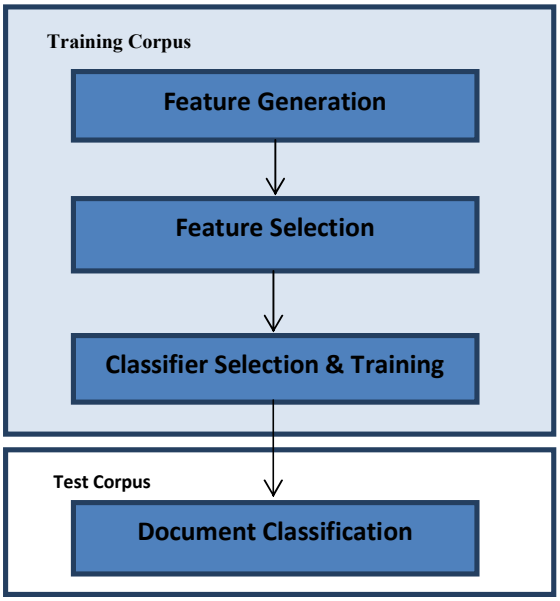


Figure 17: Step-wise approach to test classification

1. Feature generation

The full text corpus with SGML mark-up offered an opportunity to explore the use of several types of features. While many text classification methods view text as a "bag-of-words," we have opted to use the information contained in the SGML mark-up to generate unique section type features. Since we merged features that could occur several times in a single document with features that could only occur once, after some initial testing, we decided to view each feature as binary, that is, each feature was either present in a document or absent. One type of function we created consisted of pairs of section names and stemmed words using the Porter stemming algorithm. Upon applying a stop list of the 300 most common English words, the individual parts of the text collected were coded to include abstract sections, body paragraphs, captions and section titles. We have created a similar hybrid section, stammedword features, using the stopped and stammed section title in conjunction with the stopped and stammed words in the named section. In addition, we have downloaded the related MEDLINE documents from PubMed. For each post, the corresponding MeSH headings have been extracted. We included MeSH-based features based on the full MeSH headings, the MeSH main headings and the MeSH subheadings. Finally, we included features based on details in the reference section of each text. The key author of each reference was taken as a form of attribute. We also

included a long form of references as a feature sort, including the primary author, journal name, length, year, and page number. Running the feature generation process on a full set of 5837 training documents created over 100,000 potentially useful features along with a count of the number of documents containing each feature.

2. Feature selection

We have opted to use the Chi-square selection method to pick the features that best differentiated between positive and negative documents in the training corpus. The 2x2 Chisquare table is constructed as shown in Table 1, using the number of documents obtained in the previous stage. During machine tuning, an alpha value of 0.025 was found to produce the best results. Using this value as a cut-off, 1885 features were selected as the most important. The number and type of each feature found significant and used in the following steps are shown in Table 2.

3. Classifier selection and training

Three specific classifiers were applied to the problem: Naive Bayes, SVM, and Voting Perceptron. Although it is widely thought that the best classifiers are based on Vapnik's SVM method (Vapnik, 2000), the distinctive aspects of the current classification issue discussed above inspired us to apply three different classifiers. By using the same feature set with each of the classifiers, this helped us to compare the efficiency of the classifier algorithms with the particular requirements of the triage function.

Table 1. 2x2 arrangement for testing feature significance

		Feature is the one under test?	
		Yes	**No**
Training document is triage positive?	**Yes**	Number of times feature seen in positive documents	Number of times other features seen in positive documents
	No	Number of times feature seen in negative documents	Number of times other features seen in negative documents

Table 2. Number and type of features used

Feature type	Number significant
Abstract stemmed words	127
Body paragraph stemmed words	778
Caption stemmed words	291
MeSH full headings	15
MeSH main headings	52
MeSH subheadings	1
Author of referenced work	35
References	4
Section title stemmed words	69
Section title with stemmed section words	509
Total number of features: significant features	**1885**

Neither Naive Bayes nor the implementation of the SVM we used, SVMLight (Joachims, 2004), offered adequate means to change the low frequency of positive and the high value of true positive relative to false positive. We used our own implementation for Naive Bayes. Naive Bayes sets a classification probability threshold that can be used to switch between precision and recall. Nonetheless, this is an indirect form of compensation, and in practice, for this task of classification, we found that raising the likelihood threshold did not have a meaningful impact.

We fully expected SVMLight to perform better than Naive Bayes, as it contained a cost factor parameter that could be changed to require unequal penalties for false positives and false negatives. Nevertheless, we found that the amount of impact of this parameter was limited and insufficient to account for the 20 difference between the cost of false positives and negatives. Since neither Naive Bayes nor one of the most common SVM implementations addressed our requirements, something else was required.

An analysis of the classification literature reveals considerable progress in adjusting the classical Rosenblatt Perceptron algorithm (Rosenblatt, 1958) to achieve efficiency at or near SVM for several problems. One algorithm in particular, the Voting Perceptron algorithm (Freund and Schapire, 1999), has quite good efficiency, is very fast and easy to implement. Although the algorithm as published does not provide a way to account for asymmetric false positive and negative penalties, we have made a change to the algorithm that does. Perceptron is basically an equation for a linear combination of the values of the set of features.

For every element in the feature set, there is one term in the perceptron plus an optional bias phrase. The document is defined by taking the dot product of the document's feature vector with the perceptron and adding it to the bias word. When the result is greater than zero, the document is classified as positive, if it is less than or equal to zero, then the document is classified as negative. The original algorithm of Rosenblatt trained the perceptron by applying it to each sample in the training results.

If the sample was wrongly labeled, the perceptron was changed by adding or subtracting the sample back into the perceptron, adding when the sample was a true positive, and subtracting when the sample was a true negative. Over a large number of training samples, the perceptron converges on a solution that better approximates the distinction between positive and negative documents in the training package. Freund and Schapire improved the performance of the perceptron by modifying the algorithm to produce a series of perceptrons, each of which makes a prediction about the class of each document and receives a number of "votes" depending on how many documents the perceptron has correctly classified in the training set.

The class with the most votes is the class allocated to the paper. Our extension to this algorithm is based on a specific modification of the perceptron learning rate for false negatives and false positives. Although incorrectly classified samples are directly added or subtracted back to the perceptron in the typical implementation, we first multiply the sample by a factor known as the learning rate. In addition, we use separate learning thresholds for false positives and false negatives. Given the concept of the utility function, we predicted that the optimal learning rate for false negatives will be around 20 times that for false positives.

In reality, that's what we noticed during the training. We used 20.0 for false negatives, and 1.0 for false positives. The training corpus was applied to each of the three classifiers. Ten-fold cross-validation has been used to optimize all free parameters. The Naive Bayes classifier had one free parameter, the threshold for the probability classification. It was left to the default value of 0.50. The selected SVM-Light classifier settings used a linear kernel and a cost factor of 20.0. The Voting Perceptron classifier was used with a linear kernel and the learning rate was given above. For each of the three approaches, a trained classifier model was developed.

4. Classification of test documents

Eventually, the test corpus was added to the models developed by the Naive Bayes, SVM and Voting Perceptron classifiers. It's done in two steps. The documents in the study sample were first examined for the presence or absence of significant features observed during the selection process. This has generated a vector function for each test paper. The documents were then categorized by applying each of the three qualified classifiers.

Table 3. Performance of classification system on test and training corpi

Corpus	Classifier	Precision	Recall	Fscore	Utility
Training corpus	Naive Bayes	0.1556	0.7650	0.2587	0.5577
	SVMLight	0.3140	0.5550	0.4010	0.4940
	Voting Perceptron	0.1857	0.8453	0.3045	0.6600
Test corpus	Naive Bayes	0.1290	0.6548	0.2155	0.4337
	SVMLight	0.2309	0.3524	0.2790	0.2937
	Voting Perceptron	0.1714	0.6571	0.2719	0.4983

5. Evaluation of conceptualdrift

One critical problem in applying text classification systems to documents of interest to curators and annotators is how well the available training data reflect the documents to be categorized. When classifying the biomedical text, the available training manuals must have been written in advance of the text to be categorized. However, because of its very existence, the field of science shifts over time, as does the vocabulary used to explain it.

How easily written science literature changes has a direct effect on the creation of biomedical text classification systems in terms of how the features are developed and chosen, how much the systems need to be re-trained, how much training is required, and the overall performance that can be expected from such systems can be affected. Throughout biomedical literature, we decided to begin to understand this significant topic of conceptual drift.

In order to determine how well the features chosen from the training collection reflected the information that was relevant in classifying the document in the test collection, we took additional steps in producing the features and selecting the test collection. The exact same method and parameters were used for the collection of tests as for the collection of testing. We then calculated how well the training collection feature set reflected the test collection feature set by the computational similarity metrics between the two sets (Dunham, 2003).

9. FEATURE SELECTION ALGORITHMS

Feature selection is also called selection of variables or selection of attributes.

It is the automated selection of attributes in your data (such as columns in tabular data) that are most important to the issue of predictive modeling that you are working on.

"Feature Selection... is the method of selecting a subset of the applicable features for use in model construction.*"*

The selection of features is distinct from the reduction of dimensionality. Both methods aim to minimize the number of attributes in the dataset, but the dimensional reduction approach does so by introducing new combinations of attributes, while the feature selection methods include and remove attributes present in the data without modifying them.

Examples of dimensionality reduction methods include Principal Component Analysis, Singular Value Decomposition and Sammon'sMapping.

"Feature selection is itself useful, but it mostly acts as a filter, muting out features that aren't useful in addition to your existing features".

9.1 The Problem the Feature Selection Solves

Feature selection methods allow you to build an effective predictive model in your task. We support you by selecting features that will give you as good or better accuracy while needing less data.

Feature selection approaches may be used to recognize and delete unwanted, obsolete and redundant attributes from data that do not contribute to the accuracy of the predictive model or can potentially reduce the accuracy of the model.

Fewer attributes are preferable because they reduce the complexity of the model, and a simpler model is easier to understand and describe.

The goal of variable selection is threefold:

1. to enhance predictor efficiency,
2. to provide quicker and more cost-effective predictors,
3. and to provide a better understanding of the underlying process that generated the data.

9.2 Feature Selection Algorithms

There are three general classes of feature selection algorithms:

1. Filter methods,
2. Wrapper methods,
3. Embedded methods.

Filter Methods

Filter feature selection approaches use a statistical test to assign a score to each element. The features are ranked by the score and either selected to be stored or deleted from the dataset. Methods are mostly univariate and consider the function separately or in relation to the dependent variable.

Examples of some of the filter methods include the Chi squared test, information gain and correlation coefficient ratings.

Wrapper Methods

Wrapper approaches consider the collection of a set of features as a search problem where various combinations are planned, evaluated and compared to other combinations. A predictive algorithm used to test a combination of features and give a score based on the accuracy of the formula.

The search process may be methodical, such as a best-first search, stochastic, such as a random hill-climbing algorithm, or heuristics, such as forward and backward passes, may be used to add and remove features.

An example if the wrapper approach is a recursive elimination algorithm.

Embedded Methods

Embedded methods learn which features better contribute to the accuracy of the model when the model is being built. Regularization methods are the most common type of embedded feature selection methods. Regularization methods are often called penalization methods that apply additional constraints to the design of a predictive algorithm (such as a regression algorithm) that moves the model towards lower complexity (lower coefficients). Examples of regularization algorithms include LASSO, Elastic Net and Ridge Regression.

9.3 How to Choose a Feature Selection Method for Machine Learning

Feature selection is a method that reduces the number of input variables when designing a predictive model. It is beneficial to reduce the number of input variables, both to reduce the computational cost of modeling and, in some cases, to increase the efficiency of the model.

Feature-based feature selection approaches include analyzing the relationship between each input variable and the target variable using statistics and choosing those input variables that have the best relationship with the target variable. These methods can be fast and efficient, although the choice of statistical measures depends on the data type of both input and output variables. As such, it may be difficult for a machine learning practitioner to choose an appropriate statistical measure for a data set when choosing filter-based apps.

1. Feature Selection Methods

Feature selection approaches are designed to reduce the number of input variables to those deemed most useful for the model in order to predict the target variable.

Some predictive modeling problems have a large number of variables that can slow down the creation and training of models and require a large amount of machine memory. In addition, the output of certain models can be degraded by adding input variables that are not important to the target variable.

There are two major types of feature selection algorithms: the wrapper method and the filter method.

1. Wrapper Feature Selection Methods.
2. Filter Feature Selection Methods.

1. **Wrapper feature selection** approaches generate several models with various input features subsets and pick those features that result in the best output model according to the performance metric. These methods are not concerned with variable types, although they can be computationally costly. RFE is a good example of a method for selecting a wrapper function.

   Wrapper methods test several models using procedures that add and/or extract predictors to find the optimum combination that maximizes model efficiency.

2. **Filter feature selection** approaches use statistical techniques to test the relationship between each input variable and the target variable, and these scores are used as the basis for selecting (filtering) the input variables that will be used in the model. Filter methods test the importance of predictors outside the predictive models, and then only model predictors that pass any criterion.

   Correlation style statistical measurements between input and output variables are widely used as the basis for filter function selection. As such, the choice of statistical measures is highly dependent on variable data types. Popular data types include numerical (such as height) and categorical (such as label), although each can be further subdivided as integer and floating point for numerical variables, and boolean, ordinal, or nominal for categorical variables.

UNIT- III (Mining Social-Network Graphs & Data Visualization)

1. WHAT IS SOCIAL NETWORK GRAPH?

The social graph is a diagram showing the relations between individuals, groups and organizations within the social network. The term is also used to identify a person's social network. A social graph appears as a set of network nodes connected to a line when it is depicted as a map.

Essential Characteristics of a social network are:

• Building Block

◊ Entities (Entities are typically people)

• At least one relationship between entities of a network

• Assumption: no randomness

Naturally modeled as undirected graphs:

- Entities ⟶ Nodes
- Nodes connected if there is a relationship between entities
- Degree ⟶ labeling the edges

Facebook/Twitter is not only Networks that are Social. Other than "friends" networks, there are many examples that exhibit locality of relationships:

- Telephone Networks
- Email Networks
- Collaboration Networks
- Airport Networks

Graphs with Several Node Types

Entities can be of various types (e.g. Facebook has users, accounts, and a network). A natural way of representing the k-part graph Consists of k sets of nodes (no edges between nodes of the same set) shown in Figure 1.

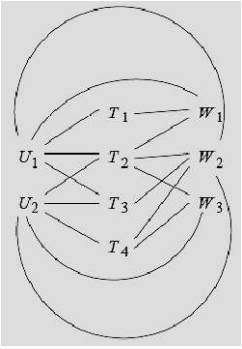


Figure 1: k-partite graph

How do you define a distance measure for a social network?

- If we were to implement standard clustering techniques, our first step would be to establish distance measurement.
- Question: What would be the correct distance calculation for the graph below in Figure 2? Hint: The smaller the nodes, the better the nodes.

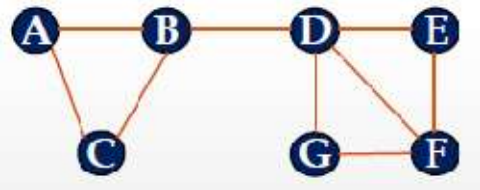


Figure 2: Distance measure for the graph

Problems in Hierarchical Clustering

Consider the graph in the Figure 3 below.

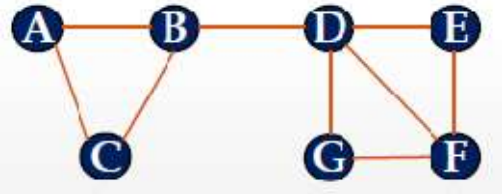


Figure 3: Hierarchical Clustering

1. Based on the geometry of the graph, identify some of the large clusters that we can obtain.

2.By using hierarchical clustering (bottom-up approach), what is the probability that we cluster B and D together first?

Problems with Point Assignment (*k*-Means Clustering)

• If we start by choosing two points at random, they may be in the same cluster.

• If we start by choosing a point at random and selecting a point farthest away from it, we can still screw it up – an example?

• Even though we have two rational starting points, e.g. B and F, how are we going to delegate D?

Betweenness Definition: The betweenness of the edge (a, b) is defined as the number of pairs of nodes (x, y) such that the edge (a, b) is on the shortest path between them.

Properties:

High Betweenness is bad!!

Interpretation: High scores suggest that (a,b) runs between two different communities

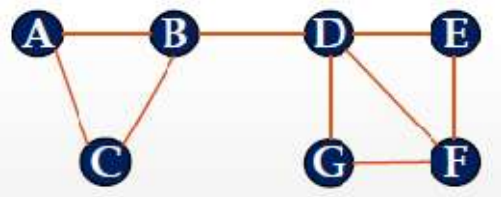


Figure 4: Example/Exercise: Calculate the betweenness

Betweenness – An Observation

The distance score for the edges of the graph behaves like (i.e. not exactly) a distance measure on the nodes of the graph shown in Figure 5. Therefore, we can cluster by eliminating the edges in a rising order of space!!

Example:

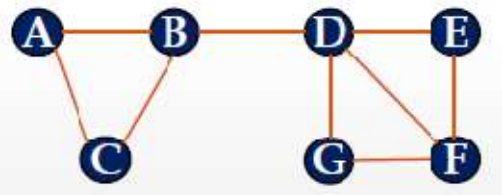


Figure 5: Distance measure graph

What makes a good partition?

A good partition has the following properties:

1. Maximize the number of within-group connections
2. Minimize the number of between-group connections

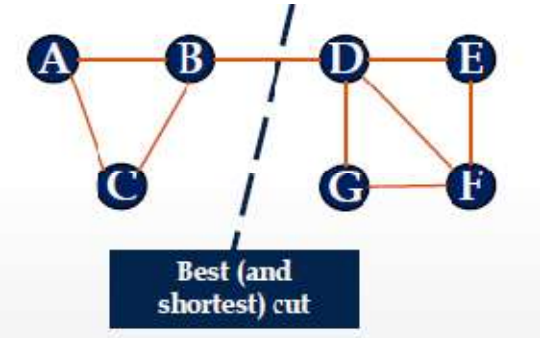


Figure 6: Partition of graph

Normalized Cuts

A proper definition of a "good" cut must produce balanced sets.

Suppose we want to divide the figure into two distinct sets of nodes: S and T, then the normalized cutis:

$$ncut(A,B) = \frac{cut(A,B)}{vol(A)} + \frac{cut(A,B)}{vol(B)}$$

Example: Identify the *ncut*:

- Smallest cut
- Optimal cut

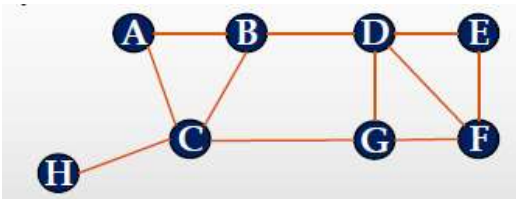


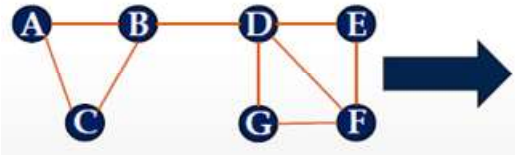
Figure 7: Normalized cuts graph

Some Matrices that Describe the Graphs

1. Adjacency Matrix (*A*)

- $n \times n$matrix
- $A=[a_{ij}]$, a_{ij}= 1 if edge exists between node *i*and *j*

  0 otherwise



	A	B	C	D	E	F	G
A	0	1	1	0	0	0	0
B	1	0	1	1	0	0	0
C	1	1	0	0	0	0	0
D	0	1	0	0	1	1	1
E	0	0	0	1	0	1	0
F	0	0	0	1	1	0	1
G	0	0	0	1	0	1	0

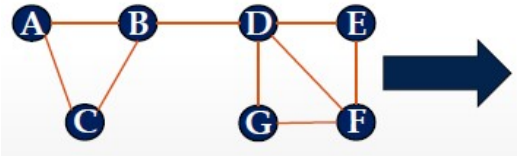
Figure 8: Adjacency Matrix

Important Property:

- Symmetric Matrix

2. Degree Matrix (*D*)

- $n \times n$matrix
- $D=[d_{ii}]$, d_{ii}= degree of node



	A	B	C	D	E	F	G
A	2	0	0	0	0	0	0
B	0	3	0	0	0	0	0
C	0	0	2	0	0	0	0
D	0	0	0	4	0	0	0
E	0	0	0	0	2	0	0
F	0	0	0	0	0	3	0
G	0	0	0	0	0	0	2

Figure 9: Degree Matrix

Important Property:

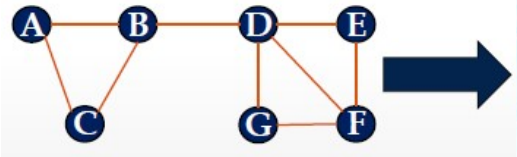
- Diagonal

3. Laplacian Matrix (*L*)

- $n \times n$matrix
- $L=D-A = [_{lij}]$, $l_{ii}= d_{ii}$

  $l_{ij}=$ -1 if edge exists

  0 otherwise

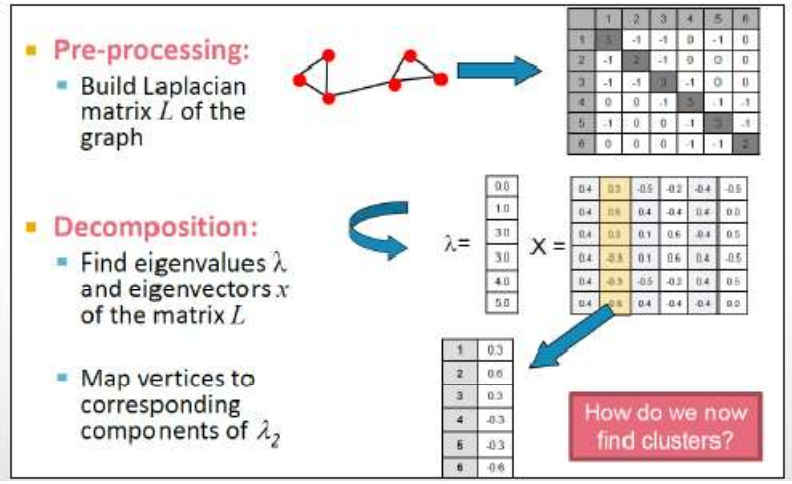


	A	B	C	D	E	F	G
A	2	-1	-1	0	0	0	0
B	-1	3	-1	-1	0	0	0
C	-1	-1	2	0	0	0	0
D	0	-1	0	4	-1	-1	-1
E	0	0	0	-1	2	-1	0
F	0	0	0	-1	-1	3	-1
G	0	0	0	-1	0	-1	2

Figure 10: Laplacian Matrix

Eigenvalues of the Laplacian Matrix

- Partition based on the smallest eigenvector



2. CLUSTERING OF GRAPHS

Graph clustering is an important subject and deals with graph clustering. The clustering problem data can be represented as a graph where each element to be clustered is represented as a node and the distance between the two elements is modeled by a certain weight on the edge of the node. Therefore, in graph clustering, the elements within the cluster are linked to each other but have no relation to the elements outside the cluster. Also, some of the recently proposed approaches perform clustering directly on graph-based data. Some important solutions to graph-based clusters are contiguity-based clusters and clicks. Figure 11 demonstrates the structure of the cluster.

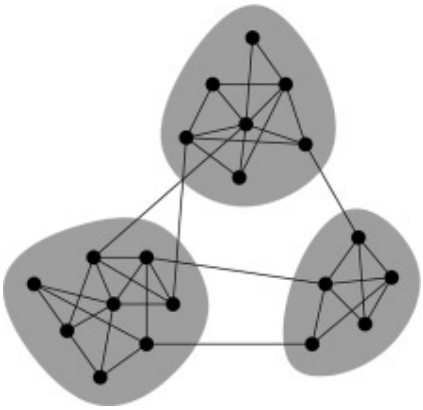


Figure 11: Structure of a cluster

The clustering principle is very useful in different fields of computer science, such as image segmentation and complex network analysis.

In biology and medicine, clustering is also sometimes used to analyze data, for example in the fields of gene expression and protein structure analysis. The clustering method is also used for astronomy.

Graphic Clustering Methods

Let's discuss how to do clustering on a graph. First, we explain the theory behind graph clustering. Then we discuss two general types of graph clustering methods.

To find clusters in a graph, assume that the graph is broken into pieces that each piece is a cluster, such that the vertices within a cluster are well connected and the vertices in separate clusters are linked in a much weaker way. Formally, for a graph, G=(V, E), a split, C=(S, T), is a partitioning of the set of vertices V in G, i.e., V = S3,T and S3,T = approximate. The cut set of the cut is the set of sides, {(u,v) . The size of the cut is the number of edges in the cut. For weighted graphs, the size of the cut is the sum of the weights of the edges in the cut.

What kinds of cuts are perfect for drawing clusters in graphs?

In graphics theory and some network applications, a minimum cut is necessary. A cut is negligible if the cut size is not greater than any other cut size. There are polynomial time algorithms to measure minimum graph cuts. Can we use these algorithms for graph clustering?

Example

Cuts and clusters

Take Graph G in Figure 12. The graph has two clusters: {a, b, c, d, e, f} and {g, h, I j, k}, and one external vertex, l.

Consider cutting C1=({a, b, c, d, e, f, g, h, I j, k},{l}). Just one side, i.e. (e, l), crosses the two partitions generated by C1. The cut set of C1 is therefore {(e, l)} and the scale of C1 is 1. (Note that the size of any cut in the linked graph cannot be less than 1.) As a minimum cut, C1 does not contribute to a good clustering since it only divides the outer graph, l, from the rest of the graph.

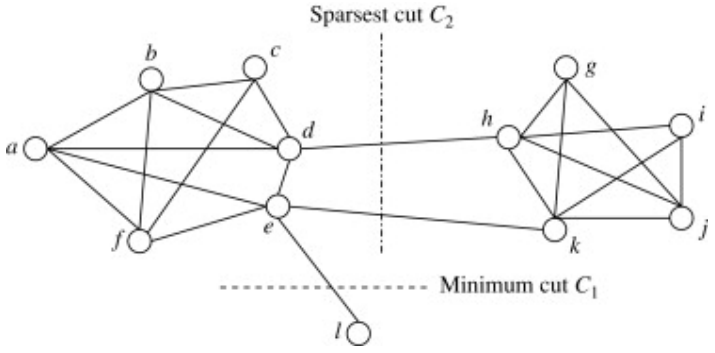


Figure 12: A graph *G* and two cuts

Cut C2=({a,b,c,d,e,f,l},{g,h,i,j,k}) leads to a much better clustering than C_1.

The edges in the cut set of C_2 are those connecting the two "natural clusters" in the graph. Specifically, for edges (*d*, *h*) and (*e*, *k*) that are in the cut set, most of the edges connecting *d*, *h*, *e*, and *k* belong to one cluster.

Theoretically, many graph clustering problems can be seen as finding good cuts, such as sparsest cuts, on the graph. In practice, however, there are a number of challenges:

• **High computational cost:** many graph-cutting problems are computationally costly. For example, the sparsest cut problem is NP-hard. It is also always difficult to find suitable solutions on large graphs. Strong trade-off between efficiency / scalability and quality must be achieved.

• **Complex graphs:** graphs can be more complicated than those listed here, including weights and/or cycles.

• **High dimensionality:** the graph can have many vertexes. In a similarity matrix, the vertex is defined as a vector (a row in the matrix) with a dimensionality that is the number of vertexes in the graph. Graph clustering methods must also accommodate high dimensionality.

• **Sparsity:** A large graph is always sparse, meaning that each vertex connects only a small number of other vertexes on average. A similarity matrix of a large, sparse graph can also be sparse.

There are two types of clustering graph data methods that tackle these challenges. One uses clustering methods for high-dimensional data, while the other is specifically designed for clustering graphs.

The first group of methods is focused on generic clustering methods for high dimensional data. They derive a similarity matrix from a graph using a similarity measure. A generic clustering approach can then be extended to the cluster discovery matrix. Usually, clustering techniques for high-dimensional data are used. For example, spectral clustering methods can be used in several cases, once a similarity matrix has been obtained. Spectral clustering can approximate optimal graph cutting solutions.

The second category of methods is graph-specific. They search the graph to find well-connected components like clusters. Let's look at the SCAN (Structural Clustering Algorithm for Networks) method as an example.

In the case of an undirected line, G=(V, E), for a vertex, u-vi, the u-circuit is name(u)={v, v)-fileE}-file{u}. Using the concept of structural-context similarity, SCAN tests the similarity between the two vertexes, u, v, to the normalized local neighborhood dimension, i.e.

$\sigma(u,v)=|\Gamma(u)\cap\Gamma(v)||\Gamma(u)||\Gamma(v)|$.

The larger the value computed, the more similar the two vertices. SCAN uses a similarity threshold ε to define the cluster membership. For a vertex, u∈V, the *ε-neighborhood* of *u* is defined as Nε(u)={v∈Γ(u)|σ(u,v)≥ε}. The ε-neighborhood of *u* contains all neighbors of *u* with a structural-context similarity to *u* that is at least ε.

The central vertex in SCAN is the vertex within the cluster. In other words, if, where μ is a popularity threshold, the core vertex of the μV is. SCAN develops clusters from the center edge. If the vertex v is in the 5-007-neighborhood of the nucleus u, then v is assigned to the same cluster as u. This cycle of increasing clusters continues until no cluster can be further created. The process is similar to DBSCAN, a density-based clustering tool.

Formally, the vertex v can be reached directly from the nucleus u if it is v-polyn(u). Transitionally, the vertex v can be reached from the nucleus u if there are vertices w1, ..., if w1 can be reached from u, wi can be reached from wi−1 to 1<i≤n, and v can be reached from wn. In addition, two vertexes, u, v, V, which may or may not be cores, are said to be connected if there is a core w such that both u and v can be reached from w. All vertices in a cluster are linked together. The cluster is a maximum set of vertices so that each pair in the set is connected.
Any vertexes may not belong to a cluster. Such a vertex u is a node if there are vertices of more than one cluster in the neighborhood of name(u)u. If a vertex doesn't belong to a set, and it's not a node, it's an outlier.

Figure 13 displays the SCAN algorithm. The search mechanism is very similar to the cluster-finding method in DBSCAN. SCAN finds a graph cut where each cluster is a group of vertices that are related in a structural sense centered on a transitive similarity.

```
Algorithm: SCAN for clusters on graph data.
Input: a graph G = (V, E), a similarity threshold ε, and a
  population threshold μ
Output: a set of clusters
Method: set all vertices in V unlabeled
  for all unlabeled vertex u do
    if u is a core then
      generate a new cluster-id c
      insert all v ∈ N_ε(u) into a queue Q
      while Q ≠ do
        w ← the first vertex in Q
        R ← the set of vertices that can be directly reached from w
        for all s ∈ R do
          if s is not unlabeled or labeled as nonmember then
            assign the current cluster-id c to s
          endif
          if s is unlabeled then
            insert s into queue Q
          endif
        endfor
        remove w from Q
      end while
    else
      label u as nonmember
    endif
  endfor
  for all vertex u labeled nonmember do
    if ∃x, y ∈ Γ(u) : x and y have different cluster-ids then
      label u as hub
    else
      label u as outlier
    endif
  endfor
```

Figure 13: SCAN algorithm for cluster analysis on graph data.

The advantage of SCAN is that its time complexity is linear with the number of edges. In very large and sparse graphs, the number of edges is the same size as the number of vertexes. SCAN is therefore supposed to have a good scalability for clustering large graphs.

3. DIRECT DISCOVERY OF COMMUNITIES IN GRAPHS

Social network analysis is a prevalent field of research that attracts the attention of many data mining experts. The study of the social network is a basic field in sociology and anthropology. It shares a number of characteristics that are similar to a real network.

Some real networks like Facebook and Twitter pose the idea of group structure within the network. The social network is described as a network graphic. Detection of groups includes the identification of densely connected nodes. Overlapping populations is possible if a node is a member of more than one community.

Structural properties of Real Networks
Graphs representing real systems are neither normal (like lattices) nor random.

• The distribution of the number of links per node of several real networks is different from what is anticipated in random networks

• The degree distribution is large, with the tail always following the power law

• Protein interaction networks: some proteins function as hubs, they are highly linked, while most of the others interact.

• Biological networks: high-level nodes systemically connected to low-level nodes

• Social networks: nodes of identical degrees appear to connect together

• The size of the organization of complex networks indicates a hierarchical structure.

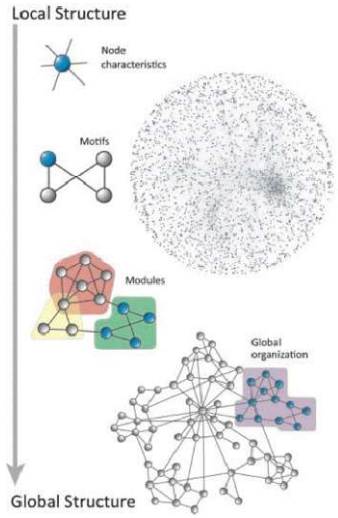


Figure 14: Hierarchical random graph model

Communities and Applications

Community structure:

Vertices in networks are often found tocluster into tightly-knit groups with ahigh density of within-group edges and alower density of between-group edges.

Applications:

• Identify web clients with similar interestand geographically near each other

• Identify customer with similar interests(purchasing history)

• Graph Compression

• Classification of vertices

Relationships in real-world networks

• **Link direction**

- Relationships between nodes may not to be reciprocal
- In the Web few hyperlinks are reciprocal (<10%)
- Community detection on **directed graph** is a hard task

• **Overlapping Communities**: some vertices may belong to more than onegroup

• **Heterogeneous Networks**: different classes of vertices, playing differentroles

- Multipartite Networks

• **Weighted relationships**

Finding Communities

Before telling about finding communities first we should know the definition of community and its properties.

Community: Definition and Properties

Informally, a community C is a subset of nodes of V such that there are**more edges inside** the community than edges linking vertices of Cwith the rest of the graph.

• Intra Cluster Density

$$\delta_{int}(\mathcal{C}) = \frac{\#\text{ internal edges of } \mathcal{C}}{n_c(n_c-1)/2}.$$

• Inter Cluster Density

$$\delta_{ext}(\mathcal{C}) = \frac{\#\text{ inter-cluster edges of } \mathcal{C}}{n_c(n-n_c)}.$$

Where ∂ext(C)<< 2m/ n(n-1)<<∂int(C). Notations are shown in Table 1.

Notations	
V	set of vertices
E	set of edges
n	|V|
m	|E|
C	A subset of V
n_c	|C|

Table 1: Notations

There is not a universally accepted definition of community

• **Connectedness** is a required property: for each pair of vertices in Cthere must exist a path

• Community detection makes sense only on **sparse graphs.**

Local Definitions

• **Clique**: subset of V such that all the vertices are adjacent to eachother

• Triangles are really frequent in real networks

• Finding cliques in a graph is NP Complete

• Too strict definition

• **k-clique**: is a maximal subgraph in which the largest geodesicdistance between any two nodes is no greater than k shown in Figure 15.

• **k-club**: restricts the geodesic distance within the group to be nogreater than k shown in Figure 16.

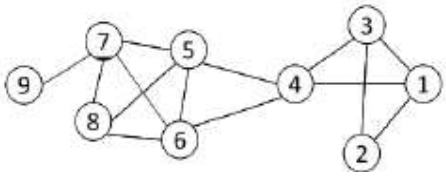


Figure 15: k-clique

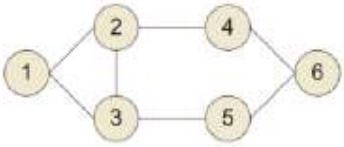


Figure 16: k-club

Global Definitions

• Communities can be also defined with respect to the whole graph
• A graph has a community structure if it is different from a randomgraph
• A random graph is not expected to have any community structure:
    • any two vertices have the same probability to be adjacent

• We can define a **null model** and use it to investigate whether thegraph under consideration exhibit a community structure

Similarity

• A community can be defined as a subset of vertices that are **similar** to each other
  • Do not consider connection

• **Structural equivalence**: v1 and v2 are structural equivalent if theyshare the same neighbors and they are not adjacent

$$d_{ij} = \sqrt{\sum_{k \neq i,j} (A_{ik} - A_{jk})^2},$$

• **Overlap** between the neighborhoods Γ(i) and Γ(j) of vertices i and j

$$\omega_{ij} = \frac{|\Gamma(i) \cap \Gamma(j)|}{|\Gamma(i) \cup \Gamma(j)|}.$$

• **Pearson**

$$C_{ij} = \frac{\sum_k (A_{ik} - \mu_i)(A_{jk} - \mu_j)}{n\sigma_i\sigma_j}, \quad \mu_i = (\textstyle\sum_j A_{ij})/n \quad \sigma_i = \sqrt{\textstyle\sum_j (A_{ij} - \mu_i)^2/n}.$$

• **Commute-time**: average number of steps needed for a randomwalker, starting at either vertex, to reach the other vertex for the firsttime and to come back to the starting vertex

4. PARTITIONING OF GRAPHS

Graph is an abstract concept of representing any entity that is related to each other in the form of a relationship. Here, the object is called a node, and its relationship is defined as an edge. Graph partitioning is a technique used to transfer entire graph data as a disjoint subset to another unit. The need to spread a broad graph data set is to process data efficiently and rapidly for any graph-related applications. A good graph partitioning algorithm is often designed to minimize the contact between machines in their distributed environment and to distribute vertices approximately equal to all machines. Graph partitioning has been studied in the field of information science and data science.

Distribution of massive, unparalleled data is often useful in order to process any data-intensive task. Graphic data partitioning is not the same as they are. Nevertheless, one inherent difficulty in the distribution of graph data, unlike the conventional distribution of data, is that graph data is related to each other. As a consequence, if two connected nodes are spread to another network, the relationship must be preserved.

Graphic partitioning is important and applicable to the following real-world applications:
1) Complex networks: major complex networks include: biological networks, social networks, transport networks.

Social networks: social networks are very important and are commonly used in this field.

All of these implementations are focused on the maintenance and use of graph theory. Graph partitioning technology is used to process user query efficiently, as answering a question in a distributed manner is very convenient and successful. Biological networks: the majority of the biological network can be defined by graph abstraction. They are protein-protein interactions, metabolic networks, and gene co-expression networks. This kind of network is formed through the use of biological entities (e.g. proteins and genes) and their interactions with each other. In this framework, biological entities are vertices and their interactions with each other are boundary. This graph-structured network plays a crucial role in solving the question of biological interaction in a large biological network.

Transportation Networks: Travel and route planning is very popular from the transport network using a GPS (Global Positioning System) tool in the digital age. Graph partitioning can speed up and can be efficient in planning a route by using a strong graph partitioning algorithm. The node in this network is an intersection, and the edge is a path between two intersections. A variety of algorithms have been proposed to minimize search space, minimize processing time and also reduce accuracy.

2) PageRank: PageRank is a site rating program from the site network. PageRank denotes the rank or the value of the web page to others. Finding the rank and significance of a web page from a web-graph would be efficient by splitting the entire graph into many distributed machines.

3) VLSI Design: very large-scale integration (VLSI) system is one of the problems of graph partitioning in order to reduce the relation between circuits when designing VLSI. The main purpose of this partitioning is to reduce the complexity of the VLSI design by splitting it into a smaller portion. Another goal of good partitioning is to reduce the number of connections between these circuit components. Here, the vertices are the cells, and the edges are the wires connecting them.

4) Image processing: Image segmentation is a crucial problem for the image processing of any application. Graph partitioning is one of the most attractive tools for splitting into several sections of an image. Pixels are denoted as a vertex, and if there are similarities between pixels, they are represented as an edge.

5. NEIGHBORHOOD PROPERTIES IN GRAPHS

The neighborhood graph N(G) of the graph G = (V, E) is a graph with a vertex set of $V \cup S$ where S is a set of all open neighborhood sets of G and with two vertices of u, $v \in V \cup S$ adjacent if $u \in V$ and v is an open neighborhood set containing u. Some properties of this new graph are obtained here. Characterization for graphs G is given in such a way that N(G) = G.

Characterizations are also provided for graphs:

(i)whose neighborhood graphs are connected

(ii) whose neighborhood graphs are r-regular

(iii) whose neighborhood graphs are eulerian.

6. DATA VISUALIZATION, BASIC PRINCIPLES

6.1 Data Visualization

Information visualization is the presentation of information in a pictorial or graphic format. It allows decision-makers to see visually oriented analytics so that they can understand complex concepts or recognise new trends. For interactive visualization, you can take the idea a step further by using the application to dig down into charts and graphs for more detail, interactively manipulating the data.

6.2 History of Data Visualization

- The practice of using pictures to interpret data has been around for decades, from maps and graphs of the 17th century to the development of the pie chart in the early 1800s. Many decades later, one of the most cited examples of historical analysis occurred when Charles Minard mapped out Napoleon's invasion of Russia. The map represented the size of the army as well as the direction of Napoleon's withdrawal from Moscow – and related the detail to the temperature and time scales for a deeper understanding of the battle.
- It's technology, however, that really sparked the fire as a result of data visualization. Computers have made it possible to process vast volumes of data at flash-fast speeds. Today, data visualization has become a rapidly emerging combination of science and art that is expected to transform the business environment in the next few years.

6.3 Why is data visualization important?

The way human brain processes information, using charts or graphs to represent large volumes of complex data, is easier than porting over spreadsheets or reports. Data visualization is a simple, easy way to express concepts in a standardized way – and you can play with various scenarios by making minor changes.

Data visualization can also:

- Identify areas that need attention or improvement.
- Clarify which factors influence customer behavior.
- Help you understand which products to place where.
- Predict sales volumes.

6.4 What makes data visualization effective?

Visualizing data is successful when done correctly. We define correctly when the visualization of the data has served its purpose. Fast test-when people can understand the diagram by asking more questions about the details shown than how or what is seen, then you know you're on the right road. So, in order to be highly successful, it is important to create the right visualizations for your data so that you and your team members can perceive and make decisions based on what they see. How are we going to do that? It's really easy. We construct the right visualizations by recognizing the various types of visualizations and answering 5 questions.

Why visualization?

Without the concept of visualization, mining and analysis have no important role to play, as data mining is the idea of making inferences by analyzing data through patterns, and these patterns can only be interpreted by different visualization techniques.

Usage of data visualization.

• Effective way of testing data with presentable results.

• The primary use is the pre-processing part of the data mining process.

• Helps the data cleaning process by identifying incorrect and missing values.

• For variable derivation and selection means to decide which variable to be included and excluded in the study.

• It also plays a role in integrating categories as part of the data reduction process

6.5 Five types of Big Data Visualization groups

(1) Temporary.

Data visualizations fall into the temporal category if they satisfy two conditions: that they are linear and that they are one-dimensional. Temporal visualizations typically show lines that either stand alone or overlap with each other, with time to start and finish.

The plus, huh? They are common maps that can be used from school and the workplace, which means that we have better comprehension when reading them.

Examples of time data visualization include:

- Scatter plots
- Polar area diagrams
- Time series sequences
- Timelines
- Line graphs

(2) Hierarchical

The data visualizations that belong to the hierarchical category are those that organize groups within larger groups. Hierarchical visualizations are best suited if you want to view clusters of information, particularly if they flow from a single point of origin.

The downside to these diagrams is that they appear to be more complicated and difficult to understand, which is why the tree diagram is more widely used. It's the easiest to follow because of its linear direction.

Examples of hierarchical data visualization include:

- Tree diagrams
- Ring charts
- Sunburst diagrams

(3) Network

Datasets are closely linked to other datasets. Network data visualizations demonstrate how they connect to each other within a network. In other words, to explain the relationship between datasets without wordy explanations.

Examples of network data visualization include:

- Matrix charts
- Node-link diagrams
- Word clouds
- Alluvial diagrams

(4) Multidimensional

Like the word, multidimensional data visualizations have multiple dimensions. This means that there are often two or more variables in the mix to create a 3D data visualization. Such types of visualizations appear to be the most vivid or eye-catching graphics due to the multiple overlapping layers and datasets. Another bonus, huh? Such graphics will break down a lot of data to main takeaways.

Examples of multidimensional data visualization include:

- Scatter plots
- Pie charts
- Venn diagrams
- Stacked bar graphs
- Histograms

(5) Geospatial

Geospatial or geographic data visualizations refer to real-life physical locations, overlaying common maps of specific data points. Such types of data visualizations are widely used to display revenues or acquisitions over time, and may be most popular for use in political campaigns or to show market share of multinational corporations.

Types of geospatial data visualization include:

- Flow map
- Density map
- Cartogram
- Heat map

7. COMMON DATA VISUALIZATION TOOLS

Typically speaking, R, ggplot2 and Python are used in academia. Excel is the most common resource for ordinary users. Professional goods include Tableau, FineReport, Power BI, etc. (1) D3.

D3.js is a JavaScript library focused on data processing documentation. D3 integrates efficient visualization elements with data-driven DOM handling methods.

Evaluation: D3 has a good SVG capability. It can easily map data to the SVG attribute and incorporates a wide range of tools and methods for data analysis, layout algorithms and graphics calculation. It's got a good group and rich demos. The API is too weak, however. There's not a lot of reusability at the expense of learning.

2) HighCharts

HighCharts is a graph library written in pure JavaScript that makes it simple and convenient for users to add interactive graphs to web applications. It is the most commonly used web-based chart method, and business usage includes the purchase of a commercial license.

Evaluation: the threshold for usage is very small. HighCharts has excellent usability and is mature and commonly used. But the style is old, and it's hard to broaden the charts. Commercial use often includes the purchase of copyright.

3) Echarts

Echarts is an enterprise-level map tool from the Baidu Data Visualization Team. It's a pure JavaScript map library that runs smoothly on PCs and mobile devices, and is compatible with most current browsers.

Evaluation: Charts are rich in chart forms, including standard statistical charts. But it's not as robust as Vega and other graphical graph libraries, so it's hard for users to customize those complex relational graphs.

4) Leaflet is a JavaScript library of interactive maps for mobile devices. It has all the mapping features most developers need.

Evaluation: It can be explicitly targeted for map applications and has excellent device usability. The API supports a plug-in function, but the functionality is fairly simple. Users need secondary technology capabilities.

5) Vega

Vega is a set of interactive graphical grammars that describe mapping rules from data to graphics, common grammatical interactions, and common graphical elements. Users can freely combine Vega grammar to create a variety of maps.

Evaluation: Based entirely on JSON grammar, Vega provides mapping rules from data to graphics and supports common interaction grammar. But the nature of grammar is complicated, and the cost of use and learning is high.

6) deck.gl

Deck.gl is a WebGL-based visual class library for big data analytics. It is developed by the Uber Visualization Team.

Evaluation: deck.gl focuses on the visualization of 3D maps. There are several growing scenes of integrated geographic information visualization. Supports the visualization of large-scale data. But users need to learn about WebGL, and layer extension is more difficult.

7) Power BI

Power BI is a suite of market research tools that provide insights into the company. It can link hundreds of data sources, simplify the preparation of data and provide instant analysis. Organizations can display Power BI-generated reports on web and mobile devices.

Evaluation: Power BI is similar to Excel's desktop BI tool, though it is more powerful than Excel. Supports multiple data sources. It's not a high price. But it can only be used as a separate BI device, and there is no way to incorporate it into existing systems.

8) Tableau

Tableau is a business intelligence platform for visual analysis of results. Users can build and distribute interactive and collaborative dashboards, representing patterns, changes and data densities in graphs and charts. Tableau can connect to databases, relational data sources and large data sources to capture and process data.

Evaluation: Tableau is the simplest business intelligence tool on the mobile framework. It's not asking users to write custom code. The platform allows data mixing and real-time collaboration. But it's costly and does less well in configuration and after-sales facilities.

9) FineReport

FineReport is an enterprise-level online reporting platform written in pure Java that incorporates data visualization and data entry. It is based on the principle of "no-code creation." FineReport helps users to make detailed reports and fun dashboards and create a decision-making framework with easy drag-and-drop operations.

Evaluation: FineReport can be directly linked to all sorts of databases, and it is easy and simple to customize a number of complex reports and interesting dashboards. The interfaceis identical to the one in Excel. It includes 19 categories and over 50 types of self-developed HTML5 maps, with cool 3D and dynamic effects. The most significant thing is that his personal edition is completely free of charge.

8. EXAMPLES OF INSPIRING (INDUSTRY) PROJECTS

Best Data Science Projects

Below are the top Data Science project ideas to master the technology:

1. Movie Recommendation System Project
2. Customer Segmentation using Machine Learning
3. Sentiment Analysis Model in R
4. Uber Data Analysis Project
5. Credit Card Fraud Detection Project in R

1. Movie Recommendation System Project

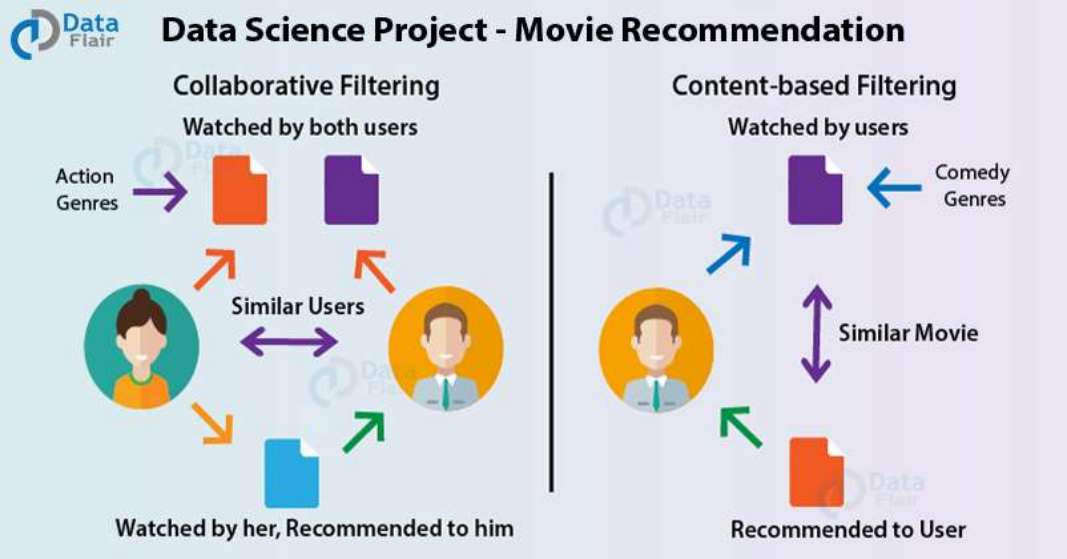


Figure 17: Layout of Movie Recommendation

The purpose of this fascinating Data Science project, including its code, is to create a recommendation system that recommends movies to users shown in Figure 17.

Let's use an example to grasp this. Have you ever been to an online entertainment site like Netflix or Amazon Prime? If yes, then you must have found that after some time, this website starts recommending different movies and TV shows according to your preferences of genre. This R programming project is intended to help you understand how the recommendation system works.

2. Customer Segmentation using Machine Learning

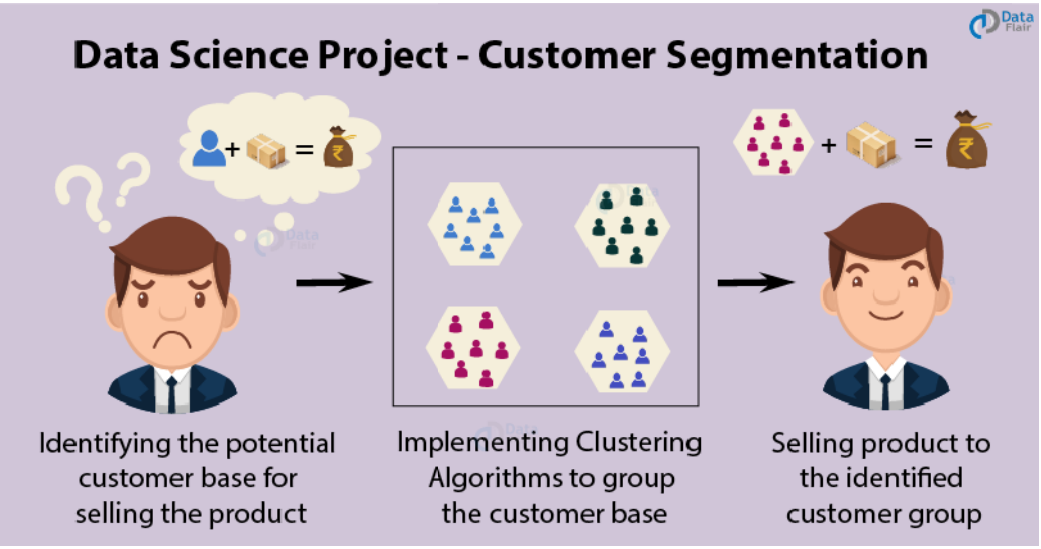


Figure 18: Layout of Customer Segmentation using Machine Learning

Consumer segmentation is one of the most important technologies for all consumer-oriented businesses (B2C companies). It uses the Machine Learning clustering algorithm that enables businesses to target the potential user base and also to recognize the best customers.

This uses clustering strategies in which businesses can classify a variety of groups of consumers, allowing them to target the potential consumer base for a particular campaign. Customer segmentation also uses the K-means clustering algorithm, which is important for clustering unmarked datasets. The Layout of Customer Segmentation is shown in Figure 18.

3. Sentiment Analysis Model in R

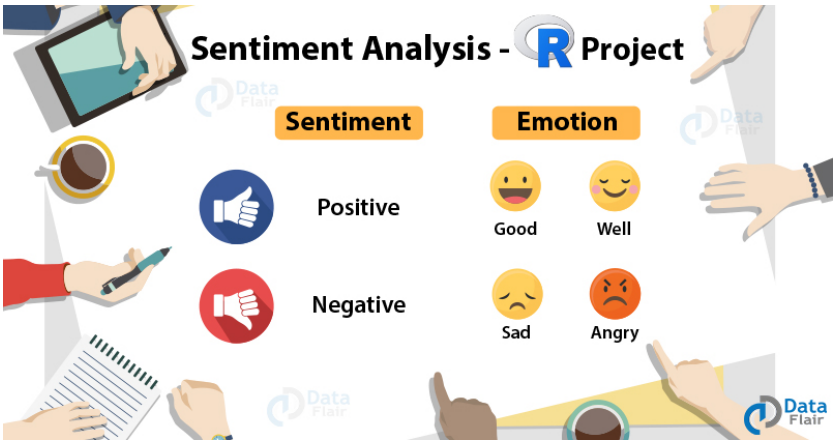


Figure 19: Layout of Sentiment Analysis Model in R

Almost every data-driven organization uses a sentiment analysis model to assess the attitude of its consumers towards the company's goods. The layout of Sentiment Analysis Model is shown in Figure 19.

In short, it is a method of computational recognition and categorization of opinions expressed in the text, in particular with a view to deciding whether the consumer's attitude towards a particular product or subject is positive, negative or neutral. You may need to use a small text program to interpret the data and assign scores to the corresponding terms that are already present in the dataset.

4. Uber Data Analysis Project

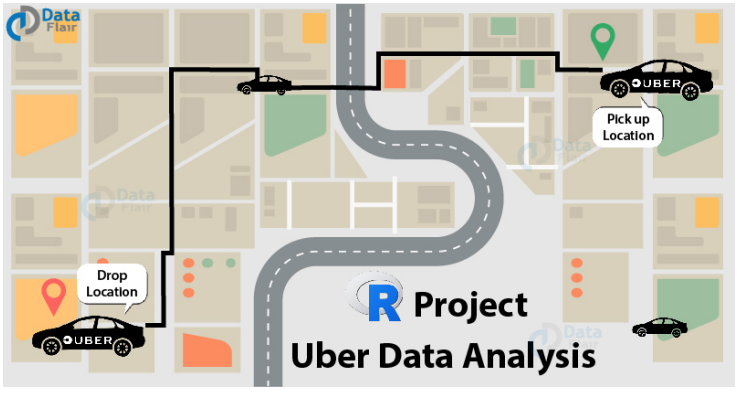


Figure 20: Layout of Uber Data Analysis Project

The data is Uber oil. Uber strengthens its decision-making, marketing policy, promotional deals and predictive analytics with data mining tools and insights. This layout is shown in Figure 20.

With more than 15 million rides a day across 600 cities in 65 countries, Uber is rising increasingly with Data Science starting from data analysis and gaining insights that help them make better decisions. Data Science tools play a key role in any Uber project.

5. Credit Card Fraud Detection Project in R

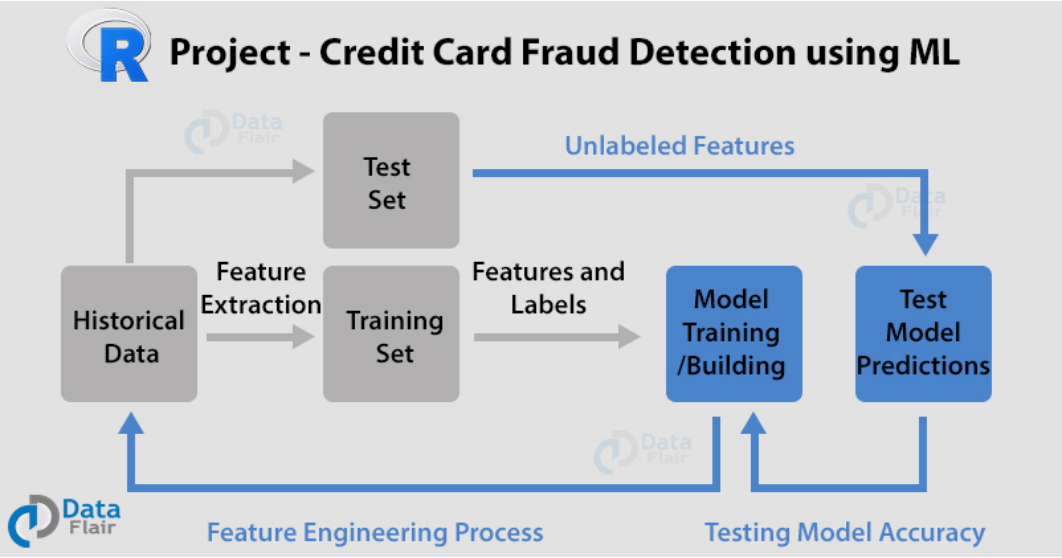


Figure 21: Layout of Credit Card Fraud Detection Project in R

The credit card fraud detection project uses machine learning and programming principles for R shown in Figure 21.

The goal of this project is to develop a classifier that can detect fraudulent credit card transactions using a variety of machine learning algorithms that will be able to distinguish fraudulent transactions from non-fraudulent ones. Learn how to apply machine learning algorithms and data analysis and visualization to identify fraudulent transactions from other data forms from the Credit Card Fraud Detection Data Science Project.

9. DATA SCIENCE AND ETHICAL ISSUES

What is ethics in data science?

Computer Science Ethics * ... If ethics is characterized as common principles that allow us to distinguish right from wrong, the digitization of human behavior affects the very meaning of how we judge the world around us.

Ethics of Data Science If ethics is characterized as common principles that allow us to discern right from wrong, the digitization of human behavior affects the very meaning of how we judge the world around us.

Margo Boenig-Liptsin points out that our ever-increasing reliance on information technology has radically changed the conventional conceptions of "privacy," "fairness" and "representation," not to mention "free choice," "right" and "reality".

These mutations underscore the growing footprint and obligations of data science — beyond bytes and pieces, data science is shaking the perceptual foundations of meaning, culture, and equity. If the Academy has been swift to set up data science programs on statistics, computing and software engineering, few programs discuss the wider social issues of data science, and fewer still explore how ethical data practices can be shaped and even encouraged. Let us outline the contours of this challenge.

1. Data Citizens: Maybe no field of data science ethics has gained more attention today than the security of personal data. The digital transformation of our relationships with the social and economic systems shows who we are, what we think, and what we do.

2. Automated decision-making: the ability to actively take decisions on alternate choices has long been seen as a characteristic that distinguishes man (or at least the living) from the system.

3. Micro-targeting: The use of micro-targeting has since been developed as important instruments of leverage in the fields of marketing, politics and economics. Even if the individual's right to exercise "free choice" has long been the topic of controversy, the method of feeding customers simply offers evidence that they can choose to further attach rationa*lity.*

In addition, micro-targeting techniques allow researchers to extrapolate sensitive information and personal preferences of individuals even when such data are not directly collected. Finally, when the "company is a commodity," there is a real danger that data science will be used less to enhance the organization's commodity or service offering than to turn users into objects of exploitation.

4. Distributed ledgers: distributed ledger systems in general, and blockchain systems in particular, offer their share of hope for a more open and traceable source of information. Yet this view of the Internet of Value is partly clouded by the future social problems of relatively untested technologies. Can technology, in and of itself, be the benchmark of both truth and trust? To what degree do individuals and organisations recognize the primacy of transparency? In what basis will universal principles, such as the right to be forgotten, be reconciled with the legal conditions of the public authorities? Freed from the rules and logic of financial capitalism, can human nature embrace a fundamentally different basis for the distribution of wealth?

5. Human and machine intelligence: While the effect of information technology on the organization of private and public organizations has been extensively discussed over the last four decades, much less attention has been given to the effects of data science on management functions. Within the commercial press, technology is mostly seen as ethically neutral and at any point in time appears to be a digital mirror of existing managerial paradigms. In research, the relationship is subject to greater examination, scholars such as Latour, Callon and Law have shown how various types of technology affect the way in which managers, workers and consumers view the nature of social and economic transactions, the economy, etc.

Contact Us:

University Campus Address:

Jayoti Vidyapeeth Women's University

Vadaant Gyan Valley, Village-Jharna, Mahala Jobner Link Road,

Jaipur Ajmer Express Way, NH-8, Jaipur- 303122, Rajasthan (INDIA)

(Only Speed Post is Received at University Campus Address, No. any Courier Facility is available at Campus Address)

Pages	: 131
Book Price	: ₹ 150/-

JPD Women University Press

Jayoti Publication Desk