SCHOOL OF COMPUTER SCIENCE,ENGINEERING AND APPLICATIONS ,

BHARATHIDASAN UNIVERSITY,

KHAJAMALAI CAMPUS,

TRICHY-620 023

PRINCIPLES OF DATA SCIENCE

(SUB. CODE: MDS24011)

STUDY MATERIAL

FACULTY NAME : Mrs. R. RAMYA

DESIGNATION : GUEST LECTURER

SEMESTER : I

CLASS : M.Sc., **(DATA SCIENCE)**

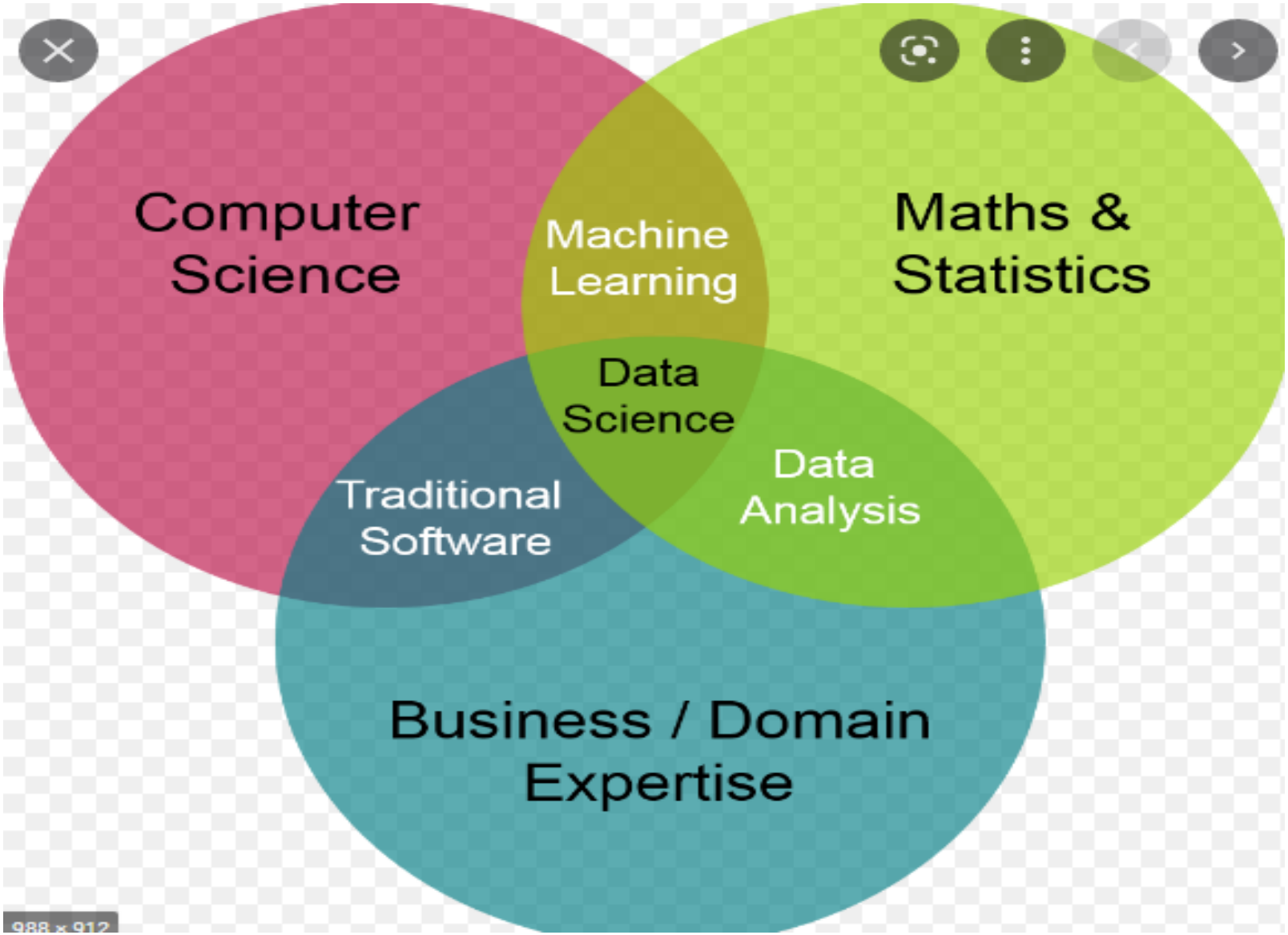
Principles of Data Science

Unit – I: Introduction

- Introduction to Data Science
- Evolution of Data Science
- Data Science Roles
- Stages in a Data Science Project
- Applications of Data Science in various fields
- Data Security Issues.

Introduction to Data Science

- **Data Science** is the area of study which involves extracting insights from vast amounts of data using various scientific methods, algorithms, and processes.
- It helps you to discover hidden patterns from the raw data.
- The term Data Science has emerged because of the evolution of mathematical statistics, data analysis, and big data.
- Data Science is an interdisciplinary field that allows you to extract knowledge from structured or unstructured data.
- Data science enables you to translate a business problem into a research project and then translate it back into a practical solution.


Computer Science
Machine Learning
Maths & Statistics
Data Science
Traditional Software
Data Analysis
Business / Domain Expertise
988 x 912

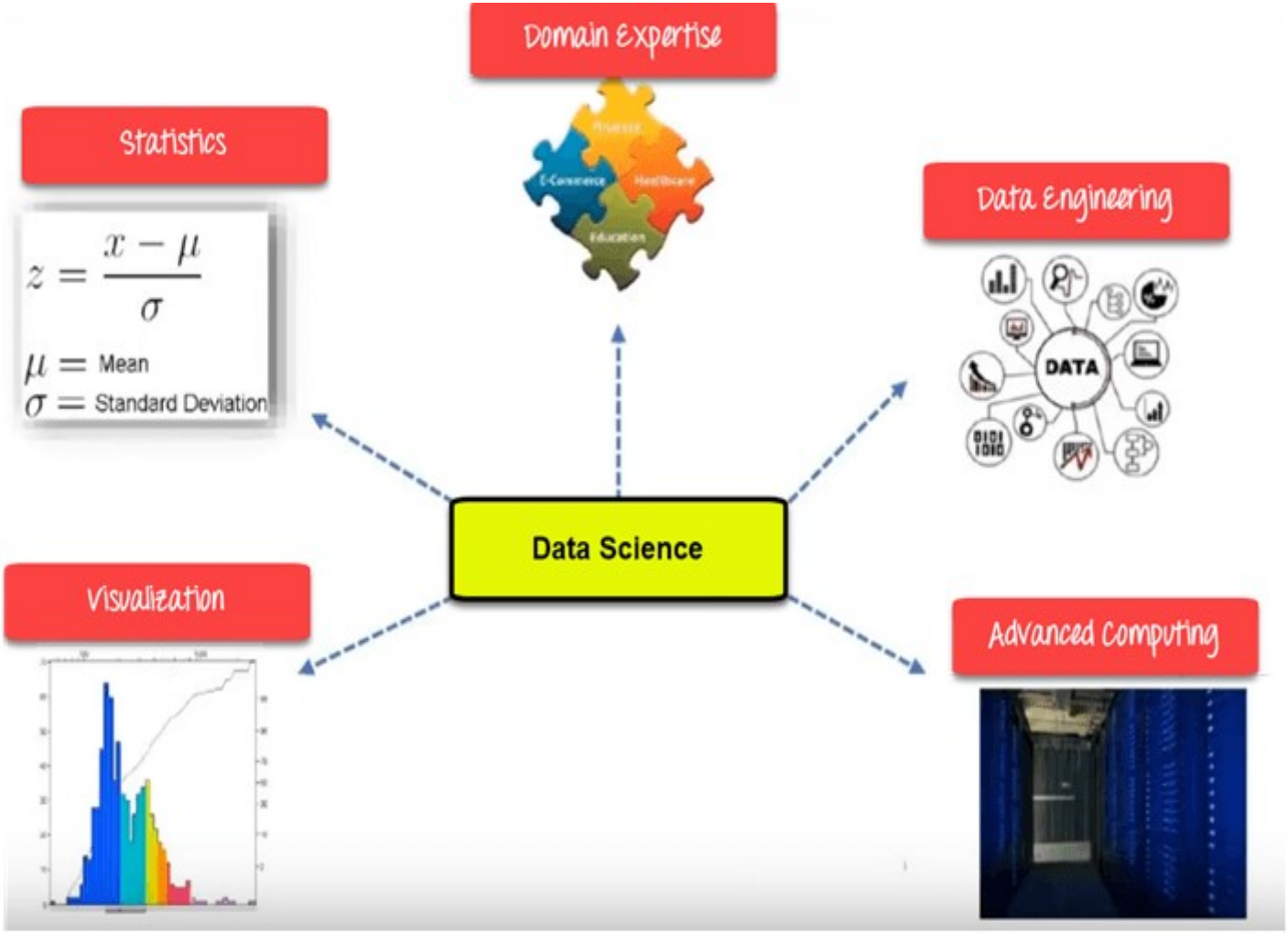
Why Data Science?

Here are significant advantages of using Data Analytics Technology:

- Data is the oil for today's world. With the right tools, technologies, algorithms, we can use data and convert it into a distinct business advantage
- Data Science can help you to detect fraud using advanced machine learning algorithms
- It helps you to prevent any significant monetary losses
- Allows to build intelligence ability in machines
- You can perform sentiment analysis to gauge customer brand loyalty
- It enables you to take better and faster decisions
- It helps you to recommend the right product to the right customer to enhance your business

Evolution of DataSciences

Data Science Components



Statistics:

- Statistics is the most critical unit of Data Science basics, and it is the method or science of collecting and analyzing numerical data in large quantities to get useful insights.

Visualization:

- Visualization technique helps you access huge amounts of data in easy to understand and digestible visuals.

Machine Learning:

- Machine Learning explores the building and study of algorithms that learn to make predictions about unforeseen/future data.

Deep Learning:

- Deep Learning method is new machine learning research where the algorithm selects the analysis model to follow.

DATA SCIENCE JOBS ROLES

Most prominent Data Scientist job titles are:

- Data Scientist
- Data Engineer
- Data Analyst
- Statistician
- Data Architect
- Data Admin
- Business Analyst
- Data/Analytics Manager

Data Scientist:

- **Role:** A Data Scientist is a professional who manages enormous amounts of data to come up with compelling business visions by using various tools, techniques, methodologies, algorithms, etc.
- **Languages**: R, SAS, Python, SQL, Hive, Matlab, Pig, Spark

Data Engineer:

- **Role**: The role of a data engineer is of working with large amounts of data. He develops, constructs, tests, and maintains architectures like large scale processing systems and databases.
- **Languages**: SQL, Hive, R, SAS, Matlab, Python, Java, Ruby, C + +, and Perl

Data Analyst:

- **Role**: A data analyst is responsible for mining vast amounts of data. They will look for relationships, patterns, trends in data. Later he or she will deliver compelling reporting and visualization for analyzing the data to take the most viable business decisions.
- **Languages**: R, Python, HTML, JS, C, C+ + , SQL

Statistician:

- **Role**: The statistician collects, analyses, and understands qualitative and quantitative data using statistical theories and methods.
- **Languages**: SQL, R, Matlab, Tableau, Python, Perl, Spark, and Hive

Data Administrator:

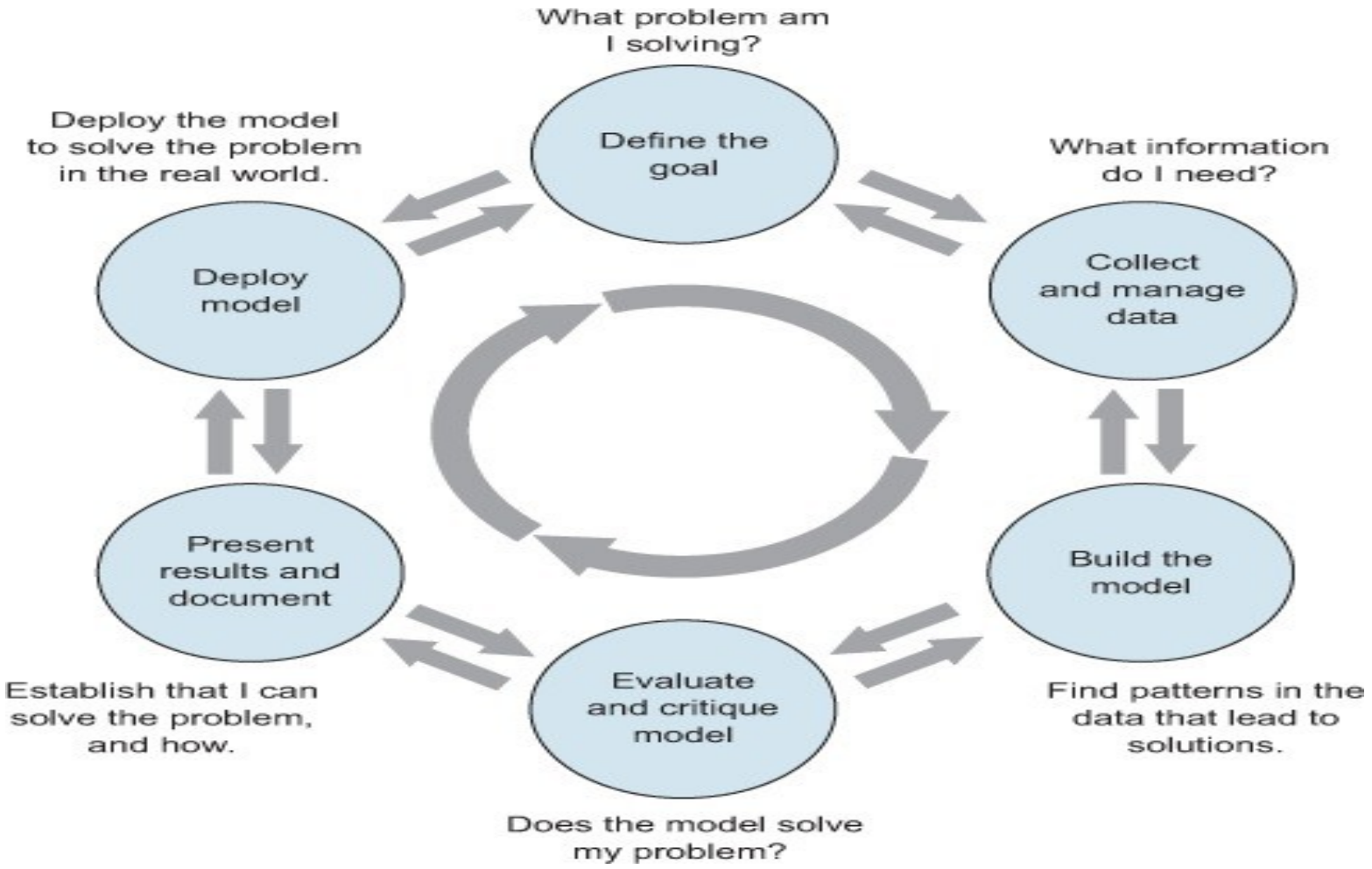
- **Role**: Data admin should ensure that the database is accessible to all relevant users. He also ensures that it is performing correctly and keeps it safe from hacking.
- **Languages**: Ruby on Rails, SQL, Java, C#, and Python

Business Analyst:

- **Role**: This professional needs to improve business processes. He/she is an intermediary between the business executive team and the IT department.
- **Languages**: SQL, Tableau, Power BI and, Python

STAGES OF A DATA SCIENCE PROJECT

- The ideal data science environment is one that encourages feedback and iteration between the data scientist and all other stakeholders. This is reflected in the lifecycle of a data science project.



Defining the goal

- The first task in a data science project is to define a measurable and quantifiable goal. At this stage, learn all that you can about the context of your project:
    1. Why do the sponsors want the project in the first place? What do they lack, and what do they need?
    2. What are they doing to solve the problem now, and why isn't that good enough?
    3. What resources will you need: what kind of data and how much staff? Will you have domain experts to collaborate with, and what are the computational resources?
    4. How do the project sponsors plan to deploy your results? What are the constraints that have to be met for successful deployment?

Data collection and management

- This step encompasses identifying the data you need, exploring it, and conditioning it to be suitable for analysis. This stage is often the most time-consuming step in the process. It's also one of the most important:

    1. What data is available to me?
    2. Will it help me solve the problem?
    3. Is it enough?
    4. Is the data quality good enough?

Status.of.existing.checking.account *(at time of application)*

Duration.in.month *(loan length)*

Credit.history

Purpose *(car loan, student loan, etc.)*

Credit.amount *(loan amount)*

Savings.Account.or.bonds *(balance/amount)*

Present.employment.since

Installment.rate.in.percentage.of.disposable.income

Personal.status.and.sex

Cosigners

Present.residence.since

Collateral *(car, property, etc.)*

Age.in.years

Other.installment.plans *(other loans/lines of credit—the type)*

Housing *(own, rent, etc.)*

Number.of.existing.credits.at.this.bank

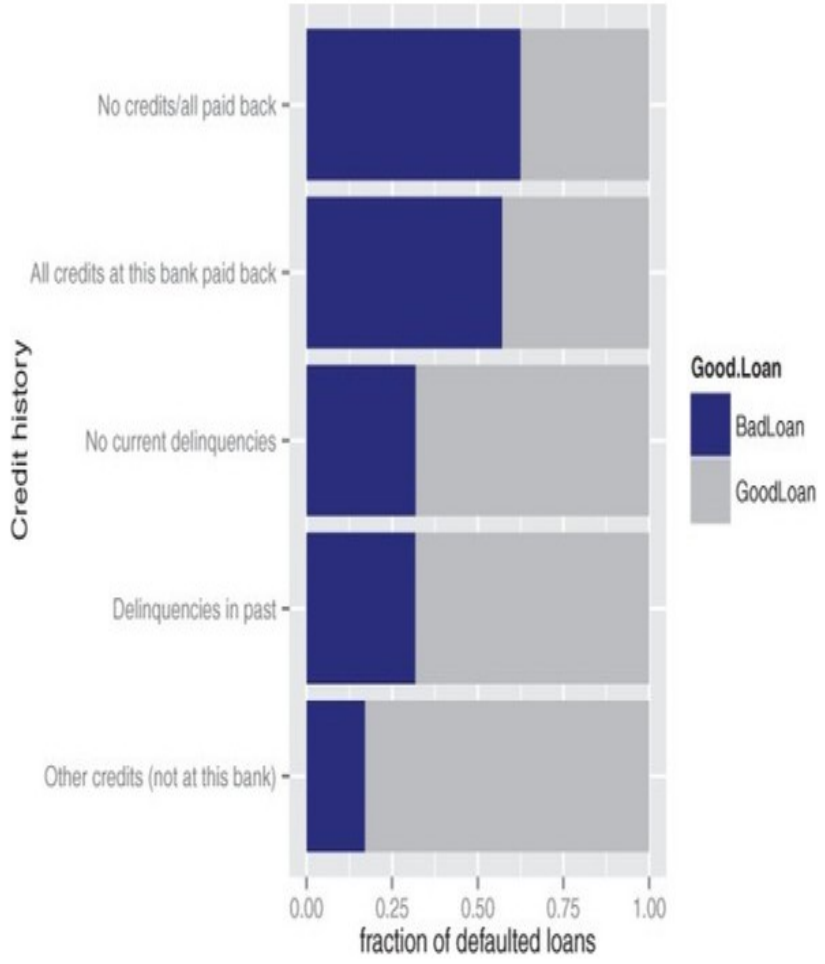
Job *(employment type)*

Number.of.dependents

Telephone *(do they have one)*

Good.Loan *(dependent variable)*

Figure 1.2. The fraction of defaulting loans by credit history category. The dark region of each bar represents the fraction of loans in that category that defaulted.



Modeling

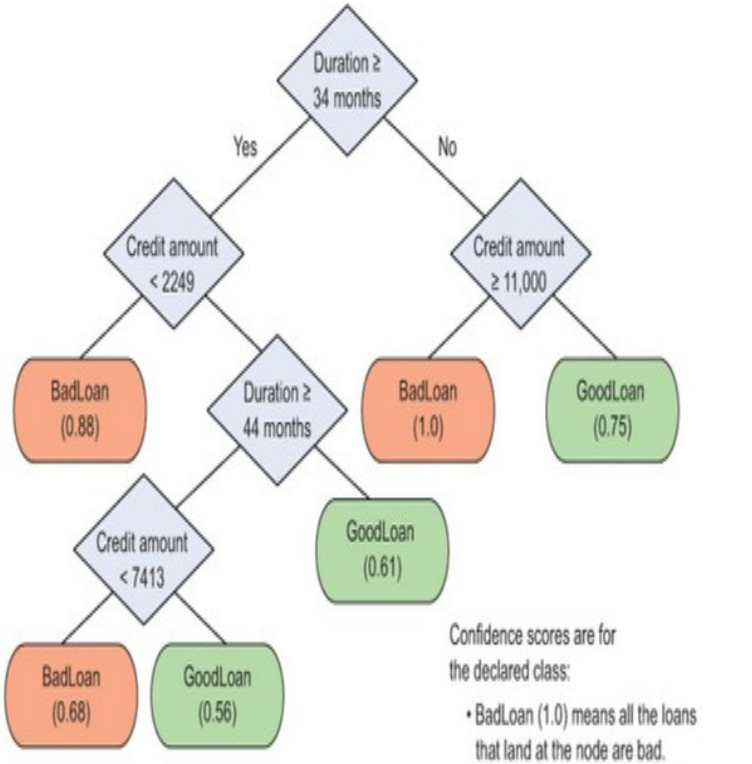
- Finally get to statistics and machine learning during the modeling, or analysis, stage. Here is where you try to extract useful insights from the data in order to achieve your goals.
- The most common data science modeling tasks are these:

1. ***Classification—*** *Deciding* if something belongs to one category or another
2. ***Scoring—*** *Predicting* or *estimating* a numeric value, such as a price or probability
3. ***Ranking—*** Learning to *order items* by preferences
4. ***Clustering—*** *Grouping items* into most-similar groups
5. ***Finding relations—*** *Finding correlations* or potential causes of effects seen in the data
6. ***Characterization—*** Very general *plotting* and *report generation* from data

Listing 1.1. Building a decision tree

```
1  library('rpart')
2  load('GCDData.RData')
3  model <- rpart(Good.Loan ~
4    Duration.in.month +
5    Installment.rate.in.percentage.of.disposable.income +
6    Credit.amount  +
7    Other.installment.plans,
8    data=d,
9    control=rpart.control(maxdepth=4),
10   method="class")
```

Figure 1.3. A decision tree model for finding bad loan applications, with confidence scores



Model evaluation and critique

- Once you have a model, you need to determine if it meets your goals:
  1. Is it accurate enough for your needs? Does it generalize well?
  2. Does it perform better than "the obvious guess"? Better than whatever estimate you currently use?
  3. Do the results of the model (coefficients, clusters, rules) make sense in the context of the problem domain?

Listing 1.2. Plotting the confusion matrix

```
> resultframe <- data.frame(Good.Loan=creditdata$Good.Loan,
                            pred=predict(model, type="class"))
> rtab <- table(resultframe)
> rtab
          pred
Good.Loan  BadLoan GoodLoan
  BadLoan       41      259
  GoodLoan      13      687
> sum(diag(rtab))/sum(rtab)
[1] 0.728
> sum(rtab[1,1])/sum(rtab[,1])
[1] 0.7592593
> sum(rtab[1,1])/sum(rtab[1,])
[1] 0.1366667
> sum(rtab[2,1])/sum(rtab[2,])
[1] 0.01857143
```

Create the confusion matrix. Rows represent actual loan status; columns represent predicted loan status. The diagonal entries represent correct predictions.

Overall model accuracy: 73% of the predictions were correct.

Model precision: 76% of the applicants predicted as bad really did default.

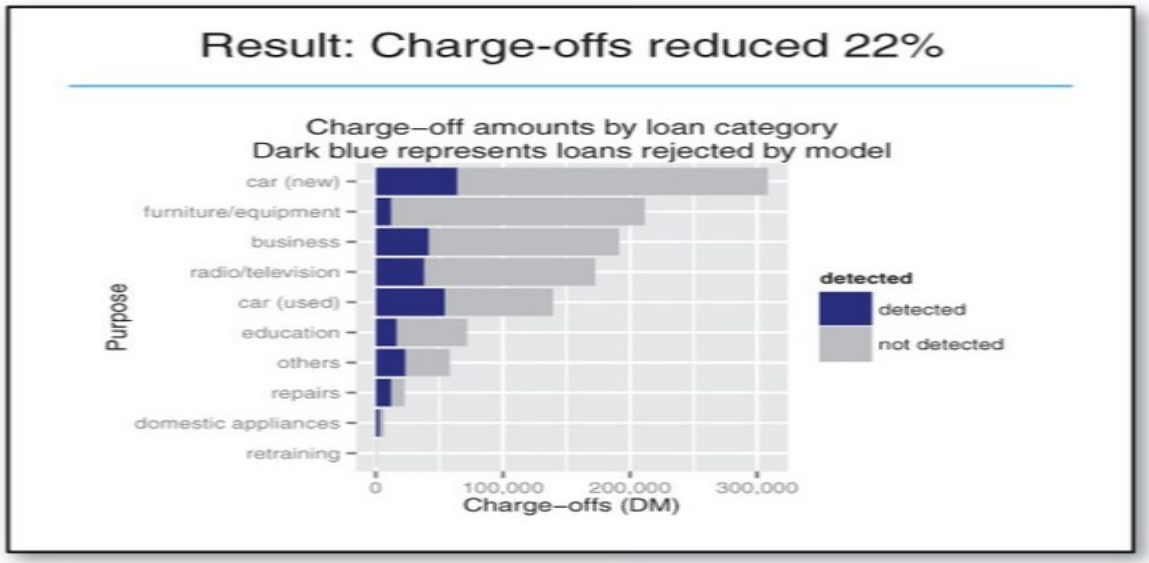
Model recall: the model found 14% of the defaulting loans.

False positive rate: 2% of the good applicants were mistakenly identified as bad.

Presentation and documentation

Once you have a model that meets your success criteria, you'll present your results to your project sponsor and other stakeholders. You must also document the model for those in the organization who are responsible for using, running, and maintaining the model once it has been deployed.

Figure 1.4. Notional slide from an executive presentation



Model deployment and maintenance

- Finally, the model is put into operation. In many organizations this means the data scientist no longer has primary responsibility for the day-to-day operation of the model.

DATA SCIENCE APPLICATION IN DIFFERENT AREAS

- The field of Data Science is filled with wonderful applications. In this modern era of digitization, Data Science is making a huge difference in making businesses successful.
- Not only in business but fields like healthcare, aeronautics, robotics, medicine, etc., Data Science is the game-changer. Here is the list of applications of Data Science that you will be introduced to in this blog:
    - Education
    - Airline Route Planning
    - Healthcare Industry
    - Delivery Logistics
    - Banking and Finance
    - Filtered Internet Search
    - Product Recommendation Systems
    - Digital Marketing and Advertising

Education

Data Science application in education, we can analyze the following:

- Students' need to improve their performance by analyzing the same based on factors such as specific books, study materials, or effective learning strategies
- The requirement of designing or updating the course curriculum as per the analysis of students' IQ levels and performance
- Performance of teachers based on student reviews, results of students, improvements by the weak students, etc.

Airline Route Planning

Data Science has helped airlines in the following ways:

- Identifying potential customers to offer calculated discounts, instead of providing discounts to everyone
- Deciding on the optimized routes by analyzing the traffic on different routes. It helps in saving expensive fuel that gets unnecessarily exhausted otherwise
- Predicting delays in flight
- Setting the cost of flights as per seasons, festivals, and the number of travelers. This is done by analyzing the number of potential travelers and frequent travelers
- This is how the application of Data Science is optimizing the profits of the airline industry.

Healthcare Industry

The areas where the **Data Science application in healthcare** is playing a major role are as follows:

- **Patient Diagnosis:** Data Science helps doctors monitor patients' health with the help of IoT devices. These wearable devices help monitor various medical conditions such as heart rate, body temperature, blood pressure, etc. These devices send patients' data to concerned doctors for medical analysis. This helps doctors take the necessary steps for treating patients accordingly.
- **Drug Research and Creation:** For creating a pharmaceutical drug, it takes a lot of research, time, and money. Also, there are millions of test cases required for research. Using Data Science, we can process all these test cases and make predictions on the success rate of these, based on certain parameters used for evaluating drugs in less time. With this application of Data Science, we can successfully create highly effective medicines.
- **Medical Image Analysis:** This is one of the interesting applications of Data Science, which is rapidly changing the way doctors do patient diagnoses. With the help of medical image analysis, a machine predicts diseases such as cancer, tumor, organ delineation, and many others.
- **Managing Patient Data:** Apart from the other applications of Data Science, it helps in managing patient data. The patient data is stored in databases and can be used in the future for the analysis of several medical conditions and the improvement of medical diagnosis and treatment

Delivery Logistics

- It helps in the analysis of profit generation, the causes of loss, the best route for delivery, the time required, and the scope for improvements.
- Other than that, the application of Data Science in delivery logistics helps the companies analyze the market trend and increase their competence. Further, with the help of route optimization, the number of deliveries increases and the freight cost reduces.

Banking and Finance

As we all know, the sector of banking and finance is prone to financial frauds and thefts. This happens due to the lack of proper analysis of customer data.

the application of Data Science in Finance helps in the following ways:

- **Stamping out Tax Fraud:** The economy of a nation depends on its taxpayers. Therefore, governments have started implementing Data Science to analyze citizen data to prevent tax fraud. With the application of Data Science, the income tax departments keep track of the income and calculate the tax. If the calculated tax is not collected, they track the suspicious taxpayers to take action against them.
- **Credit Scoring:** Credit scoring is another application of Data Science that helps check the financial civil score of an individual. The credit score is rated out of 10. It helps financial institutions make decisions on sanctioning loans. They decide the loan amount and its sanctioning on the basis of the credit score calculated out of 10.

Filtered Internet Search

- This is a type of filtered Internet search and also one of the wonderful applications of Data Science.
- It is only possible with the help of Data Science. Google collects and stores the data of search history to analyze and visualize it. Then, it uses algorithms and techniques that apply filters to the data to check the frequency of the searched keyword and related topics to show you the best results.

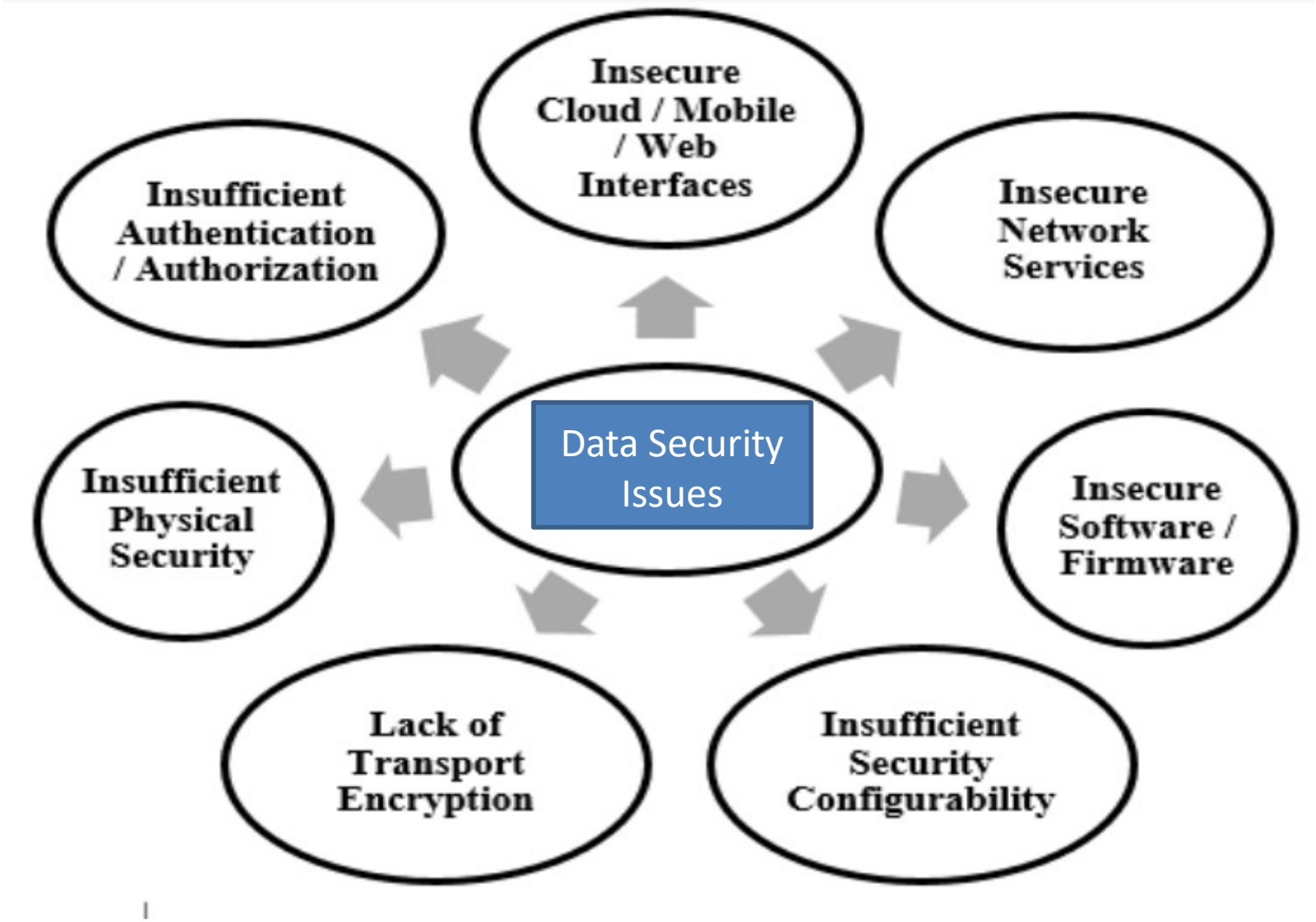
Product Recommendation Systems

- Product recommendation is an effective way of converting leads into sales.
- All the industries based on sales use recommendation systems for improving their profitability. But, how do these recommendation engines work? It is again a Data Science application.

Digital Marketing and Advertising

- The Data Science application in digital marketing helps organizations advertise their products to the right customers.
- Data Scientists design algorithms to analyze and visualize customers' data related to their search history, interests, and previously shopped items.

Data Security Issues:



The Data Security Issues

- Data Storage
- Fake Data
- Data Privacy
- Data Management
- Data Access Control
- Data Poisoning
- Employee Theft

Data Storage

- Businesses are adopting Cloud Data Storage to move their data easily to expedite business operations. However, the risks involved are exponential with security issues.
- While mission-critical information can be stored in on-premise databases, less sensitive data is kept in the cloud for ease of use Although it increases the cost of managing data in on-premise databases, companies must not take security risks for granted by storing every data in the cloud.

Fake Data

- Fake Data generation poses a severe threat to businesses as it consumes time that otherwise could be spent to identify or solve other pressing issues.
- There is more scope for leveraging inaccurate information on a very large scale, as assessing individual data points can be a daunting task for companies.

Data Privacy

- Data Privacy is a big challenge in this digital world.
- It aims to safeguard personal or sensitive information from cyberattacks, breaches, and intentional or unintentional data loss.
- The general rules are knowing your data, having more grip over your data stores and backup, safeguarding your network against unauthorized access, conducting regular risk assessments, and training the users regularly about Data Privacy and Data Security.

Data Management

- A security breach can have crushing consequences on businesses, including the vulnerability of critical business information to a completely compromised database.
- Deploying highly secured databases is vital to ensure data security at all levels.
- A superior Database Management System comes with various access controls.

Data Access Control

- Controlling which data users can view or edit enables companies to ensure not only data integrity but also preserves its privacy. But managing access control is not straightforward, especially in larger companies that have thousands of employees.

Data Poisoning

- Data Poisoning, a technique to attack Machine Learning models' training data.
- It can be considered as an integrity attack as the tampered training data can affect the model's ability to provide correct predictions.
- The results can be catastrophic, ranging from logic corruption to Data Manipulation and Data Injection.

Employee Theft

- The risk of an employee leaking sensitive information, intentionally or unintentionally, is high.
- Employee Theft is prevalent not only in big tech companies but also in startups.
- To avoid Employee Theft, companies have to implement legal policies along with securing the network with a virtual private network. In addition, companies can use a Desktop as a Service (DaaS) to eliminate the functionalities of data stored in local drives.

Principles of Data Science

Unit – II

Data Collection and Data Pre-Processing

- Data Collection Strategies
- Data Pre-Processing Overview
- Data Cleaning
- Data Integration and Transformation
- Data Reduction
- Data Discretization

DATA PREPROCESSING

What is Data Preprocessing?

- Real-world databases are highly susceptible to noisy, missing, and inconsistent data due to their typically huge size (often several gigabytes or more) and their likely origin from multiple, heterogeneous sources. Low-quality data will lead to low-quality mining results, so we prefer a preprocessing concepts.

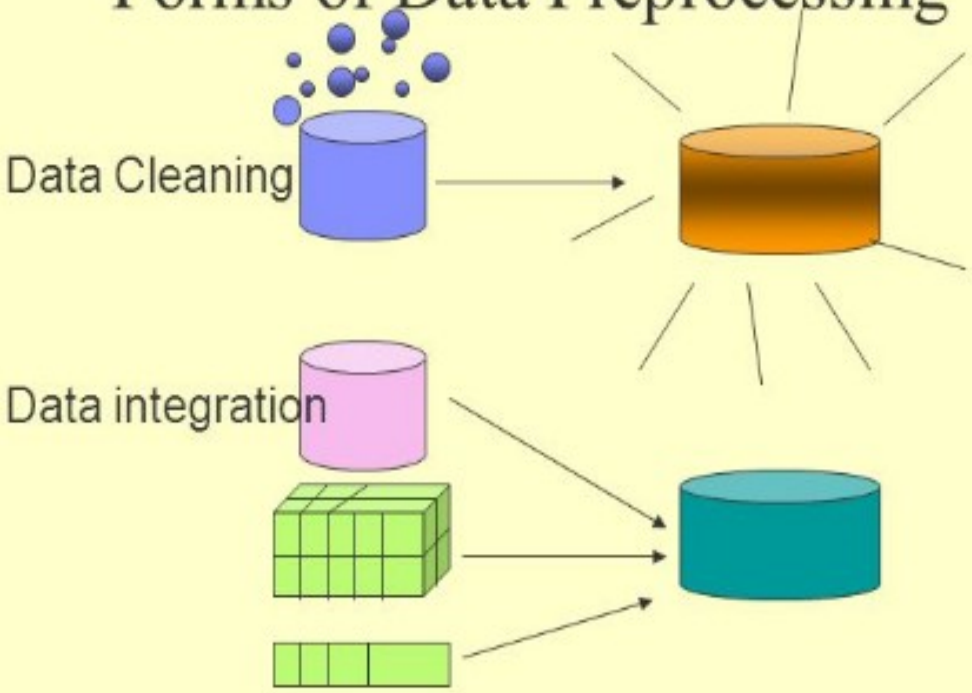
Data Preprocessing Techniques

- **Data cleaning** can be applied to remove noise and correct inconsistencies in the data.
- **Data integration** merges data from multiple sources into coherent data store, such as a data warehouse.
- **Data reduction** can reduce the data size by aggregating, eliminating redundant features, or clustering, for instance. These techniques are not mutually exclusive; they may work together.
- **Data transformations**, such as normalization, may be applied.

Need for preprocessing

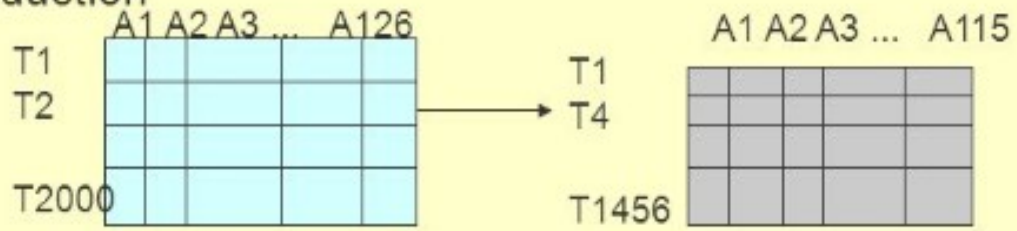
- Incomplete, noisy and inconsistent data are common place properties of large real world databases and data warehouses.
- Incomplete data can occur for a number of reasons:
  - Attributes of interest may not always be available
  - Relevant data may not be recorded due to misunderstanding, or because of equipment malfunctions.
  - Data that were inconsistent with other recorded data may have been deleted.
  - Missing data, particularly for tuples with missing values for some attributes, may need to be inferred.
  - The data collection instruments used may be faulty.
  - There may have been human or computer errors occurring at data entry.
  - Errors in data transmission can also occur.
  - There may be technology limitations, such as limited buffer size for coordinating synchronized data transfer and consumption.
  - Data cleaning routines work to ―clean‖ the data by filling in missing values, smoothing noisy data, identifying or removing outliers, and resolving inconsistencies.
  - Data integration is the process of integrating multiple databases cubes or files. Yet some attributes representing a given may have different names in different databases, causing inconsistencies and redundancies.
  - Data transformation is a kind of operations, such as normalization and aggregation, are additional data preprocessing procedures that would contribute toward the success of the mining process.
  - Data reduction obtains a reduced representation of data set that is much smaller in volume, yet produces the same(or almost the same) analytical results.

Forms of Data Preprocessing



Data transformation -2, 32, 100, 59, 48 → -0.02, 0.32, 1.00, 0.59, 0.48

Data reduction



DATA CLEANING

Real-world data tend to be incomplete, noisy, and inconsistent. Data cleaning (or data cleansing) routines attempt to fill in missing values, smooth out noise while identifying outliers and correct inconsistencies in the data.

Missing Values

Many tuples have no recorded value for several attributes, such as customer income. so we can fill the missing values for this attributes.

The following methods are useful for performing missing values over several attributes:

1. **Ignore the tuple:** This is usually done when the class label missing (assuming the mining task involves classification). This method is not very effective, unless the tuple contains several attributes with missing values. It is especially poor when the percentage of the missing values per attribute varies considerably.
2. **Fill in the missing values manually:** This approach is time –consuming and may not be feasible given a large data set with many missing values.
3. **Use a global constant to fill in the missing value:** Replace all missing attribute value by the same constant, such as a label like ―unknown‖ or -∞.
4. **Use the attribute mean to fill in the missing value:** For example, suppose that the average income of customers is $56,000. Use this value to replace the missing value for income.
5. **Use the most probable value to fill in the missing value:** This may be determined with regression, inference-based tools using a Bayesian formalism or decision tree induction. For example, using the other customer attributes in the sets decision tree is constructed to predict the missing value for income.

Noisy Data

Noise is a random error or variance in a measured variable. Noise is removed using data smoothing techniques.

Binning: Binning methods smooth a sorted data value by consulting its ―neighborhood,‖ that is the value around it. The sorted values are distributed into a number of ―buckets‖ or ―bins―. Because binning methods consult the neighborhood of values, they perform local smoothing.

Sorted data for price (in dollars): 3,7,14,19,23,24,31,33,38.

Example 1: Partition into (equal-frequency) bins:

Bin 1: 3,7,14

Bin 2: 19,23,24

Bin 3: 31,33,38

In the above method the data for price are first sorted and then partitioned into equal- frequency bins of size 3.

Smoothing by bin means:

Bin 1: 8,8,8

Bin 2: 22,22,22

Bin 3: 34,34,34

In smoothing by bin means method, each value in a bin is replaced by the mean value of the bin. For example, the mean of the values 3,7&14 in bin 1 is 8[(3+7+14)/3].

Smoothing by bin boundaries:

Bin 1: 3,3,14

Bin 2: 19,24,24

Bin 3: 31,31,38

In smoothing by bin boundaries, the maximum & minimum values in give bin or identify as the bin boundaries. Each bin value is then replaced by the closest boundary value.

Example 2: Remove the noise in the following data using smoothing techniques:

8, 4,9,21,25,24,29,26,28,15

Regression: Data can be smoothed by fitting the data to function, such as with regression. Linear regression involves finding the ―best‖ line to fit two attributes (or variables), so that one attribute can be used to predict the other. Multiple linear regressions is an extension of linear regression, where more than two attributes are involved and the data are fit to a multidimensional surface.

Clustering: Outliers may be detected by clustering, where similar values are organized into groups, or ―clusters. Intuitively, values that fall outside of the set of clusters may be considered outliers.

Inconsistent Data

Inconsistencies exist in the data stored in the transaction. Inconsistencies occur due to occur during data entry, functional dependencies between attributes and missing values. The inconsistencies can be detected and corrected either by manually or by knowledge engineering tools.

Data cleaning as a process

a) Discrepancy detection

b) Data transformations

a) Discrepancy detection

The first step in data cleaning is discrepancy detection. It considers the knowledge of meta data and examines the following rules for detecting the discrepancy.

- **Unique rules-** each value of the given attribute must be different from all other values for that attribute.
- **Consecutive rules –** Implies no missing values between the lowest and highest values for the attribute and that all values must also be unique.
- **Null rules -** specifies the use of blanks, question marks, special characters, or other strings that may indicates the null condition.

Discrepancy detection Tools:

- **Data scrubbing tools** - use simple domain knowledge (e.g., knowledge of postal addresses, and spell-checking) to detect errors and make corrections in the data.
- **Data auditing tools** – analyzes the data to discover rules and relationship, and detecting data that violate such conditions.

b) Data transformations

This is the second step in data cleaning as a process. After detecting discrepancies, we need to define and apply (a series of) transformations to correct them.

Data Transformations Tools:

- Data migration tools – allows simple transformation to be specified, such to replaced the string ―gender‖ by ―sex‖.
- ETL (Extraction/Transformation/Loading) tools – allows users to specific transforms through a graphical user interface(GUI)

DATA INTEGRATION

- Data mining often requires data integration - the merging of data from stores into a coherent data store, as in data warehousing. These sources may include multiple data bases, data cubes, or flat files.

Issues in Data Integration

a) Schema integration & object matching.

b) Redundancy.

c) Detection & Resolution of data value conflict

a) Schema Integration & Object Matching

- Schema integration & object matching can be tricky because same entity can be represented in different forms in different tables. This is referred to as the entity identification problem.
- Metadata can be used to help avoid errors in schema integration. The meta data may also be used to help transform the data.

b) Redundancy:

Redundancy is another important issue an attribute (such as annual revenue, for instance)may be redundant if it can be ―derived‖ from another attribute are set of attributes. Inconsistencies in attribute of dimension naming can also cause redundancies in the resulting data set. Some redundancies can be detected by correlation analysis and covariance analysis.

- For Nominal data, we use the χ^2 (Chi-Square) test.
- For Numeric attributes we can use the correlation coefficient and covariance.

χ^2 Correlation analysis for numerical data:

- For nominal data, a correlation relationship between two attributes, A and B, can be discovered by a χ^2 (Chi-Square) test. Suppose A has c distinct values, namely a1, a2, a3,……., ac. B has r distinct values, namely b1, b2, b3, …., br. The data tuples are described by table.
- The χ^2 value is computed as,

$$\chi^2 = \sum_{i=1}^{c} \sum_{j=1}^{r} \frac{o_{ij} - e_{ij}}{e_{ij}}$$

Where o_{ij} is the observed frequency of the joint event (A_i,B_j) and e_{ij} is the expected frequency of (A_i,B_j), which can computed as

$$e_{ij} = \frac{count\ A=a_i\ X count(B=b_j)}{n}$$

For Example,

	Male	Female	Total
Fiction	250	200	450
Non_Fiction	50	1000	1050
Total	300	1200	1500

$$e_{11} = \frac{count\ male\ X count\ (fiction)}{n} = \frac{300X450}{1500} = 90$$

$$e_{12} = \frac{count\ male\ X count\ (non\_fiction)}{n} = \frac{300X1050}{1500} = 210$$

$$e_{21} = \frac{count\ female\ X count\ (fiction)}{n} = \frac{1200\ X450}{1500} = 360$$

$$e_{22} = \frac{count\ female\ X count\ (non\_fiction)}{n} = \frac{1200\ X1050}{1500} = 840$$

	Male	Female	Total
Fiction	250 (90)	200 (360)	450
Non_Fiction	50 (210)	1000 (840)	1050
Total	300	1200	1500

For $\chi 2$ computation, we get

$$\chi 2 = \frac{(250-90)^2}{90} + \frac{(50-210)^2}{210} + \frac{200-360^2}{360} + \frac{(1000-840)^2}{840}$$

$$= 284.44 + 121.90 + 71.11 + 30.48 = 507.93$$

Correlation Coefficient for Numeric data:

For Numeric attributes, we can evaluate the correlation between two attributes, A and B, by computing the correlation coefficient. This is

$$r_{A,B} = \frac{\sum_{i=1}^{n} (a_i - A)(b_i - B)}{n\sigma_A\sigma_B} = \frac{\sum_{i=1}^{n} a_i b_i - nAB}{n\sigma_A\sigma_B}$$

For Covariance between A and B defined as

$$Cov(A,B) = \frac{\sum_{i=1}^{n} (a_i - A)(b_i - B)}{n}$$

C) Detection and Resolution of Data Value Conflicts

A third important issue in data integration is the detection and resolution of data Value conflicts. For example, for the same real–world entity, attribute value from different sources may differ. This may be due to difference in representation, scaling, or encoding.

Careful integration of the data from multiple sources can help to reduce and avoid redundancies and inconsistencies in the resulting data set. This can help to improve the accuracy and speed of the subsequent of mining process.

Data Reduction

Obtain a reduced representation of the data set that is much smaller in volume but yet produces the same (or almost the same) analytical results.

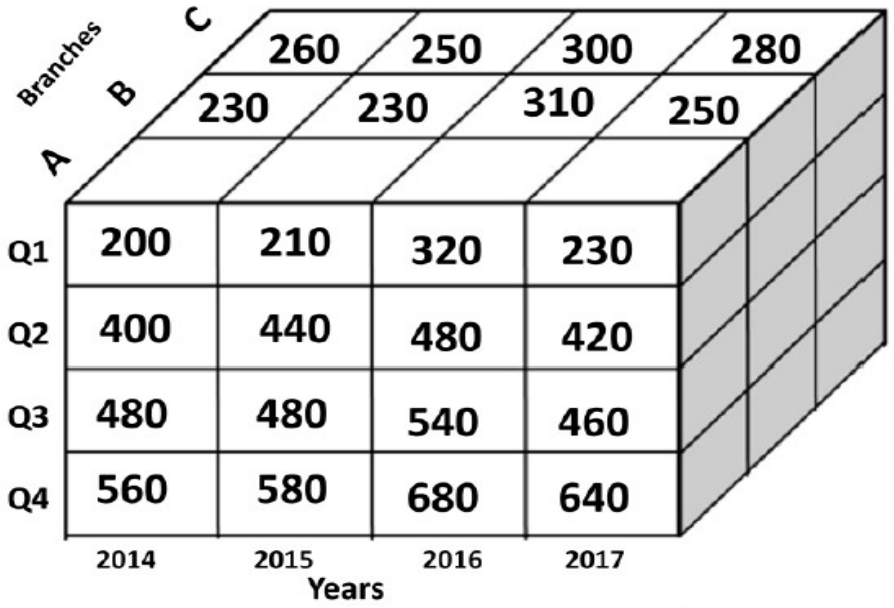
Why data reduction? — A database/data warehouse may store terabytes of data. Complex data analysis may take a very long time to run on the complete data set.

Data reduction strategies

- Data cube aggregation
- Attribute Subset Selection
- Numerosity reduction — e.g., fit data into models
- Dimensionality reduction - Data Compression

Data cube aggregation:

For example, the data consists of All Electronics sales per quarter for the years 2014 to 2017.You are, however, interested in the annual sales, rather than the total per quarter. Thus, the data can be aggregated so that the resulting data summarize the total sales per year instead of per quarter.



Year/Quarter	2014	2015	2016	2017
Quarter 1	200	210	320	230
Quarter 2	400	440	480	420
Quarter 3	480	480	540	460
Quarter 4	560	580	680	640

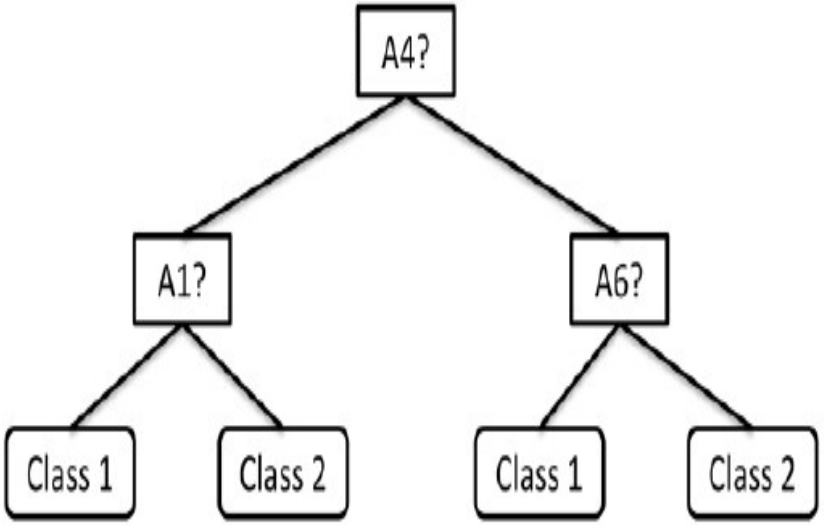
Year	Sales
2014	1640
2015	1710
2016	2020
2017	1750

Attribute Subset Selection

- Attribute subset selection reduces the data set size by removing irrelevant or redundant attributes (or dimensions). The goal of attribute subset selection is to find a minimum set of attributes.
- It reduces the number of attributes appearing in the discovered patterns, helping to make the patterns easier to understand. For n attributes, there are 2n possible subsets.

Techniques for heuristic methods of attribute sub set selection

1. **Stepwise forward selection:** The procedure starts with an empty set of attributes as the reduced set. The best of original attributes is determined and added to the reduced set. At each subsequent iteration or step, the best of the remaining original attributes is added to the set.
2. **Stepwise backward elimination:** The procedure starts with full set of attributes. At each step, it removes the worst attribute remaining in the set.
3. **Combination of forward selection and backward elimination:** The stepwise forward selection and backward elimination methods can be combined so that, at each step, the procedure selects the best attribute and removes the worst from among the remaining attributes.
4. **Decision tree induction:** Decision tree induction constructs a flowchart like structure where each internal node denotes a test on an attribute, each branch corresponds to an outcome of the test, and each leaf node denotes a class prediction.

Initial attribute set: {A1, A2, A3, A4, A5, A6} Initial Reduced Set: ➢ { } ➢ {A1} ➢ {A1, A4} ➢ {A1, A4, A6} Reduced Attribute Set: {A1, A4, A6}	Initial attribute set: {A1, A2, A3, A4, A5, A6} {A1, A2, A3, A4, A5, A6} {A1, A3, A4, A5, A6} {A1, A4, A5, A6} {A1, A4, A6} Reduced Attribute Set: {A1, A4, A6}	Initial attribute set: {A1, A2, A3, A4, A5, A6}  Reduced Attribute Set: {A1, A4, A6}

Numerosity Reduction:

Numerosity reduction is used to reduce the data volume by choosing alternative, smaller forms of the data representation

Techniques for Numerosity reduction:

- Parametric - In this model only the data parameters need to be stored, instead of the actual data. (e.g.,) Log-linear models, Regression.
- Nonparametric – This method stores reduced representations of data include histograms, clustering, and sampling Parametric model.

Parametric model

1. **Regression**
   - **Linear regression**
     - In linear regression, the data are model to fit a straight line. For example, a random variable, Y called a response variable), can be modeled as a linear function of another random variable, X called a predictor variable), with the equation $Y=\alpha X+\beta$
     - Where the variance of Y is assumed to be constant. The coefficients, α and β (called regression coefficients), specify the slope of the line and the Y- intercept, respectively.
   - **Multiple- linear regression**
     - Multiple linear regression is an extension of (simple) linear regression, allowing a response variable Y, to be modeled as a linear function of two or more predictor variables.
2. ***Log-Linear Models***
   - Log-Linear Models can be used to estimate the probability of each point in a multidimensional space for a set of discretized attributes, based on a smaller subset of dimensional combinations.

Nonparametric Model

1. Histograms

A histogram for an attribute A partitions the data distribution of A into disjoint subsets, or buckets. If each bucket represents only a single attribute-value/frequency pair, the buckets are called singleton buckets.

Ex: The following data are bast of prices of commonly sold items at All Electronics. The numbers have been sorted:

1,1,5,5,5,5,5,8,8,10,10,10,10,12,14,14,14,15,15,15,15,15,18,18,18,18,18,18,18,18,20,20,20,20,20,20,21,21,21,21,21,25,25,25,25,25,28,28,30,30,30

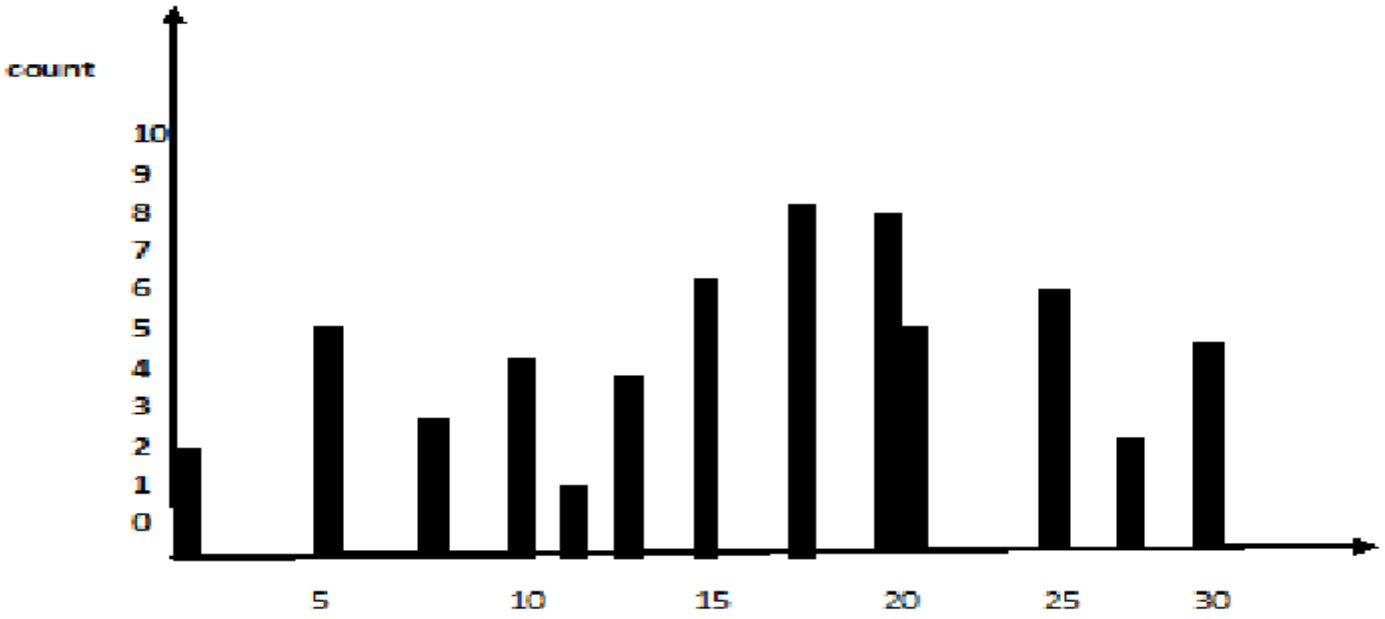


Figure 3.7 A Histogram for price using Singleton Buckets

Equal-width: The width of each bucket range is uniform. (Equal-frequency (or equi-depth): the frequency of each bucket is constant.

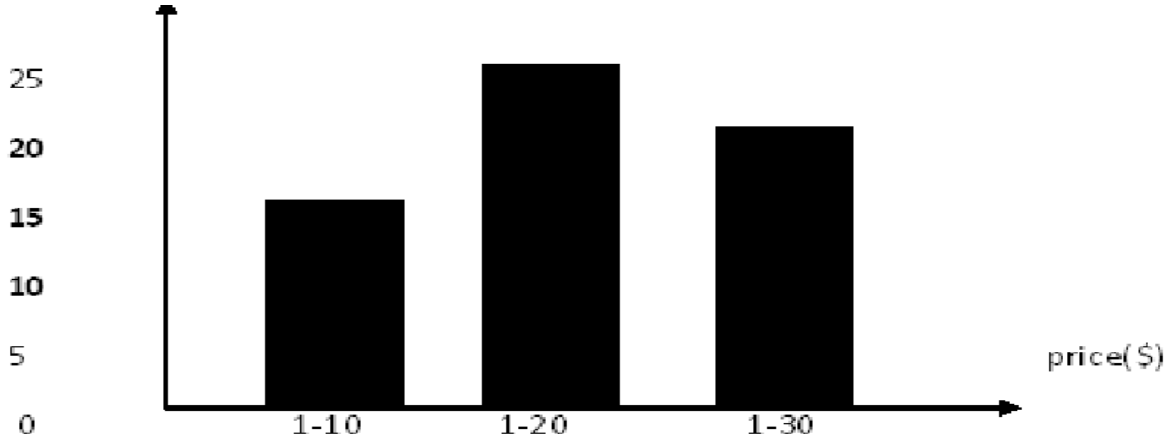
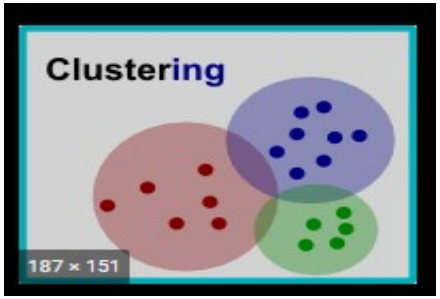


Figure.2.8 An equal-width histogram for price, where values are aggregated so ***that each bucket has a uniform width of $10***.

2. Clustering: Clustering technique consider data tuples as objects. They partition the objects into groups or clusters, so that objects within a cluster are similar to one another and dissimilar to objects in other clusters.



3. Sampling:

- Sampling can be used as a data reduction technique because it allows a large data set to be represented by a much smaller random sample (or subset) of the data.
- Suppose that a large data set D, contains N tuples, then the possible samples are Simple Random sample without Replacement (SRS WOR) of size n: This is created by drawing „n. of the „N. tuples from D (n<N), where the probability of drawing any tuple in D is 1/N, i.e., all tuples are equally likely to be sampled.

T30	Young
T200	Young
	Young
T250	Young
T320	Middle-aged
T90	Middle-aged
T150	Middle-aged
T260	Middle-aged
T300	Senior
T60	Senior
T275	

T30	Young
T320	Young
	Middle-aged
T20	Middle-aged
T260	Middle-aged
T300	Senior
T60	

Figure 2.9. Sampling can be used for data reduction.

Dimensionality Reduction:

In dimensionality reduction, data encoding or transformations are applied so as to obtained reduced or compressed. representation of the oriental data.

Dimension Reduction Types

- Lossless - If the original data can be reconstructed from the compressed data without any loss of information.
- Lossy - If the original data can be reconstructed from the compressed data with loss of information, then the data reduction is called lossy. *Effective methods in lossy dimensional reduction:*

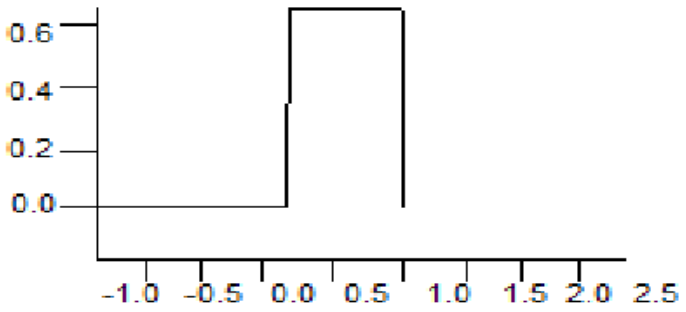
a) Wavelet transforms

b) Principal components analysis.

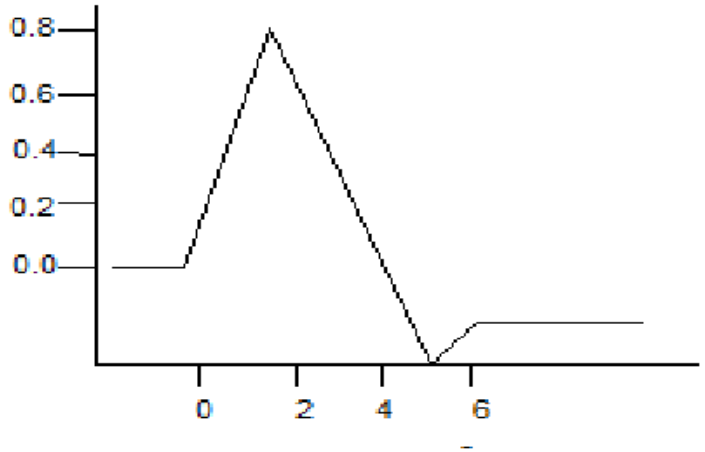
a) Wavelet transforms:

- The discrete wavelet transform (DWT) is a linear signal processing technique that, when applied to a data vector, transforms it to a numerically different vector, of wavelet coefficients.
- The two vectors are of the same length. When applying this technique to data reduction, we consider each tuple as an n-dimensional data vector, that is, X=(x1,x2,……,xn), depicting n measurements made on the tuple from n database attributes.

1. The length, L, of the input data vector must be an integer power of 2. This condition can be met by padding the data vector with zeros as necessary (L >=n).
2. Each transform involves applying two functions
   - The first applies some data smoothing, such as a sum or weighted average.
   - The second performs a weighted difference, which acts to bring out the detailed features of data.
3. The two functions are applied to pairs of data points in X, that is, to all pairs of measurements (X2i , X2i+1). This results in two sets of data of length L/2. In general, these represent a smoothed or low-frequency version of the input data and high frequency content of it, respectively.
4. The two functions are recursively applied to the sets of data obtained in the previous loop, until the resulting data sets obtained are of length 2.



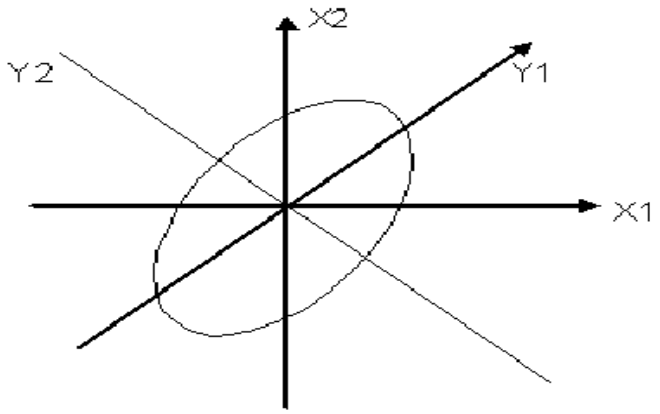
(a)haar2



(b) Daubechies4

b) Principal components analysis

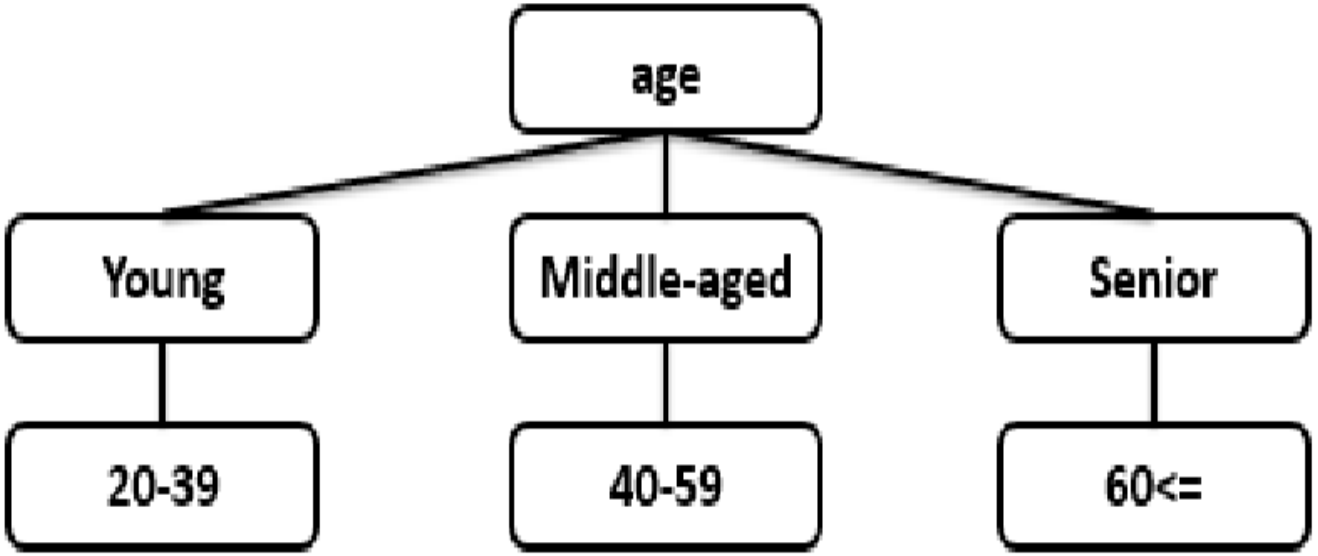
- Principal components analysis, or PCA (also called the Karhunen-Loeve, or K-L, method), searches for k n-dimensional orthogonal vectors that can best be used to represent the data where k<=n. PCA combines the essence of attributes by creating an alternative, smaller set of variables. The initial data can then be projected onto this smaller set.
- The basic procedure is as follows:
    1. The input data are normalized.
    2. PCA computes *k orthonormal vectors that provide a basis for the normalized input* data. These are unit vectors that each point in a direction perpendicular to the others.
    3. The principal components are sorted in order of decreasing significance or strength.



Data Transformation and Discretization

- Data transformation is the process of converting data from one format or structure into another format or structure.
- In *data transformation, the data are transformed or consolidated into forms appropriate for mining.* Strategies for data transformation include the following:
  1. **Smoothing, which works to remove noise from the data. Techniques include binning,** regression, and clustering.
  2. **Attribute construction (or *feature construction), where new attributes are constructed and*** added from the given set of attributes to help the mining process.
  3. **Aggregation, where summary or aggregation operations are applied to the data. For** example, the daily sales data may be aggregated so as to compute monthly and annual total amounts. This step is typically used in constructing a data cube for data analysis at multiple abstraction levels.
  4. **Normalization, where the attribute data are scaled so as to fall within a smaller range, such** as .1.0 to 1.0, or 0.0 to 1.0.

5. **Discretization, where the raw values of a numeric attribute (e.g., *age) are replaced*** byinterval labels (e.g., 0–10, 11–20, etc.) or conceptual labels (e.g., *youth, adult, senior).The*labels, in turn, can be recursively organized into higher-level concepts, resulting in a *concept hierarchy for the numeric attribute.*
6. **Concept hierarchy generation for nominal data, where attributes such as *street can be*** generalized to higher-level concepts, like *city or country. Many hierarchies for nominal* attributes are implicit within the database schema and can be automatically defined at the schema definition level.



Data Transformation by Normalization:

- The measurement unit used can affect the data analysis. For example, changing measurement units from meters to inches for height, or from kilograms to pounds for weight, may lead to very different results.
- There are many methods for data normalization. We study min-max normalization, z-score normalization, and normalization by decimal scaling. For our discussion, let A be a numeric attribute with n observed values, v1, v2, …., vn.

a) Min-max normalization performs a linear transformation on the original data. Suppose that minA and maxA are the minimum and maximum values of an attribute, A. Min-max normalization maps a value, vi, of A to vi'in the range [new_minA, new_maxA]by computing Min-max normalization preserves the relationships among the original data values. It will encounter

$$v_i' = \frac{v_i - min_A}{max_A - min_A}(new\_max_A - new\_min_A) + new\_min_A.$$

an –out-of-bounds‖ error if a future input case for normalization falls outside of the original data range for *A*.

- **Example:-Min-max normalization. Suppose that the minimum and maximum values for the** attribute income are $12,000 and $98,000, respectively. We would like to map income to the range [0.0, 1.0]. By min-max normalization, a value of $73,600 for income is transformed to

$$\frac{73{,}600 - 12{,}000}{98{,}000 - 12{,}000}(1.0 - 0) + 0 = 0.716.$$

b) Z-Score Normalization

- The values for an attribute, *A, are normalized based on the mean (i.e., average) and standard deviation* of *A. A value, vi, of A is normalized to vi' by computing*

$$v_i' = \frac{v_i - \bar{A}}{\sigma_A}$$

- **Example z-score normalization. Suppose that the mean and standard deviation of the values for** the attribute income are $54,000 and $16,000, respectively. With z-score normalization, a value of $73,600 for income is transformed to

$$\frac{73,600 - 54,000}{16,000} = 1.225.$$

c) Normalization by Decimal Scaling:

- Normalization by decimal scaling normalizes by moving the decimal point of values of attribute A. The number of decimal points moved depends on the maximum absolute value of A. A value, vi, of A is normalized to vi' by computing

$$v_i' = \frac{v_i}{10^j}$$

where j is the smallest integer such that max(|vi')< 1.

- **Example Decimal scaling. Suppose that the recorded values of A range from -986 to 917. The** maximum absolute value of A is 986. To normalize by decimal scaling, we therefore divide each value by 1000 (i.e., j = 3) so that -986 normalizes to -0.986 and 917normalizes to 0.917.

Data Discretization

a) Discretization by binning:

- Binning is a top-down splitting technique based on a specified number of bins. For example, attribute values can be discretized by applying equal-width or equal-frequency binning, and then replacing each bin value by the bin mean or median, as in *smoothing by bin means or smoothing by bin medians, respectively.*

b) Discretization by Histogram Analysis:

- Like binning, histogram analysis is an unsupervised discretization technique because it does not use class information. A histogram partitions the values of an attribute, A, into disjoint ranges called buckets or bins.

c) Discretization by Cluster, Decision Tree, and Correlation Analyses

- Cluster analysis is a popular data discretization method. A clustering algorithm can be applied to discretize a numeric attribute, A, by partitioning the values of A into clusters or groups.

Concept Hierarchy Generation for Nominal Data

- Nominal attributes have a finite (but possibly large) number of distinct values, with no ordering among the values. Examples include geographic location, job category, and item type.
- The concept hierarchies can be used to transform the data into multiple levels of granularity.

1. **Specification of a partial ordering of attributes explicitly at the schema level by users or experts: A user or expert can easily define a concept hierarchy by specifying** a partial or total ordering of the attributes at the schema level.
2. **Specification of a portion of a hierarchy by explicit data grouping: In a large** database, it is unrealistic to define an entire concept hierarchy by explicit value enumeration. For example, after specifying that province and country form a hierarchy at the schema level, a user could define some intermediate levels manually.
3. **Specification of a set of attributes, but not of their partial ordering: A user may** specify a set of attributes forming a concept hierarchy, but omit to explicitly State their partial ordering. The system can then try to automatically generate the attribute ordering so as to construct a meaningful concept hierarchy.
4. **Specification of only a partial set of attributes: Sometimes a** user can be careless when defining a hierarchy, or have only a vague idea about what should be included in a hierarchy. Consequently, the user may have included only a small subset of there irrelevant attributes in the hierarchy specification.

- **Data cleaning routines attempt to fill in missing values, smooth** out noise while identifying outliers, and correct inconsistencies in the data.
- **Data integration combines data from multiple sources to form** a coherent data store. The resolution of semantic heterogeneity, metadata, correlation analysis ,tuple duplication detection, and data conflict detection contribute to smooth data integration.
- **Data reduction techniques obtain a reduced representation of** the data while minimizing the loss of information content. These include methods of *dimensionality reduction, numerosity reduction, and data compression.*
- **Data transformation routines convert the data into appropriate** forms for mining. For example, in **normalization, attribute data** are scaled so as to fall within a small range such as 0.0 to 1.0. Other examples are **data discretization and concept hierarchy generation.**
- **Data discretization transforms numeric data by mapping values** to interval or concept labels. Such methods can be used to automatically generate *concept hierarchies for the data, which* allows for mining at multiple levels of granularity.


JSON
XML
CSV
TXT
SQL
1010101010
1010101010
1010101010
1010101010
1010101010
1010101010

Principles of Data Science

Unit – III: Exploratory Data Analytics

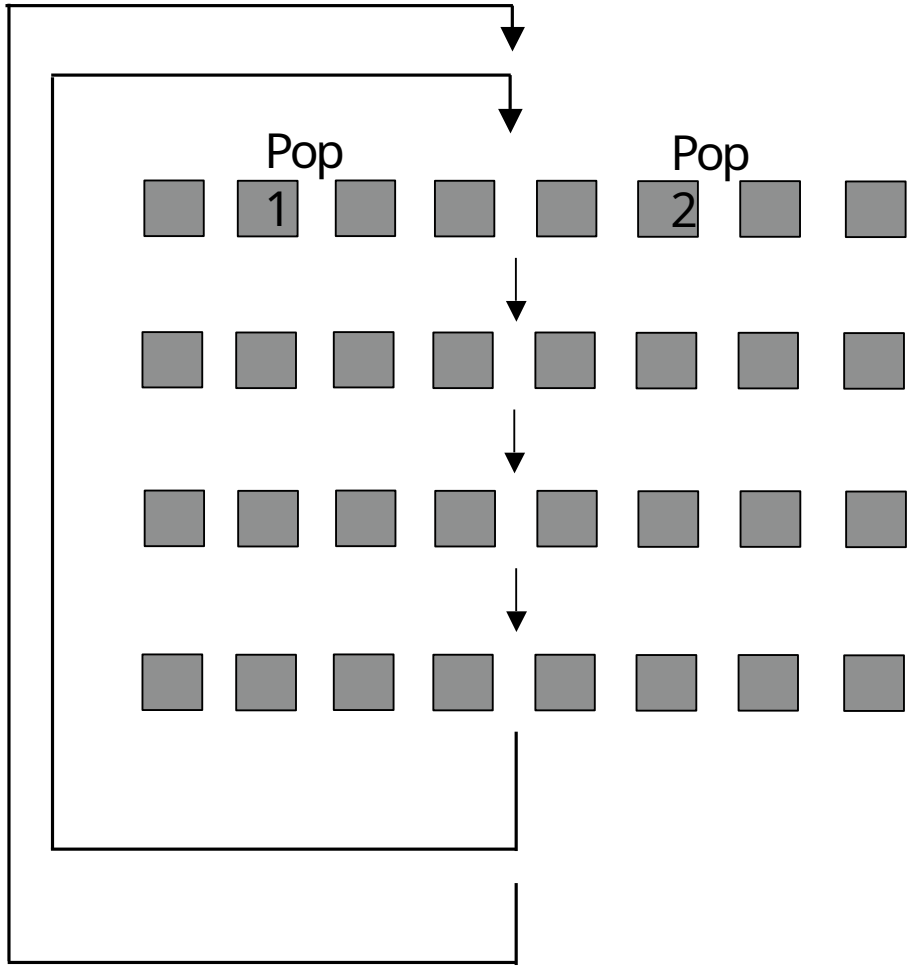
- Descriptive Statistics
- Mean, Standard Deviation, Skewness and Kurtosis
- Box Plots
- Pivot Table
- Heat Map
- Correlation Statistics
- ANOVA.

Exploratory Data Analysis

- In statistics, exploratory data analysis (EDA) is an approach to analyzing data sets to summarize their main characteristics, often with visual methods.
- A statistical model can be used or not, but primarily EDA is for seeing what the data can tell us beyond the formal modeling or hypothesis testing task.
- Exploratory data analysis is a concept developed by John Tuckey (1977) that consists on a new perspective of statistics.

Further Thoughts on Experimental Design

- 16 Individuals (8 each from two populations) with replicates



Randomly sample 4 individuals from each pop

Tissue culture and RNA extraction

Labeling and array hybridization

Slide scanning and data acquisition

Repeat 2 times processing 16 samples in total

Other Business

- Course web-site:

  http://www.gs.washington.edu/academics/courses/akey/56008/index.htm

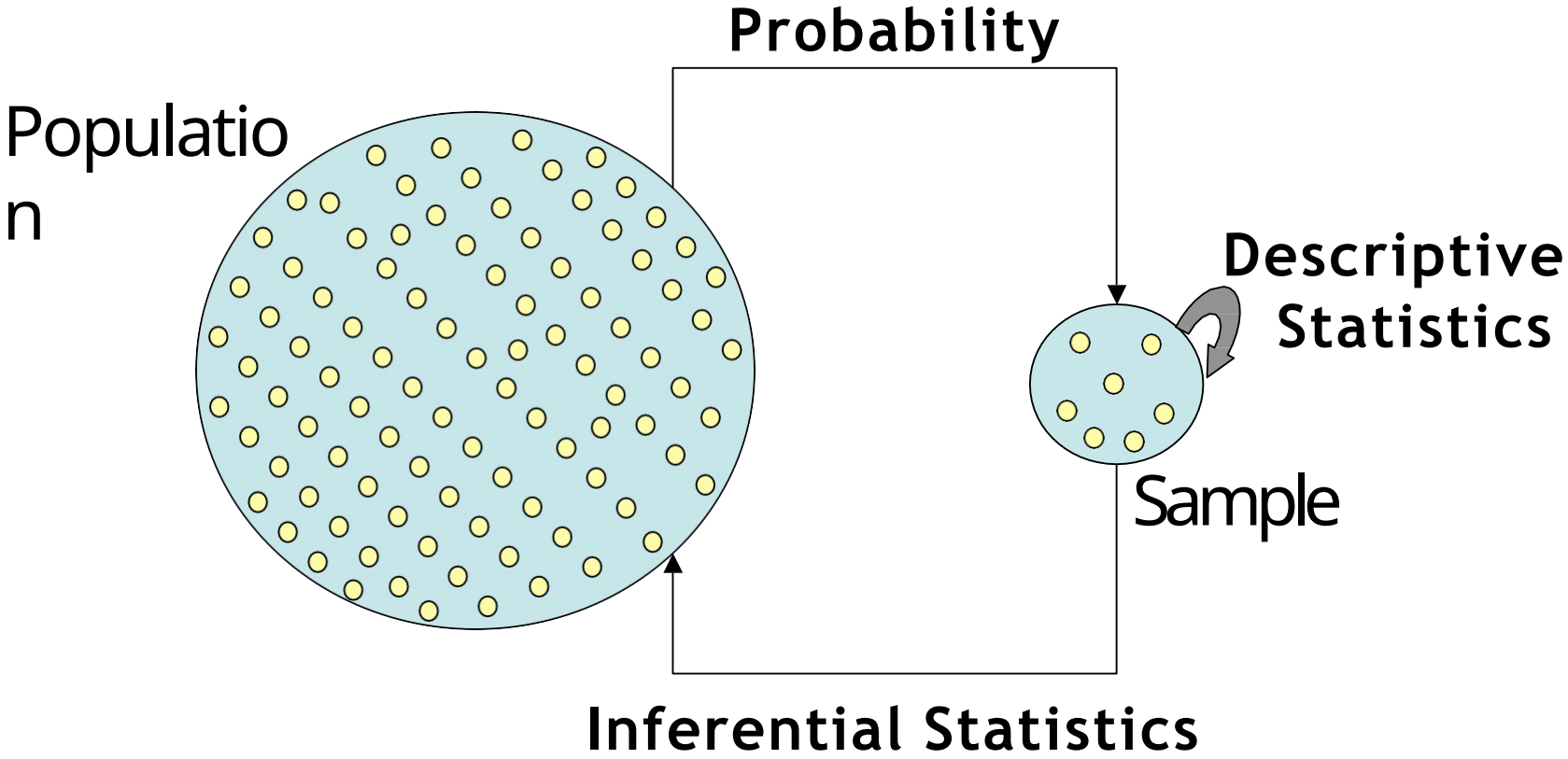
- Homework due on Thursday not Tuesday

- Make sure you look at HW1 soon and see either Shameek or myself with questions

Today

- What is descriptive statistics and exploratory data analysis?
- Basic numerical summaries of data
- Basic graphical summaries of data
- How to use R for calculating descriptive statistics and making graphs

“Central Dogma” of Statistics



EDA

Before making inferences from data it is essential to examine all your variables.

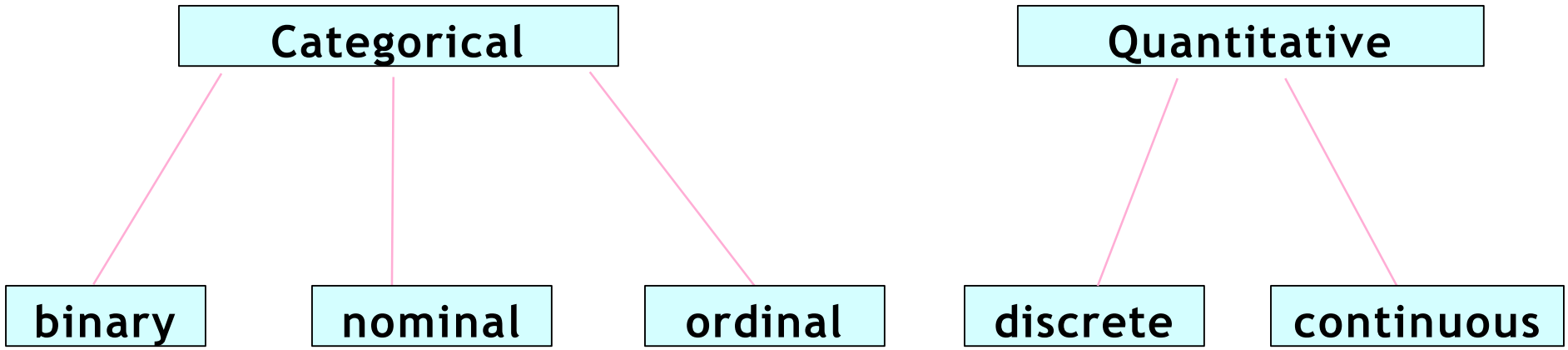
Why?

To ***listen*** to the data:

- to catch mistakes
- to see patterns in the data
- to find violations of statistical assumptions
- to generate hypotheses

...and because if you don't, you will have trouble later

Types of Data



2 categories

more categories

order matters

numerical

uninterrupted

Dimensionality of Data Sets

- **Univariate:** Measurement made on one variable per subject

- **Bivariate:** Measurement made on two variables per subject

- **Multivariate:** Measurement made on many variables per subject

Numerical Summaries of Data

- ***Central Tendency measures***. They are computed to give a "center" around which the measurements in the data are distributed.

- ***Variation or Variability measures***. They describe "data spread" or how far away the measurements are from the center.

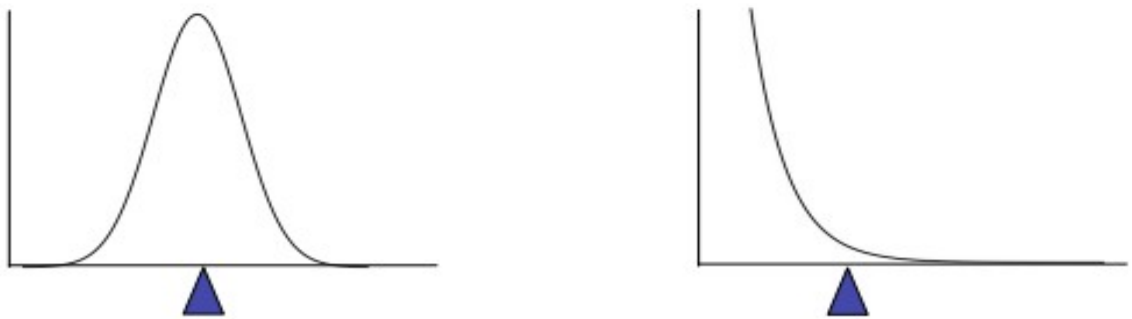
- ***Relative Standing measures***. They describe the relative position of specific measurements in the data.

Location: Mean

1. The Mean

To calculate the average $\bar{x}$ of a set of observations, add their value and divide by the number of observations:

$$\bar{x} = \frac{x_1 + x_2 + x_3 + \ldots + x_n}{n} = \frac{1}{n}\sum_{i=1}^{n} x_i$$



Other Types of Means

Weighted means:

$$\bar{x} = \frac{\sum_{i=1}^{n} w_i x_i}{\sum_{i=1}^{n} w_i}$$

Trimmed:

$$\bar{x} = \alpha$$

Geometric:

$$\bar{x} = \left(\prod_{i=1}^{n} x_i \right)^{\frac{1}{n}}$$

Harmonic:

$$\bar{x} = \frac{n}{\sum_{i=1}^{n} \frac{1}{x_i}}$$

Location: Median

- **Median** – the exact middle value

- **Calculation**:
  - If there are an odd number of observations, find the middle value

  - If there are an even number of observations, find the middle two values and average them

- **Example**

  Some data:

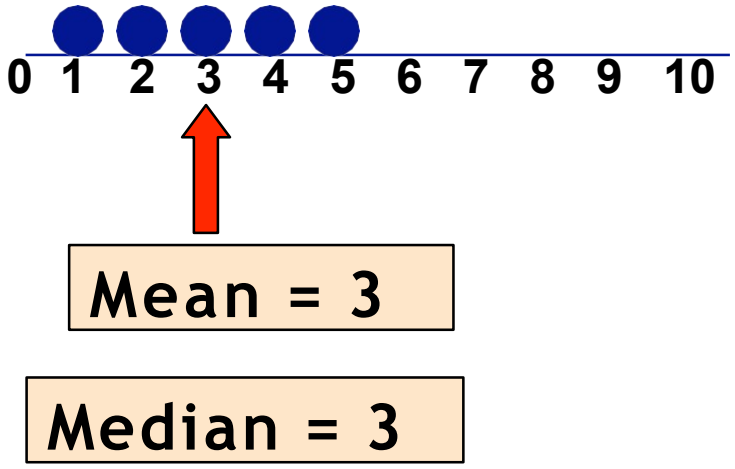
  Age of participants: 19 21 22 23 23 23 17 38

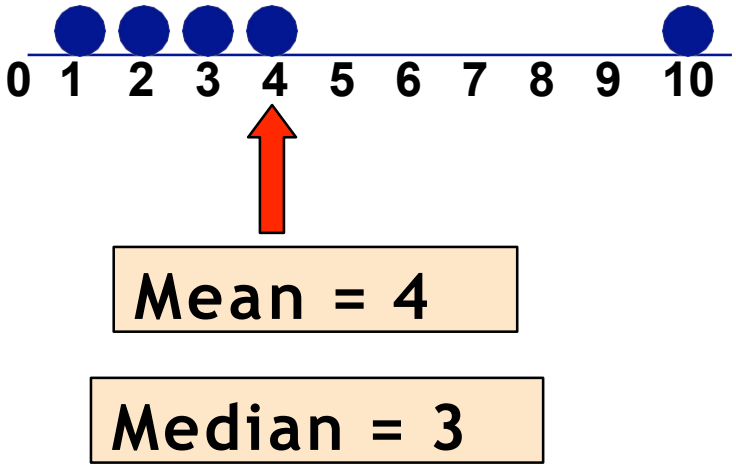
  Median = (22+23)/2 = 22.5

Which Location Measure Is Best?

- Mean is best for symmetric distributions without outliers

- Median is useful for skewed distributions or data with outliers





Scale: Variance

- Average of squared deviations of values from the mean

$$\sigma^2 = \frac{\sum_{i}^{n} (x_i - \bar{x})^2}{n-1}$$

Why Squared Deviations?

- Adding deviations will yield a sum of ?
- Absolute values do not have nice mathematical properties
- Squares eliminate the negatives

- Result:
  - Increasing contribution to the variance as you go farther from the mean.

Scale: Standard Deviation

- Variance is somewhat arbitrary
- What does it mean to have a variance of 10.8? Or 2.2? Or 1459.092? Or 0.000001?
- Nothing. But if you could "standardize" that value, you could talk about any variance (i.e. deviation) in equivalent terms
- Standard deviations are simply the square root of the variance

Scale: Standard Deviation

$$\hat{\sigma} = \sqrt{\frac{\sum_{i}^{n}(x^{i} - \bar{x})^{2}}{n-1}}$$

1. Score (in the units that are meaningful)
2. Mean
3. Each score's deviation from the mean
4. Square that deviation
5. Sum all the squared deviations (Sum of Squares)
6. Divide by n-1
7. Square root – now the value is in the units we started with!!!

Interesting Theoretical Result

- Regardless of how the data are distributed, a certain percentage of values must fall within k standard deviations from the mean:

Note use of μ (mu) to represent "mean".

Note use of σ (sigma) to represent "standard deviation."

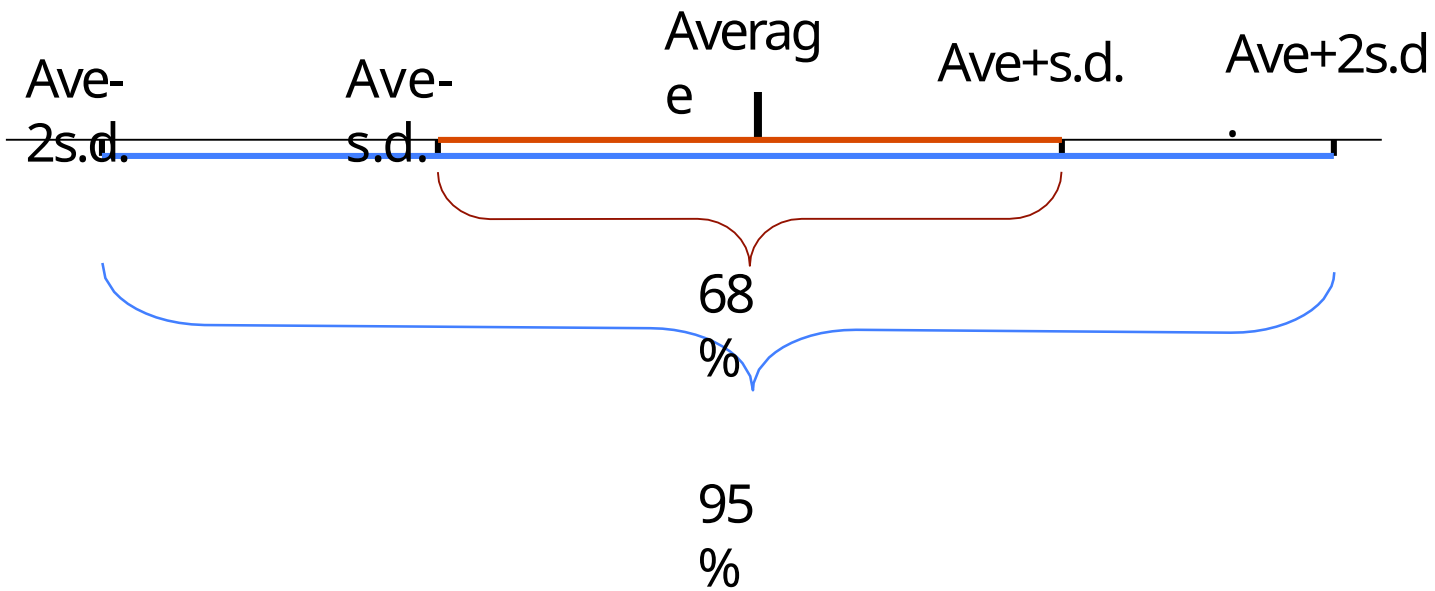
At least	within
$(1 - 1/1^2) = 0\%$	$k=1$ $(\mu \pm 1\sigma)$
$(1 - 1/2^2) = 75\%$	$k=2$ $(\mu \pm 2\sigma)$
$(1 - 1/3^2) = 89\%$	$k=3$ $(\mu \pm 3\sigma)$

Often We Can Do Better

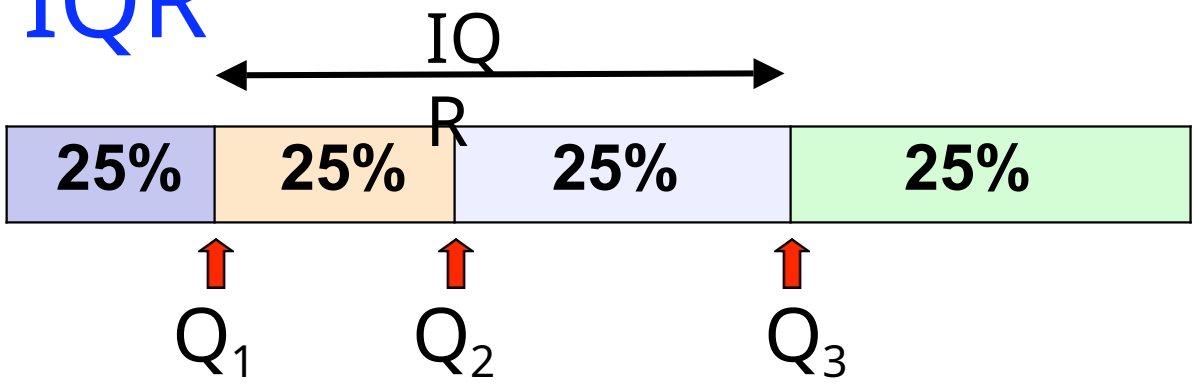
For many lists of observations – especially if their histogram is bell-shaped

1. Roughly 68% of the observations in the list lie within 1 standard deviation of the average

2. 95% of the observations lie within 2 standard deviations of the average



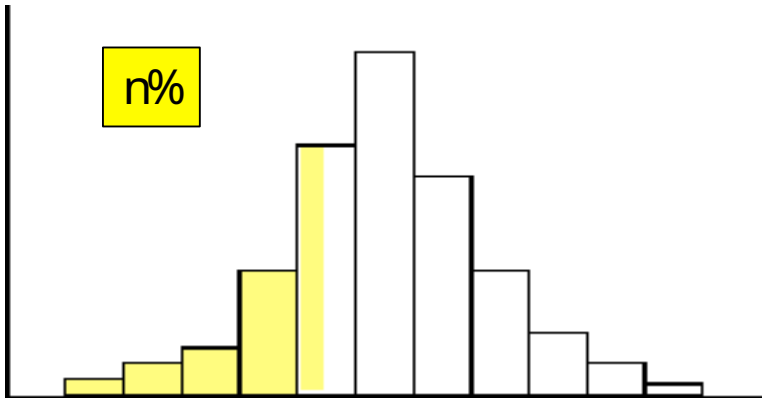
Scale: Quartiles and IQR



- The first quartile, Q_1, is the value for which 25% of the observations are smaller and 75% are larger
- Q_2 is the same as the median (50% are smaller, 50% are larger)
- Only 25% of the observations are greater than the third quartile

Percentiles (aka Quantiles)

In general the **n^{th} percentile** is a value such that n% of the observations fall at or below or it



Q_1 = 25th percentile

Median = 50th percentile

Q_2 = 75th percentile

Graphical Summaries of Data

A (Good) Picture Is Worth A 1,000 Words

Univariate Data: Histograms and Bar Plots

- What's the difference between a histogram and bar plot?

Bar plot

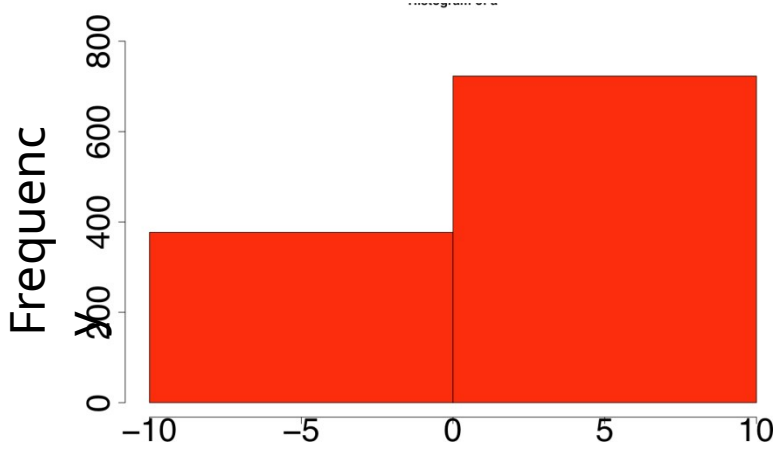
- Used for categorical variables to show frequency or proportion in each category.
- Translate the data from frequency tables into a pictorial representation…

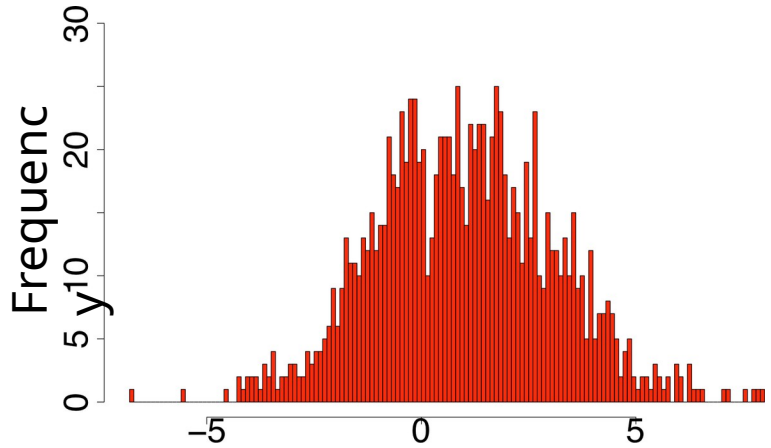
Histogram

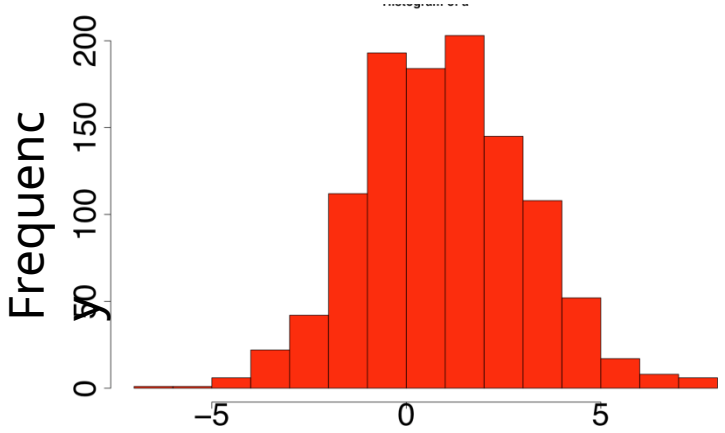
- Used to visualize distribution (shape, center, range, variation) of continuous variables
- "Bin size" important

Effect of Bin Size on Histogram

- Simulated 1000 N(0,1) and 500 N(1,1)

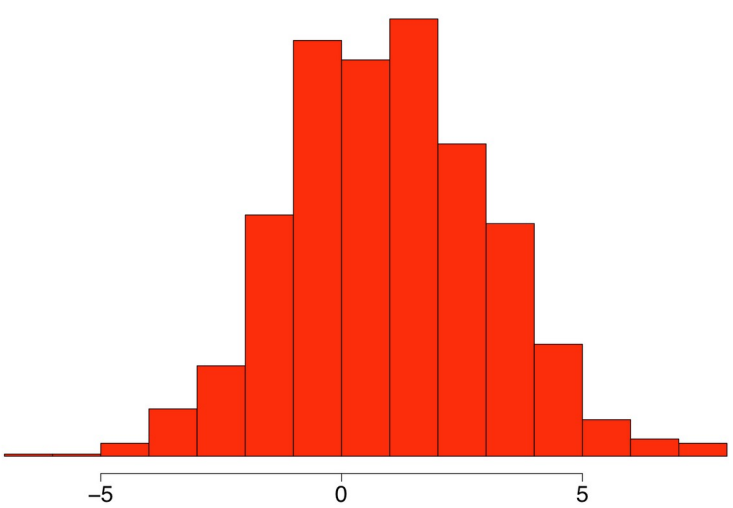


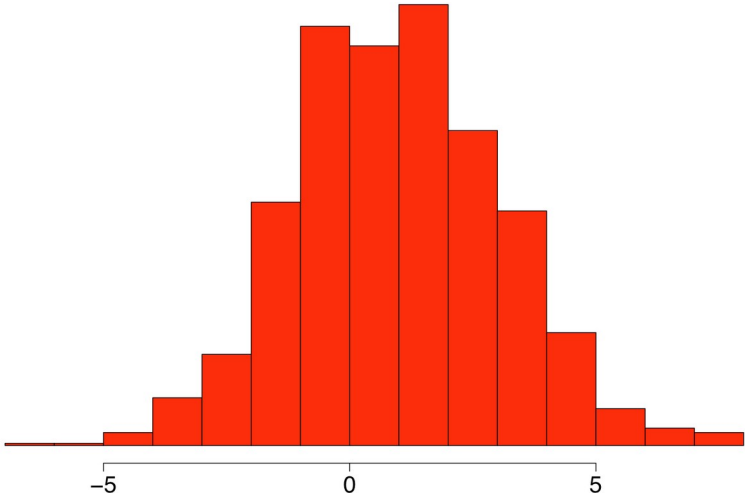




More on Histograms

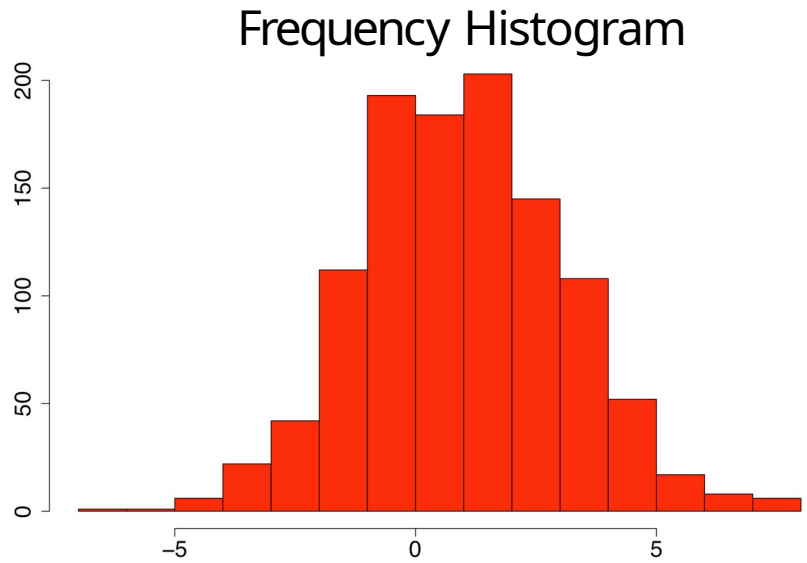
- What's the difference between a frequency histogram and a density histogram?

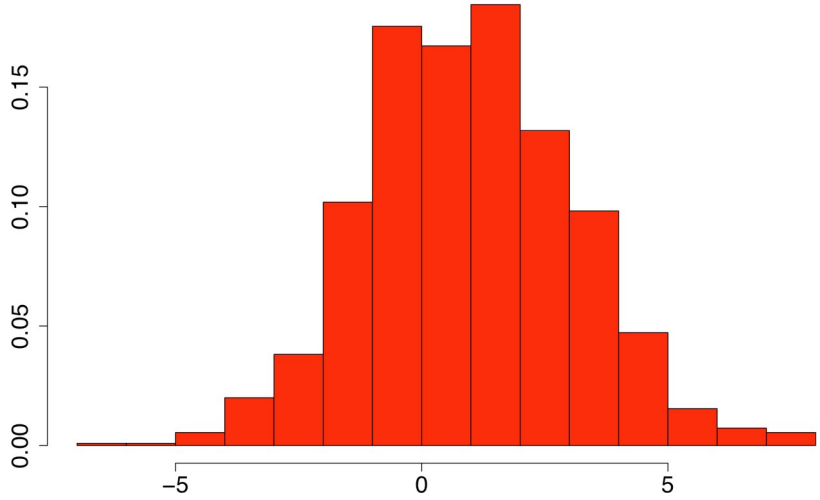




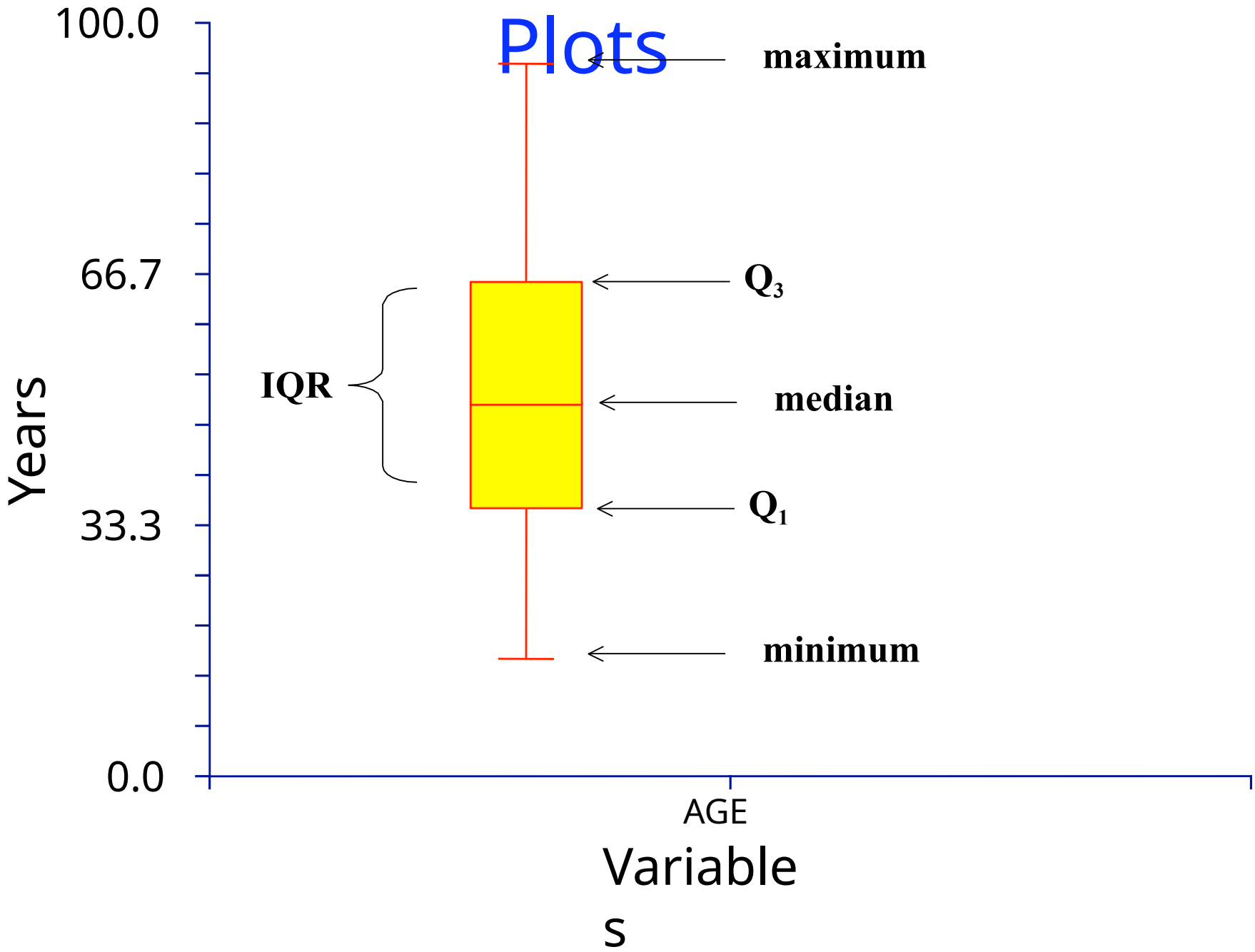
More on Histograms

- What's the difference between a frequency histogram and a density histogram?





Box Plots


100.0
66.7
33.3
0.0
Years
maximum
Q_3
IQR
median
Q_1
minimum
AGE
Variables

Bivariate Data

Variable 1	Variable 2	Display
Categorical	Categorical	Crosstabs Stacked Box Plot
Categorical	Continuous	Boxplot
Continuous	Continuous	Scatterplot Stacked Box Plot

Multivariate Data

Clustering

- Organize ***units*** into clusters
- Descriptive, not inferential
- Many approaches
- "Clusters" always produced

Data Reduction Approaches (PCA)

- Reduce n-dimensional dataset into much smaller number
- Finds a new (smaller) set of variables that retains most of the information in the total sample
- Effective way to visualize multivariate data

How to Make a Bad Graph

The aim of good data graphics:

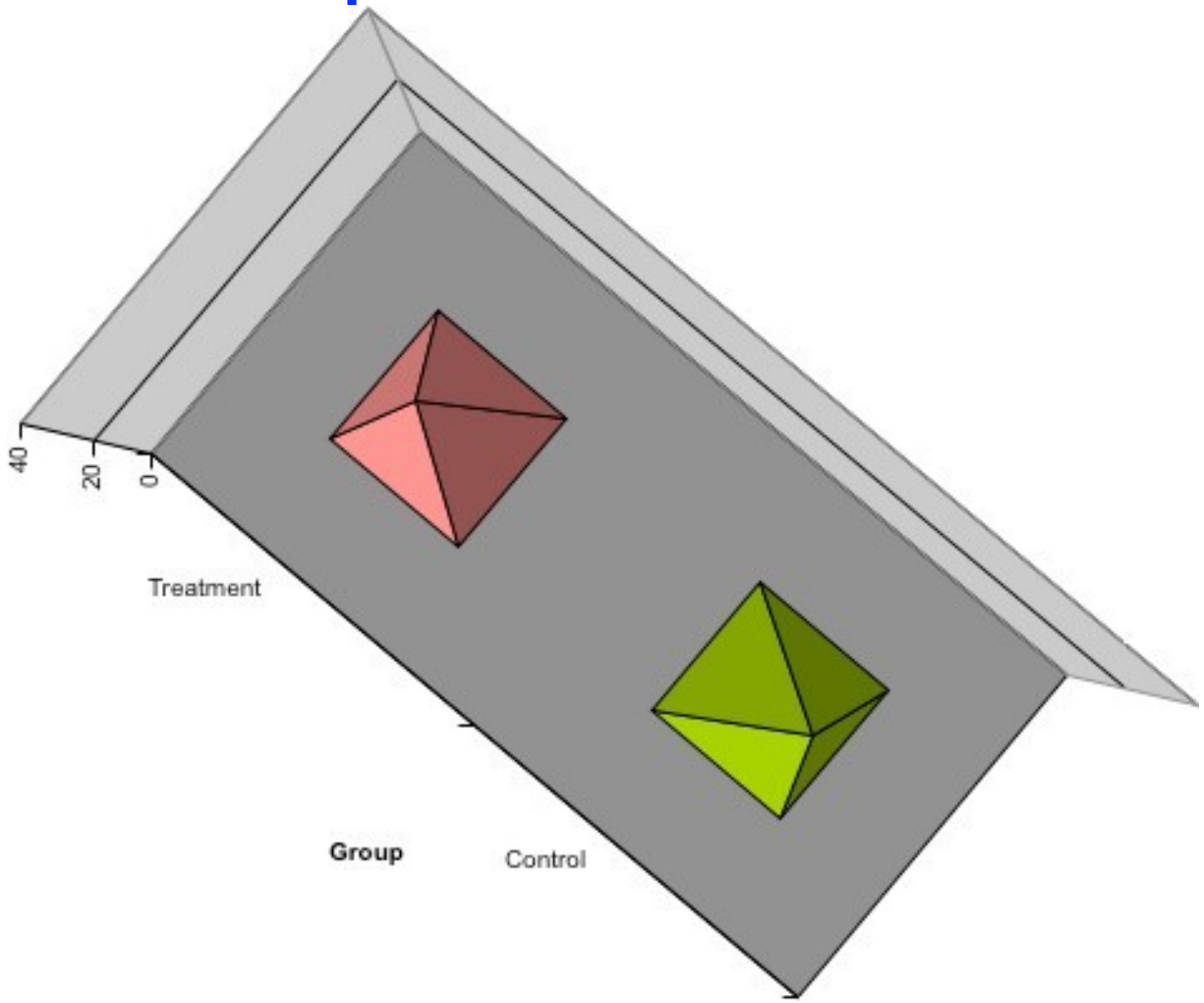
Display data accurately and clearly

Some rules for displaying data badly:

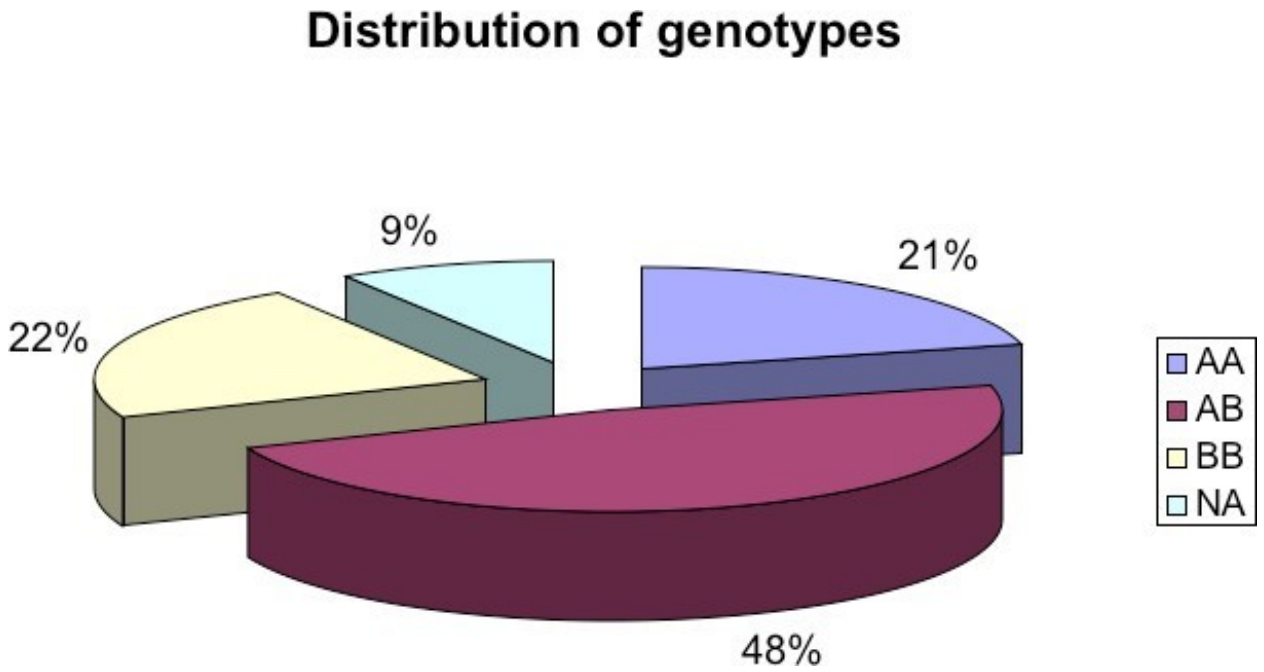
- Display as little information as possible
- Obscure what you do show (with chart junk)
- Use pseudo-3d and color gratuitously
- Make a pie chart (preferably in color and 3d)
- Use a poorly chosen scale

From Karl Broman:
http://www.biostat.wisc.edu/~kbroman/

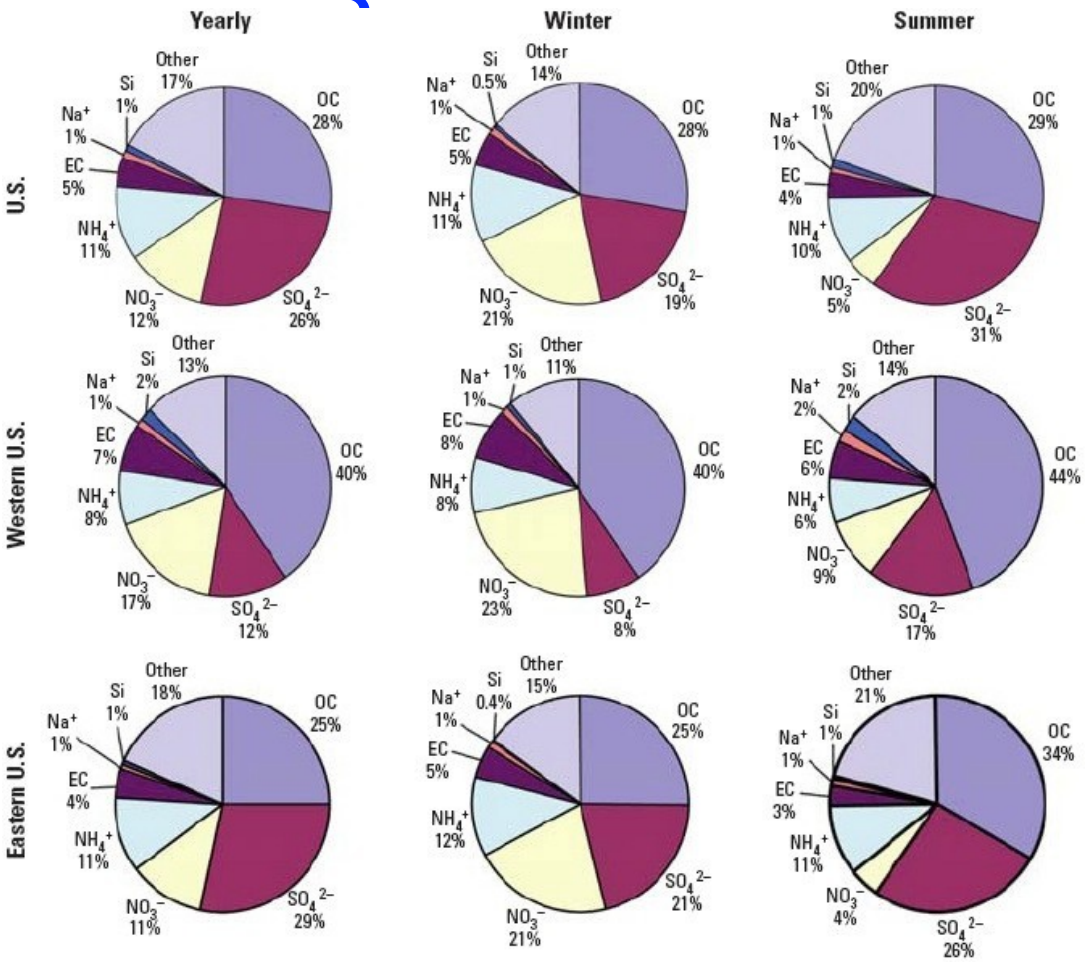
Example 1



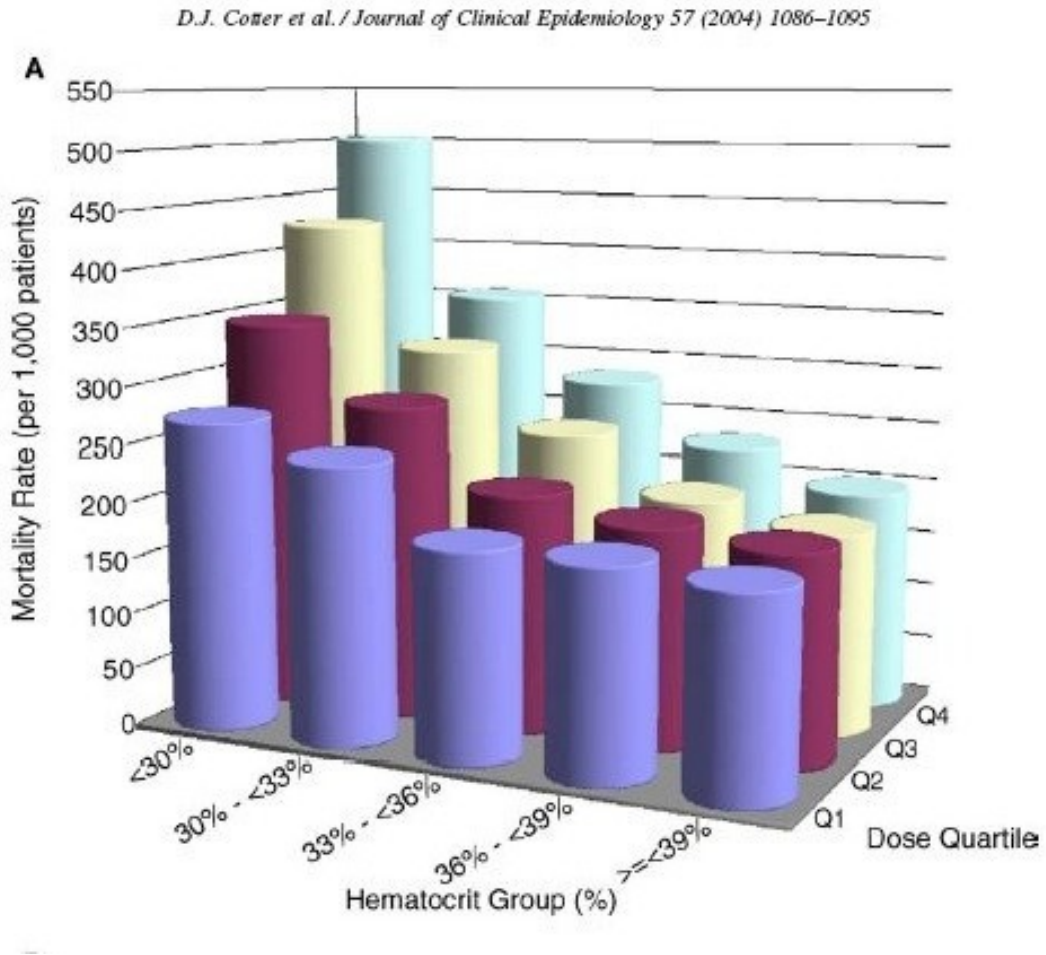
Example 2



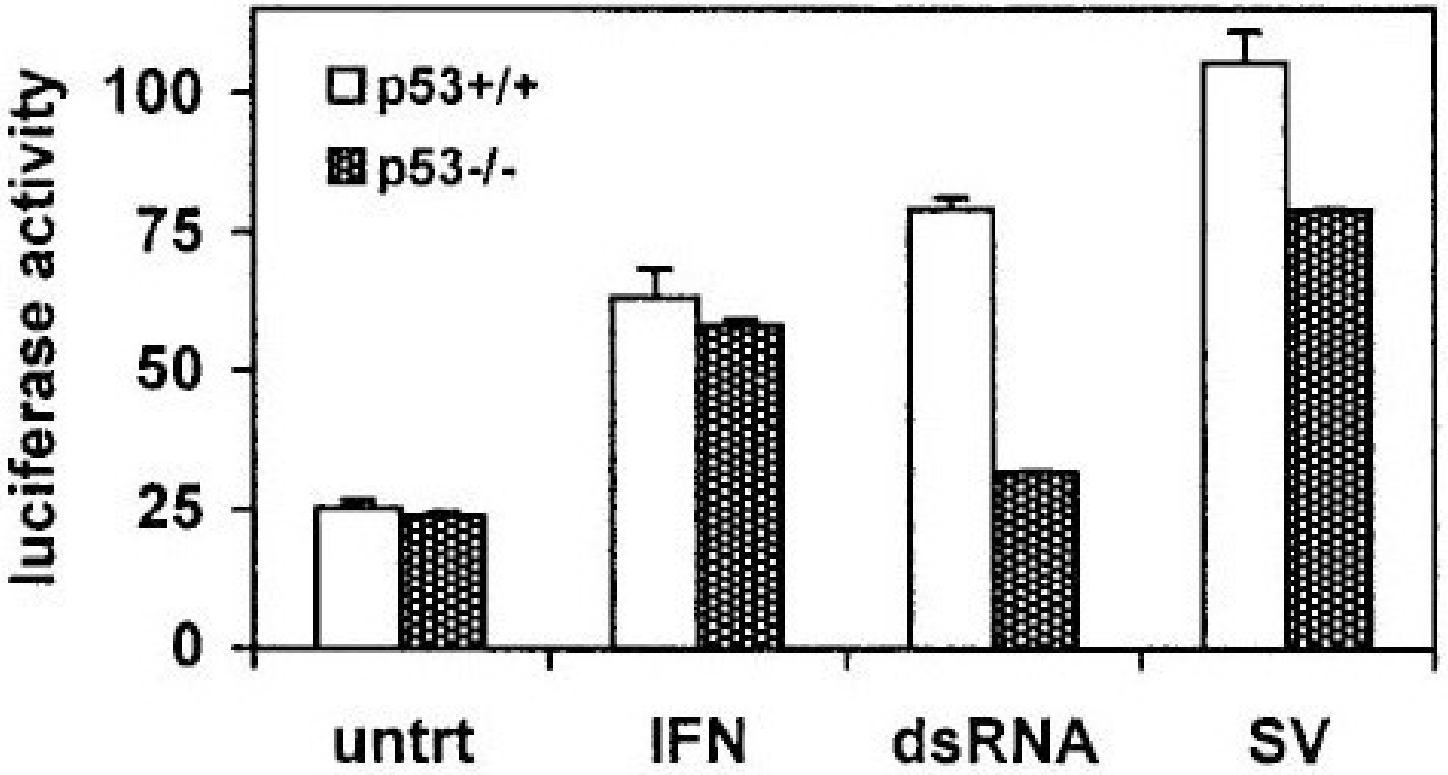
Example 3


Yearly
Winter
Summer
U.S.
Western U.S.
Eastern U.S.
Other 17%
Si 1%
Na+ 1%
EC 5%
NH4+ 11%
NO3− 12%
SO4 2− 26%
OC 28%
Other 14%
Si 0.5%
Na+ 1%
EC 5%
NH4+ 11%
NO3− 21%
SO4 2− 19%
OC 28%
Other 20%
Si 1%
Na+ 1%
EC 4%
NH4+ 10%
NO3− 5%
SO4 2− 31%
OC 29%
Other 13%
Si 2%
Na+ 1%
EC 7%
NH4+ 8%
NO3− 17%
SO4 2− 12%
OC 40%
Other 11%
Si 1%
Na+ 1%
EC 8%
NH4+ 8%
NO3− 23%
SO4 2− 8%
OC 40%
Other 14%
Si 2%
Na+ 2%
EC 6%
NH4+ 6%
NO3− 9%
SO4 2− 17%
OC 44%
Other 18%
Si 1%
Na+ 1%
EC 4%
NH4+ 11%
NO3− 11%
SO4 2− 29%
OC 25%
Other 15%
Si 0.4%
Na+ 1%
EC 5%
NH4+ 12%
NO3− 21%
SO4 2− 21%
OC 25%
Other 21%
Si 1%
Na+ 1%
EC 3%
NH4+ 11%
NO3− 4%
SO4 2− 26%
OC 34%

Example 4


D.J. Cotter et al. / Journal of Clinical Epidemiology 57 (2004) 1086–1095
A
Mortality Rate (per 1,000 patients)
550
500
450
400
350
300
250
200
150
100
50
0
<30%
30% - <33%
33% - <36%
36% - <39%
>=<39%
Hematocrit Group (%)
Q4
Q3
Q2
Q1
Dose Quartile

Example 5



R Tutorial

- Calculating descriptive statistics in R
- Useful R commands for working with multivariate data (apply and its derivatives)
- Creating graphs for different types of data (histograms, boxplots, scatterplots)
- Basic clustering and PCA analysis

Principles of Data Science

Unit – IV

Model Development

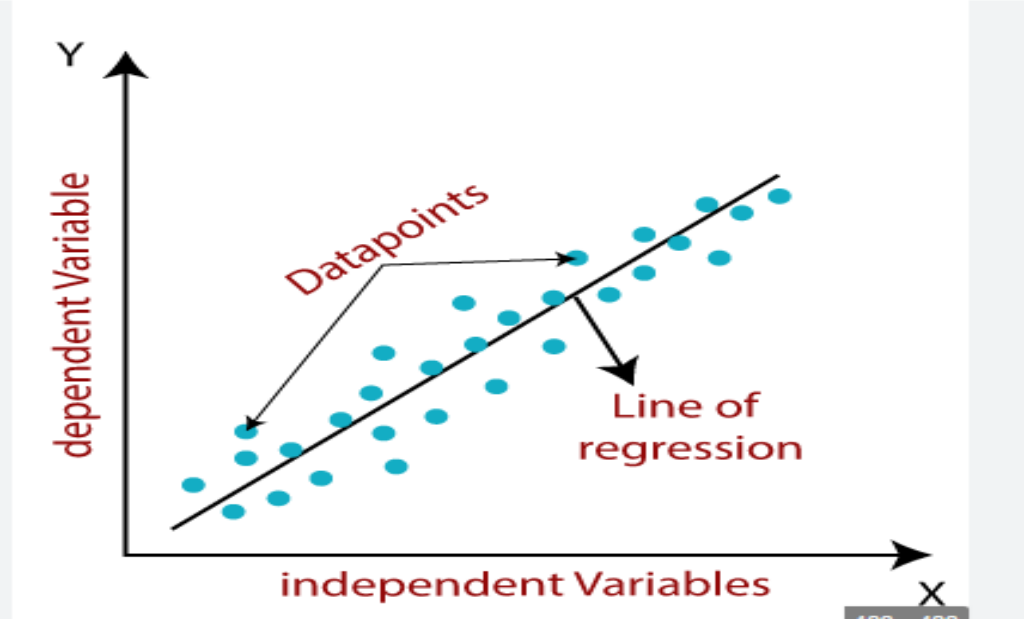
- Simple and Multiple Regression
- Model Evaluation using Visualization
- Residual Plot
- Distribution Plot
- Polynomial Regression and Pipelines
- Measures for In-sample Evaluation
- Prediction and Decision Making.

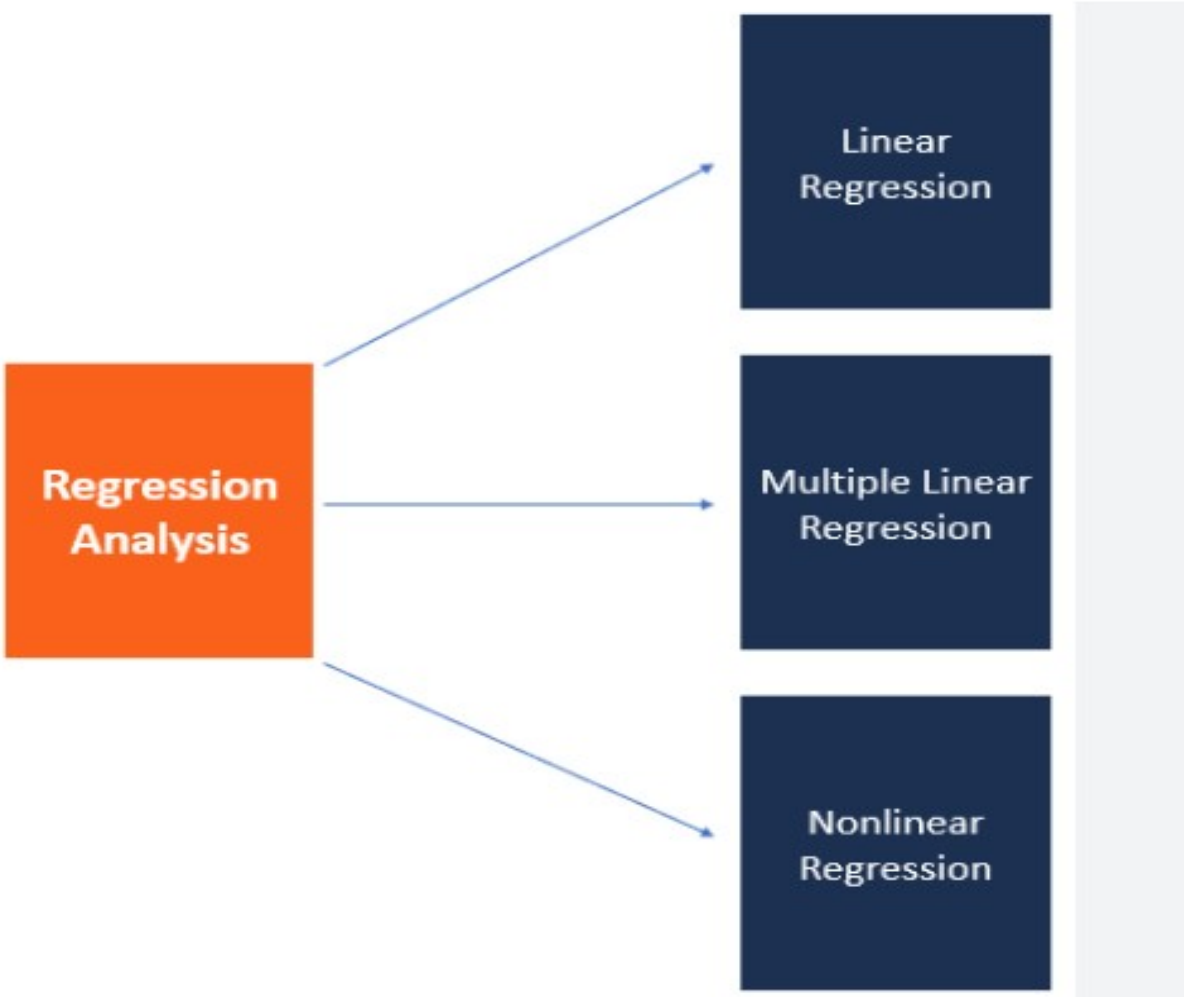
Model Development

- Model development is **an iterative process, in which many models are derived, tested and built upon until a model fitting the desired criteria is built**. Subsequent modeling work may need to begin the search at the same place as the original model building began, rather than where it finished.
- **In this section**, we will develop several models that will predict the price of the car using the variables or features.
- **This is just an estimate** but should give us an objective idea of how much the car should cost.
- **Some questions** we want to ask in this module
  - **Do I know** if the dealer is offering fair value for my trade-in?
  - **Do I know** if I put a fair value on my car?
- **Data Analytics**, we often use Model Development to help us predict future observations from the data we have.
- **A Model** will help us understand the exact relationship between different variables and how these variables are used to predict the result.
- **Load data** and store in data frame.

Regression in model development

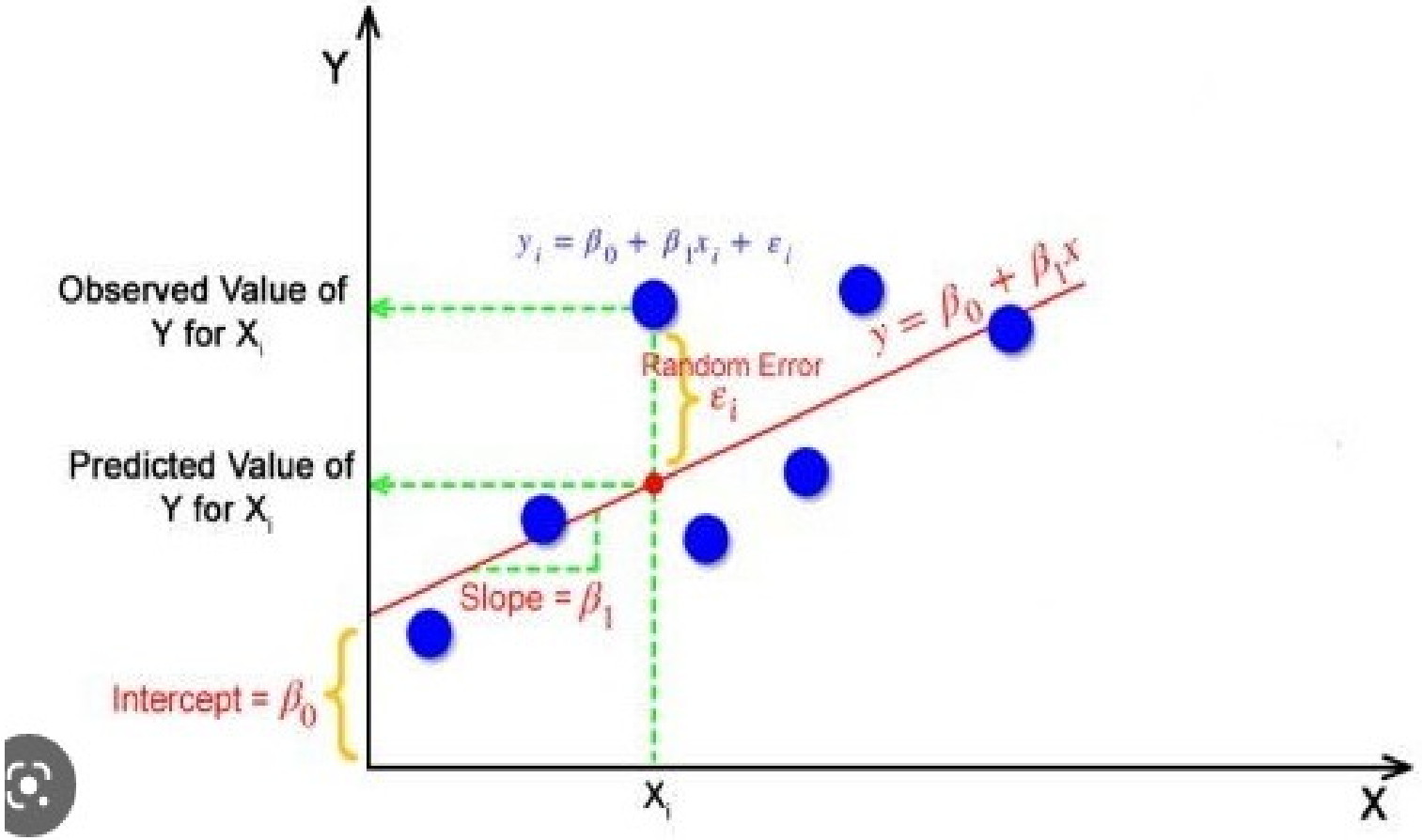
- Create Regression Model is **used to model the relationship between two or more explanatory variables and a response variable by fitting a linear equation to observed data**.
- Every value of the independent variable (x) is associated with a value of the dependent variable (y).




Regression Analysis
Linear Regression
Multiple Linear Regression
Nonlinear Regression

Linear Regression and Multiple Linear Regression

- **Linear Regression :** One example of a Data Model that we will be using is Simple Linear Regression.
- **Simple Linear Regression** is a method to help us understand the relationship between two variables:
  - **The predictor/independent variable** (X)
  - **The response/dependent variable** (that we want to predict)(Y)
- **The result of Linear Regression** is a linear function that predicts the response (dependent) variable as a function of the predictor (independent) variable.
  - Y**:**$Response\ Variable$
  - X**:**$Predictor\ Variables$
  - **Linear function:** $Yhat{=}a{+}bX$
- **a refers** to the intercept of the regression line0, in other words: the value of Y when X is 0
- **b refers** to the slope of the regression line, in other words: the value with which Y changes when X increases by 1 unit


Y
$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i$
Observed Value of
Y for X_i
Random Error
ε_i
$y = \beta_0 + \beta_1 x$
Predicted Value of
Y for X_i
Slope = β_1
Intercept = β_0
X_i
X

- **How could Highway-mpg help us predict car price?** For this example, we want to look at how highway-mpg can help us predict car price. Using simple linear regression, we will create a linear function with "highway-mpg" as the predictor variable and the "price" as the response variable.

  X = df[['highway-mpg']] Y = df['price']

- **Fit the linear model using highway-mpg.**

  lm.fit(X,Y)

  #We can output a prediction

  Yhat=lm.predict(X)

  Yhat[0:5]

- **What is the value of the intercept (a)?**

  lm.intercept_

- **What is the value of the Slope (b)?**

  lm.coef_

- **What is the final estimated linear model we get?** As we saw above, we should get a final linear model with the structure:

  $Yhat=a+bX$

  Plugging in the actual values we get:

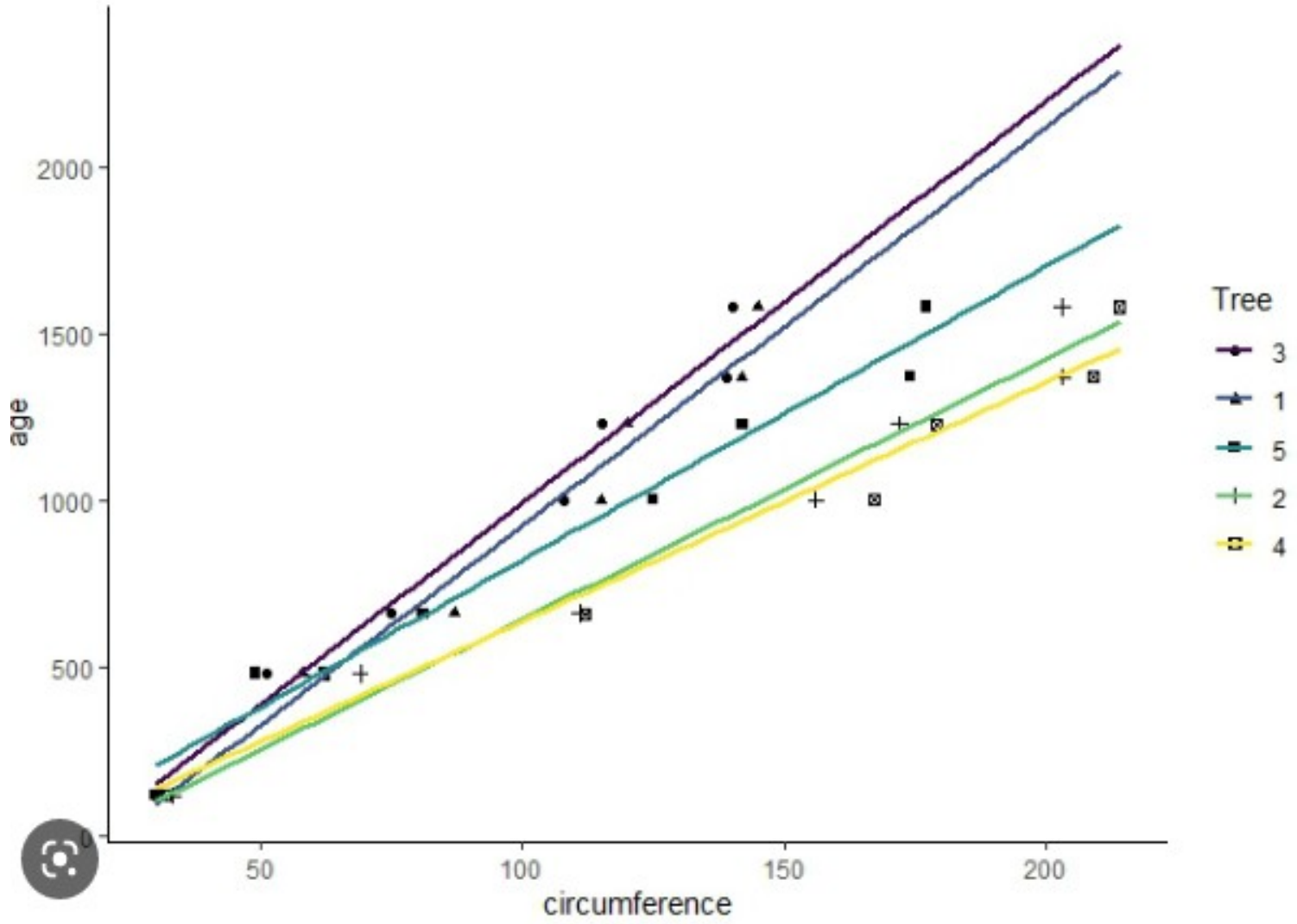
  - **price** = 38423.31 - 821.73 x highway-mpg

Multiple Linear Regression

- **What if we want to predict** car price using more than one variable?
- **If we want to use more variables** in our model to predict car price, we can use Multiple Linear Regression.
- **Multiple Linear Regression** is very similar to Simple Linear Regression, but this method is used to explain the relationship between one continuous response (dependent) variable and two or more predictor (independent) variables.
- **Most of the real-world regression models** involve multiple predictors. We will illustrate the structure by using four predictor variables, but these results can generalize to any integer:

    - Y:ResponseVariable
    - X1:PredictorVariable1
    - X2:PredictorVariable2
    - X3:PredictorVariable3
    - X4:PredictorVariable4
    - a:intercept
    - b1:coefficientsofVariable1
    - b2:coefficientsofVariable2
    - b3:coefficientsofVariable3
    - b4:coefficientsofVariable4

- **The equation is given by** $\hat{Y}=a+b1X1+b2X2+b3X3+b4X4$


2000
1500
1000
500
0
age
50
100
150
200
circumference
Tree
3
1
5
2
4

- From the previous section we know that other good predictors of price could be:
  - Horsepower
  - Curb-weight
  - Engine-size
  - Highway-mpg
- Let's develop a model using these variables as the predictor variables.

  Z = df[['horsepower', 'curb-weight', 'engine-size', 'highway-mpg']]
- Fit the linear model using the four above-mentioned variables.

  lm.fit(Z, df['price'])
- What is the value of the intercept(a)?

  lm.intercept_
- What are the values of the coefficients (b1, b2, b3, b4)?

  lm.coef_
- What is the final estimated linear model that we get? The plot obtained from above command, we should get a final linear function with the structure:

  Yhat=a+b1X1+b2X2+b3X3+b4X4
- What is the linear function we get in this example?

Price = -15678.742628061467 + 52.65851272 x horsepower + 4.69878948 x curb-weight + 81.95906216 x engine-size + 33.58258185 x highway-mpg

Model Evaluation using Visualization

- Now that we've developed some models, how do we evaluate our models and how do we choose the best one? One way to do this is by using visualization.
- import the visualization package: seaborn

```
# import the visualization package: seaborn
import seaborn as sns
%matplotlib inline
```

- Regression Plot When it comes to simple linear regression, an excellent way to visualize the fit of our model is by using regression plots.
- This plot will show a combination of a scattered data points (a scatter plot), as well as the fitted linear regression line going through the data.
- This will give us a reasonable estimate of the relationship between the two variables, the strength of the correlation, as well as the direction (positive or negative correlation).
- Let's visualize Horsepower as potential predictor variable of price:

```
width = 12
height = 10
plt.figure(figsize=(width, height))
sns.regplot(x="highway-mpg", y="price", data=df)
plt.ylim(0,)
```

- **We can see from this plot** that price is negatively correlated to highway-mpg, since the regression slope is negative.
- **One thing** to keep in mind when looking at a regression plot is to pay attention to how scattered the data points are around the regression line.
- **This will give you a good indication** of the variance of the data, and whether a linear model would be the best fit or not.
- **If the data is too far off** from the line, this linear model might not be the best model for this data.
- **Let's compare this plot** to the regression plot of "peak-rpm".

```
plt.figure(figsize=(width, height))
sns.regplot(x="peak-rpm", y="price", data=df)
plt.ylim(0,)
```

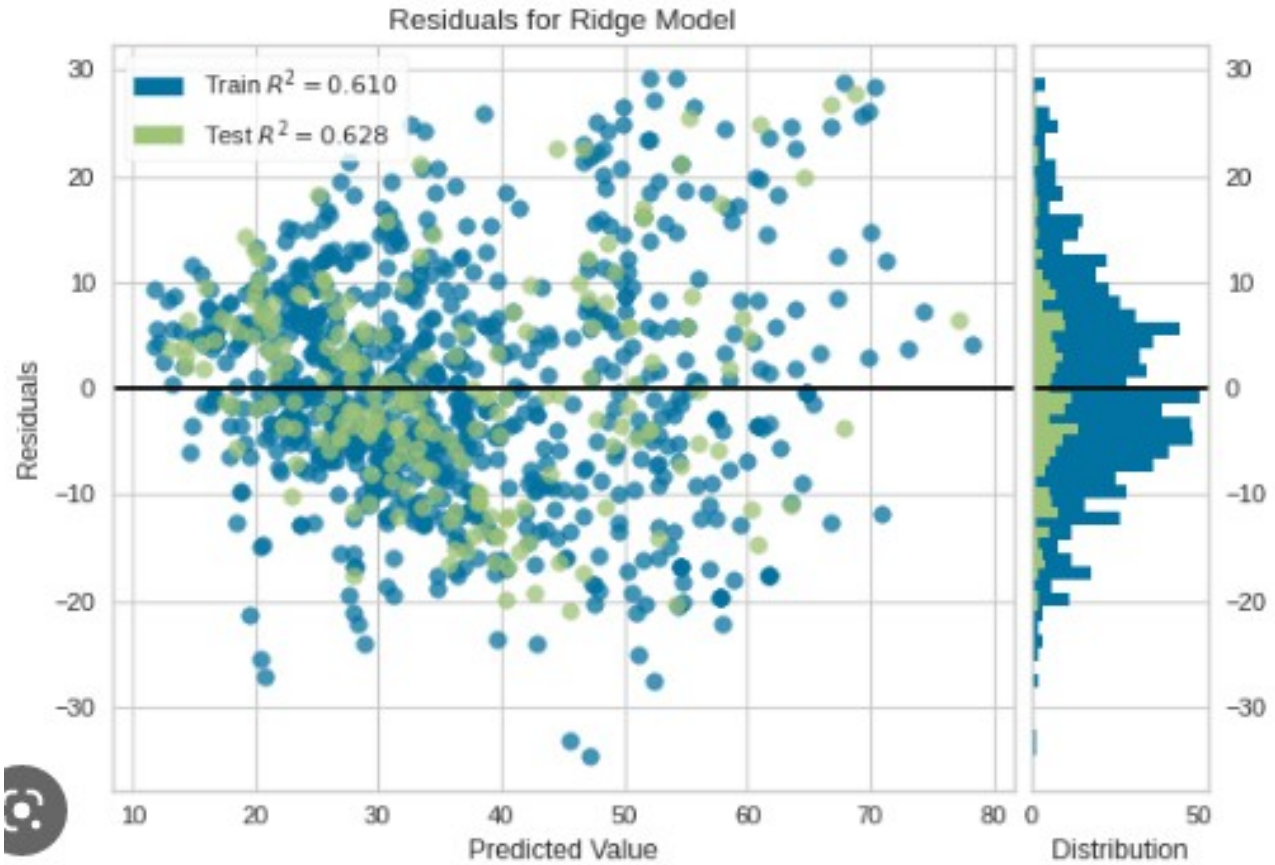
- **Comparing the regression plot** of "peak-rpm" and "highway-mpg" we see that the points for "highway-mpg" are much closer to the generated line and on the average decrease.
- **The points for "peak-rpm"** have more spread around the predicted line, and it is much harder to determine if the points are decreasing or increasing as the "highway-mpg" increases.

Residual Plot

- **A good way** to visualize the variance of the data is to use a residual plot.
- **What is a residual?** The difference between the observed value (y) and the predicted value (Yhat) is called the residual (e). When we look at a regression plot, the residual is the distance from the data point to the fitted regression line.
- **So what is a residual plot?** A residual plot is a graph that shows the residuals on the vertical y-axis and the independent variable on the horizontal x-axis.
- **What do we pay attention to when looking at a residual plot?** We look at the spread of the residuals: - If the points in a residual plot are randomly spread out around the x-axis, then a linear model is appropriate for the data. Why is that? Randomly spread out residuals means that the variance is constant, and thus the linear model is a good fit for this data.

```
width = 12
height = 10
plt.figure(figsize=(width, height))
sns.residplot(df['highway-mpg'], df['price'])
plt.show()
```

- **What is this plot telling us?** We can see from this residual plot that the residuals are not randomly spread around the x-axis, which leads us to believe that maybe a non-linear model is more appropriate for this data.


Residuals for Ridge Model
Train $R^2 = 0.610$
Test $R^2 = 0.628$
Residuals
Predicted Value
Distribution
30
20
10
0
−10
−20
−30
10
20
30
40
50
60
70
80
0
50

Residual Plot - Multiple Linear Regression

- **How do we visualize a model** for Multiple Linear Regression? This gets a bit more complicated because you can't visualize it with regression or residual plot.
- **One way to look at the fit of the model** is by looking at the distribution plot: We can look at the distribution of the fitted values that result from the model and compare it to the distribution of the actual values.
- **First** lets make a prediction

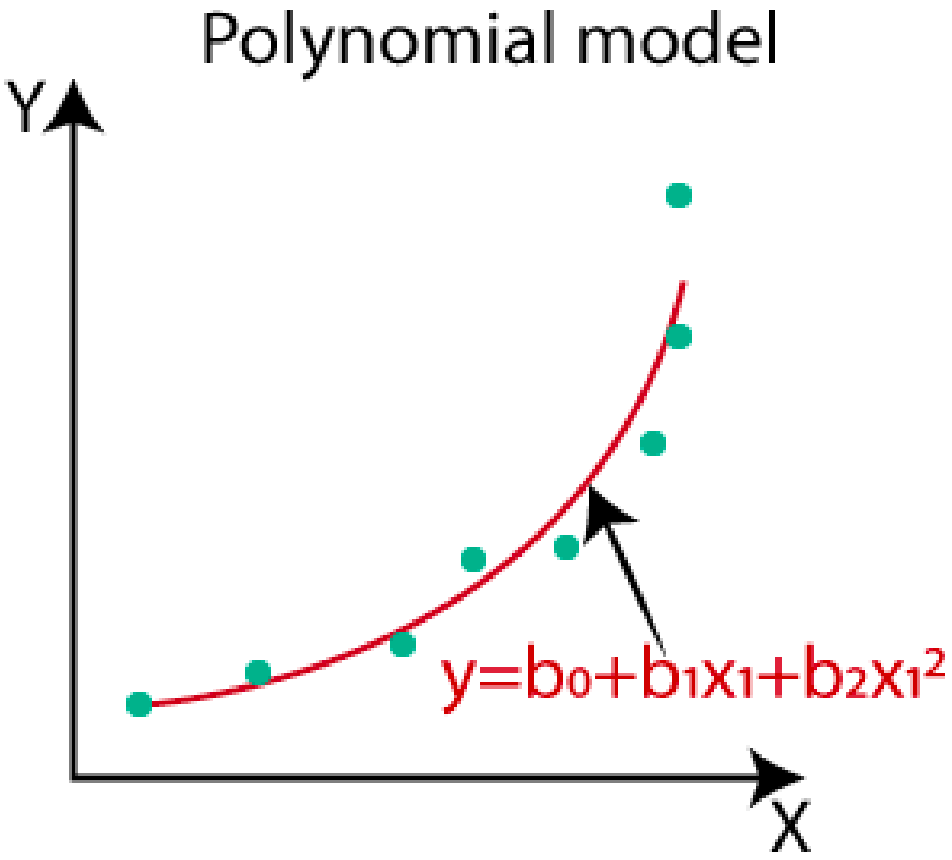
```
Y_hat = lm.predict(Z)
plt.figure(figsize=(width, height))
ax1 = sns.distplot(df['price'], hist=False, color="r", label="Actual Value")
sns.distplot(Yhat, hist=False, color="b", label="Fitted Values" , ax=ax1)
plt.title('Actual vs Fitted Values for Price')
plt.xlabel('Price (in dollars)')
plt.ylabel('Proportion of Cars')
plt.show()
plt.close()
```

- **We can see from plot obtained after above command that the fitted values** are reasonably close to the actual values, since the two distributions overlap a bit. However, there is definitely some room for improvement.

Polynomial Regression and Pipelines

- **Polynomial regression** is a particular case of the general linear regression model or multiple linear regression models.
- **We get non-linear relationships** by squaring or setting higher-order terms of the predictor variables.
- **There are different orders** of polynomial regression:
  - **Quadratic - 2nd order** Yhat=a+b1X2+b2X2
  - **Cubic - 3rd order** Yhat=a+b1X2+b2X2+b3X3
  - **Higher order:** Y=a+b1X2+b2X2+b3X3....
- **We saw earlier** that a linear model did not provide the best fit while using highway-mpg as the predictor variable. Let's see if we can try fitting a polynomial model to the data instead.
- **We will use the following function** to plot the data:

```
def PlotPolly(model, independent_variable, dependent_variabble, Name):
x_new = np.linspace(15, 55, 100) y_new = model(x_new)
 plt.plot(independent_variable, dependent_variabble, '.', x_new, y_new, '-')
 plt.title('Polynomial Fit with Matplotlib for Price ~ Length')
ax = plt.gca() ax.set_facecolor((0.898, 0.898, 0.898))
 fig = plt.gcf() plt.xlabel(Name) plt.ylabel('Price of Cars') plt.show() plt.close()
```


Polynomial model
Y
X
y=b0+b1x1+b2x1²

- **Lets get the variables**

  x = df['highway-mpg'] y = df['price']
- **Let's fit the polynomial** using the function polyfit, then use the function poly1d to display the polynomial function.

```
# Here we use a polynomial of the 3rd order (cubic)
f = np.polyfit(x, y, 3)
p = np.poly1d(f)
print(p)
```
- **Let's plot the function**

```
PlotPolly(p, x, y, 'highway-mpg')
np.polyfit(x, y, 3)
```
- **We can already** see from plotting that this polynomial model performs better than the linear model. This is because the generated polynomial function "hits" more of the data points.
- **The analytical expression** for Multivariate Polynomial function gets complicated. For example, the expression for a second-order (degree=2)polynomial with two variables is given by:

$$Yhat=a+b1X1+b2X2+b3X1X2+b4X21+b5X22$$
- **We can perform a polynomial transform** on multiple features. First, we import the module:

  from sklearn.preprocessing import PolynomialFeatures
- **We create a PolynomialFeatures** object of degree 2:

```
pr=PolynomialFeatures(degree=2)
pr
```
- **The original data** is of 201 samples and 4 features

  Z.shape
- **After the transformation**, there 201 samples and 15 features

  Z_pr.shape

Pipelines

- **Data Pipelines** simplify the steps of processing the data. We use the module Pipeline to create a pipeline. We also use StandardScaler as a step in our pipeline.

  from sklearn.pipeline import Pipeline

  from sklearn.preprocessing import StandardScaler

- **We create the pipeline**, by creating a list of tuples including the name of the model or estimator and its corresponding constructor.

  Input=[('scale',StandardScaler()), ('polynomial', PolynomialFeatures(include_bias=False)), ('model',LinearRegression())]

- **We input the list** as an argument to the pipeline constructor
- **We can normalize the data**, perform a transform and fit the model simultaneously.

  pipe.fit(Z,y)

- **Similarly**, we can normalize the data, perform a transform and produce a prediction simultaneously

  ypipe=pipe.predict(Z) ypipe[0:4]

Measures for In-Sample Evaluation

- **When evaluating our models**, not only do we want to visualize the results, but we also want a quantitative measure to determine how accurate the model is.
- **Two very important measures** that are often used in Statistics to determine the accuracy of a model are:
  - **R^2 / R-squared**
  - **Mean Squared Error (MSE)**
  - **R-squared**
- **R squared**, also known as the coefficient of determination, is a measure to indicate how close the data is to the fitted regression line.
- **The value of the R-squared** is the percentage of variation of the response variable (y) that is explained by a linear model.
- **Mean Squared Error (MSE)** The Mean Squared Error measures the average of the squares of errors, that is, the difference between actual value (y) and the estimated value ($\hat{y}$).

Model 1: Simple Linear Regression

Let's calculate the R^2

```
#highway_mpg_fit
lm.fit(X, Y)
# Find the R^2
print('The R-square is: ', lm.score(X, Y))
```

- **The R-square is:** 0.496591884339175 We can say that ~ 49.659% of the variation of the price is explained by this simple linear model "horsepower_fit".
- **Let's calculate the MSE** We can predict the output i.e., "yhat" using the predict method, where X is the input variable:

```
Yhat=lm.predict(X)
print('The output of the first four predicted value is: ', Yhat[0:4])
```

- **The output** of the first four predicted value is: [16236.50464347 16236.50464347 17058.23802179 13771.3045085]
- **Lets import** the function mean_squared_error from the module metrics

```
from sklearn.metrics import mean_squared_error
```

- **We compare** the predicted results with the actual results

```
mse = mean_squared_error(df['price'], Yhat)
print('The mean square error of price and predicted value is: ', mse)
```

- **The mean square error** of price and predicted value is: 31635042.944639895

Model 2: Multiple Linear Regression

- Let's calculate the R^2

```
# fit the model
lm.fit(Z, df['price'])
# Find the R^2
print('The R-square is: ', lm.score(Z, df['price']))
```

- **The R-square is:** 0.8093562806577458 We can say that ~ 80.896 % of the variation of price is explained by this multiple linear regression "multi_fit".
- **Let's calculate the MSE -** we produce a prediction

```
Y_predict_multifit = lm.predict(Z)
```

- **We compare** the predicted results with the actual results

```
print('The mean square error of price and predicted value using multifit is: ', \
mean_squared_error(df['price'], Y_predict_multifit))
```

- **The mean square error** of price and predicted value using multifit is: 11980366.870726489

Model 3: Polynomial Fit

- Let's calculate the R^2: let’s import the function r2_score from the module metrics as we are using a different function

```
from sklearn.metrics import r2_score
```

- **We apply** the function to get the value of r^2

```
r_squared = r2_score(y, p(x))
print('The R-square value is: ', r_squared)
```

- **We can say that ~ 67.419 %** of the variation of price is explained by this polynomial fit
- **MSE -** We can also calculate the MSE:

```
mean_squared_error(df['price'], p(x))
```

Prediction and Decision Making

- **In the previous section,** we trained the model using the method fit. Now we will use the method predict to produce a prediction.
- **Lets import pyplot** for plotting; we will also be using some functions from numpy.

```
import matplotlib.pyplot as plt
import numpy as np
%matplotlib inline
#Create a new input
    new_input=np.arange(1, 100, 1).reshape(-1, 1)
#Fit the model
     lm.fit(X, Y)
     lm
#Produce a prediction
    yhat=lm.predict(new_input)
    yhat[0:5]
#we can plot the data
    plt.plot(new_input, yhat)
    plt.show()
```

- **decision Making: Determining a Good Model Fit** Now that we have visualized the different models, and generated the R-squared and MSE values for the fits, how do we determine a good model fit?
- **What is a good R-squared value?** When comparing models, the model with the higher R-squared value is a better fit for the data.
- **What is a good MSE?** When comparing models, the model with the smallest MSE value is a better fit for the data

Let's take a look at the values for the different models.

```
Simple Linear Regression: Using Highway-mpg as a Predictor Variable of Price.

R-squared: 0.49659118843391759
MSE: 3.16 x10^7
```

```
Multiple Linear Regression: Using Horsepower, Curb-weight, Engine-size, and Highway-mpg as Predictor Variables of Price.

R-squared: 0.80896354913783497
MSE: 1.2 x10^7
```

```
Polynomial Fit: Using Highway-mpg as a Predictor Variable of Price.

R-squared: 0.6741946663906514
MSE: 2.05 x 10^7
```

Simple Linear Regression model (SLR) vs Multiple Linear Regression model (MLR)

- **Usually**, the more variables you have, the better your model is at predicting, but this is not always true. Sometimes you may not have enough data, you may run into numerical problems, or many of the variables may not be useful and or even act as noise. As a result, you should always check the MSE and R^2.
- **So to be able to compare the results of the MLR vs SLR models**, we look at a combination of both the R-squared and MSE to make the best conclusion about the fit of the model.
  - **MSE:** The MSE of SLR is 3.16x10^7 while MLR has an MSE of 1.2 x10^7. The MSE of MLR is much smaller.
  - **R-squared:** In this case, we can also see that there is a big difference between the R-squared of the SLR and the R-squared of the MLR. The R-squared for the SLR (~0.497) is very small compared to the R-squared for the MLR (~0.809).
- **This R-squared in combination** with the MSE show that MLR seems like the better model fit in this case, compared to SLR.

Simple Linear Model (SLR) vs Polynomial Fit

  - **MSE:** We can see that Polynomial Fit brought down the MSE, since this MSE is smaller than the one from the SLR.
  - **R-squared:** The R-squared for the Polyfit is larger than the R-squared for the SLR, so the Polynomial Fit also brought up the R-squared quite a bit.
- **Since the Polynomial Fit** resulted in a lower MSE and a higher R-squared, we can conclude that this was a better fit model than the simple linear regression for predicting Price with Highway-mpg as a predictor variable.

Multiple Linear Regression (MLR) vs Polynomial Fit

- **MSE:** The MSE for the MLR is smaller than the MSE for the Polynomial Fit.
- **R-squared:** The R-squared for the MLR is also much larger than for the Polynomial Fit.

- **Conclusion:** Comparing these three models, we conclude that the MLR model is the best model to be able to predict price from our dataset. This result makes sense, since we have 27 variables in total, and we know that more than one of those variables are potential predictors of the final car price.

Principles of Data Science

Unit – V

Model Evaluation

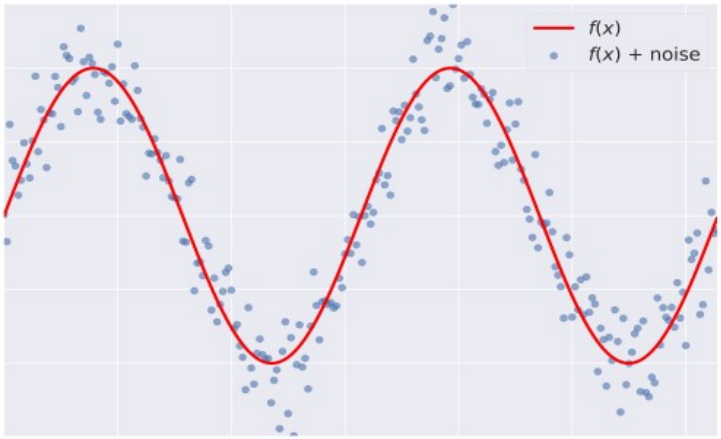
- Generalization Error
- Out-of-Sample Evaluation Metrics
- Cross Validation
- Overfitting
- Under Fitting and Model Selection
- Prediction by using Ridge Regression
- Testing Multiple Parameters by using Grid Search

Generalization Error

- For supervised learning applications in machine learning and statistical learning theory, **generalization error** (also known as the **out-of-sample error** or the **risk**) is a measure of how accurately an algorithm is able to predict outcome values for previously unseen data.
- Because learning algorithms are evaluated on finite samples, the evaluation of a learning algorithm may be sensitive to sampling error.
- As a result, measurements of prediction error on the current data may not provide much information about predictive ability on new data.
- Generalization error can be minimized by avoiding overfitting in the learning algorithm.
- The performance of a machine learning algorithm is visualized by plots that show values of *estimates* of the generalization error through the learning process, which are called learning curves.

Supervised Learning - Under the Hood

- Supervised Learning: y = f(x), f is unknown.



Goals of Supervised Learning

- Find a model $\hat{f}$ that best approximates f: $\hat{f} \approx f$
- $\hat{f}$ can be Logistic Regression, Decision Tree, Neural Network ...
- Discard noise as much as possible.
- **End goal**: $\hat{f}$ should acheive a low predictive error on unseen datasets.

Difficulties in Approximating f

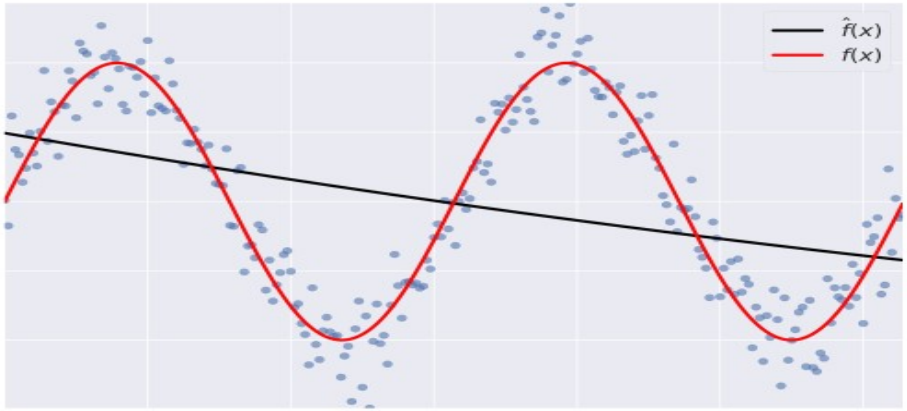
- **Overfitting:** $\hat{f}(x)$ fits the training set noise.
  - **Underfitting:** $\hat{f}$ is not flexible enough to approximate f.

Generalization Error

- **Generalization Error of** $\hat{f}$: Does $\hat{f}$ generalize well on unseen data?
- It can be decomposed as follows: Generalization Error of $\hat{f} = bias^2 + variance + \text{irreducible error}$

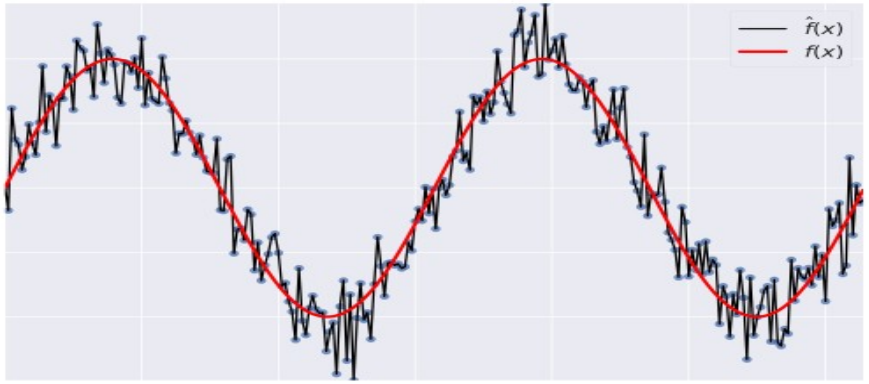
Bias

- **Bias**: error term that tells you, on average, how much $\hat{f} \neq f$.

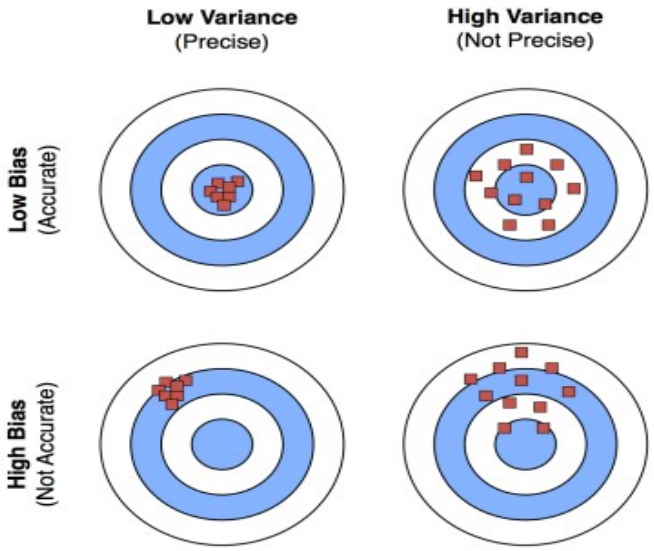


Variance

- **Variance**: tells you how much $\hat{f}$ is inconsistent over different training sets.



Bias-Variance Tradeoff: A Visual Explanation



Estimating the Generalization Error

Solution:

- split the data to training and test sets,
- fit $\hat{f}$ to the training set,
- evaluate the error of $\hat{f}$ on the **unseen** test set.
- generalization error of $\hat{f} \approx$ test set error of $\hat{f}$.

Evaluation Metrics

- The metrics used to evaluate the machine learning algorithms are very important. The choice of metrics to use influences how the performance of machine learning algorithms is measured and compared.
- The metrics influence both how you weight the importance of different characteristics in the results and your ultimate choice of algorithm.
- The main evaluation metrics for regression and classification are illustrated in Figure

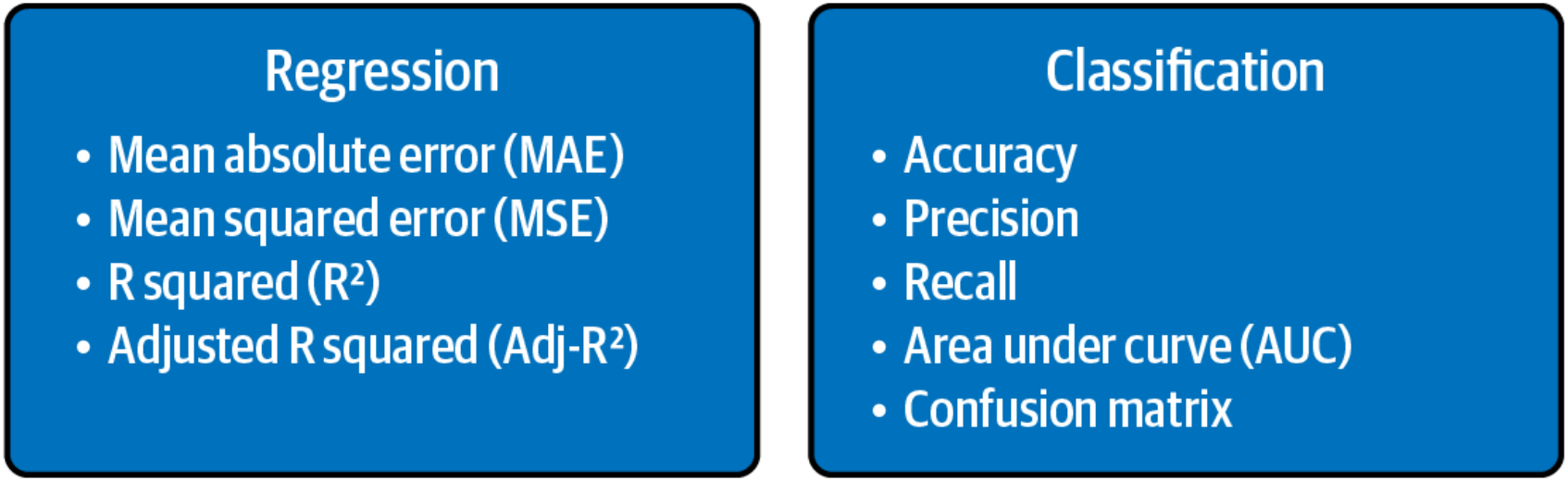


Figure 4-7. Evaluation metrics for regression and classification

Mean absolute error

- The *mean absolute error* (MAE) is the sum of the absolute differences between predictions and actual values. The MAE is a linear score, which means that all the individual differences are weighted equally in the average. It gives an idea of how wrong the predictions were. The measure gives an idea of the magnitude of the error, but no idea of the direction (e.g., over- or underpredicting).

Formula

$$\mathbf{MAE} = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n}$$

$\mathbf{MAE}$ = mean absolute error
y_i = prediction
x_i = true value
n = total number of data points

Mean squared error

- The *mean squared error* (MSE) represents the sample standard deviation of the differences between predicted values and observed values (called residuals). This is much like the mean absolute error in that it provides a gross idea of the magnitude of the error. Taking the square root of the mean squared error converts the units back to the original units of the output variable and can be meaningful for description and presentation. This is called the *root mean squared error* (RMSE).

Formula

$$\mathbf{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

$\mathbf{MSE}$ = mean squared error
n = number of data points
Y_i = observed values
$\hat{Y}_i$ = predicted values

R^2 metric

- The *R^2 metric* provides an indication of the "goodness of fit" of the predictions to actual value. In statistical literature this measure is called the coefficient of determination. This is a value between zero and one, for no-fit and perfect fit, respectively.

Formula

$$R^2 = 1 - \frac{RSS}{TSS}$$

R^2 = coefficient of determination
RSS = sum of squares of residuals
TSS = total sum of squares

Adjusted R^2 metric

- Just like *R^2*, *adjusted R^2* also shows how well terms fit a curve or line but adjusts for the number of terms in a model. It is given in the following formula:

  R adj 2 = 1 – (1–R 2)(n–1)) n–k–1

- where *n* is the total number of observations and *k* is the number of predictors. Adjusted *R^2* will always be less than or equal to *R^2*.

Selecting an evaluation metric for supervised regression

- In terms of a preference among these evaluation metrics, if the main goal is predictive accuracy, then RMSE is best. It is computationally simple and is easily differentiable. The loss is symmetric, but larger errors weigh more in the calculation. The MAEs are symmetric but do not weigh larger errors more. *R^2* and adjusted *R^2* are often used for explanatory purposes by indicating how well the selected independent variable(s) explains the variability in the dependent variable(s).

Classification

- For simplicity, we will mostly discuss things in terms of a binary classification problem (i.e., only two outcomes, such as true or false); some common terms are:
  1. True positives (TP)Predicted positive and are actually positive.
  2. False positives (FP)Predicted positive and are actually negative.
  3. True negatives (TN)Predicted negative and are actually negative.
  4. False negatives (FN)Predicted negative and are actually positive.
- The difference between three commonly used evaluation metrics for classification, accuracy, precision, and recall, is illustrated in Figure

$$\text{Precision} = \frac{\text{True positive}}{\text{Actual results}} \quad \text{or} \quad \frac{\text{True positive}}{\text{True positive + False positive}}$$

$$\text{Recall} = \frac{\text{True positive}}{\text{Predictive results}} \quad \text{or} \quad \frac{\text{True positive}}{\text{True positive + False negative}}$$

$$\text{Accuracy} = \frac{\text{True positive + True negative}}{\text{Total}}$$

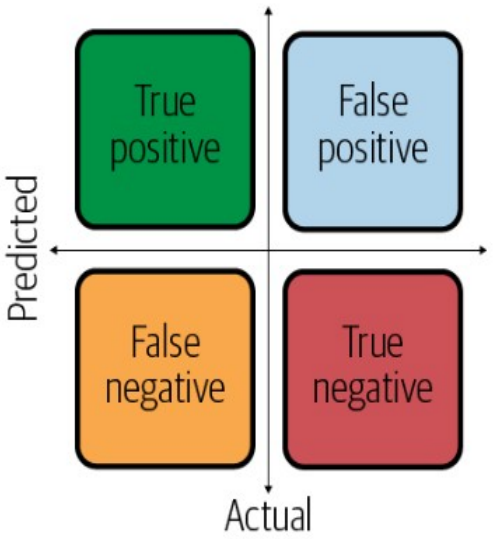


Figure 4-8. Accuracy, precision, and recall

Accuracy

- *Accuracy* is the number of correct predictions made as a ratio of all predictions made. This is the most common evaluation metric for classification problems and is also the most misused.
- It is most suitable when there are an equal number of observations in each class (which is rarely the case) and when all predictions and the related prediction errors are equally important, which is often not the case.

Precision

- *Precision* is the percentage of positive instances out of the total predicted positive instances. Here, the denominator is the model prediction done as positive from the whole given dataset.
- Precision is a good measure to determine when the cost of false positives is high (e.g., email spam detection).

Recall

- *Recall* (or *sensitivity* or *true positive rate*) is the percentage of positive instances out of the total actual positive instances. Therefore, the denominator (true positive + false negative) is the actual number of positive instances present in the dataset.
- Recall is a good measure when there is a high cost associated with false negatives (e.g., fraud detection).
- In addition to accuracy, precision, and recall, some of the other commonly used evaluation metrics for classification are discussed in the following sections.

Area under ROC curve

- *Area under ROC curve* (AUC) is an evaluation metric for binary classification problems. ROC is a probability curve, and AUC represents degree or measure of separability. It tells how much the model is capable of distinguishing between classes. The higher the AUC, the better the model is at predicting zeros as zeros and ones as ones. An AUC of *0.5* means that the model has no class separation capacity whatsoever. The probabilistic interpretation of the AUC score is that if you randomly choose a positive case and a negative case, the probability that the positive case outranks the negative case according to the classifier is given by the AUC.

Confusion matrix

- A confusion matrix lays out the performance of a learning algorithm. The confusion matrix is simply a square matrix that reports the counts of the true positive (TP), true negative (TN), false positive (FP), and false negative (FN) predictions of a classifier, as shown in Figure

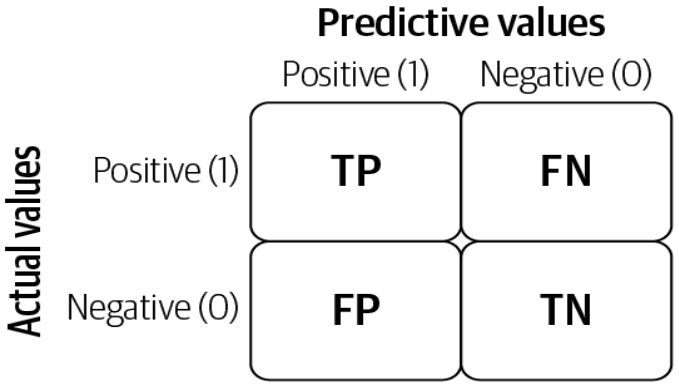


Figure 4-9. Confusion matrix

Cross Validation

- One of the challenges of machine learning is training models that are able to generalize well to unseen data (overfitting versus underfitting or a bias-variance trade-off).
- The main idea behind *cross validation* is to split the data one time or several times so that each split is used once as a validation set and the remainder is used as a training set: part of the data (the training sample) is used to train the algorithm, and the remaining part (the validation sample) is used for estimating the risk of the algorithm.
- Cross validation allows us to obtain reliable estimates of the model's generalization error.
- It is easiest to understand it with an example. When doing *k*-fold cross validation, we randomly split the training data into *k* folds. Then we train the model using *k-1* folds and evaluate the performance on the *k*th fold. We repeat this process *k* times and average the resulting scores.

Figure 4-6 shows an example of cross validation, where the data is split into five sets and in each round one of the sets is used for validation.

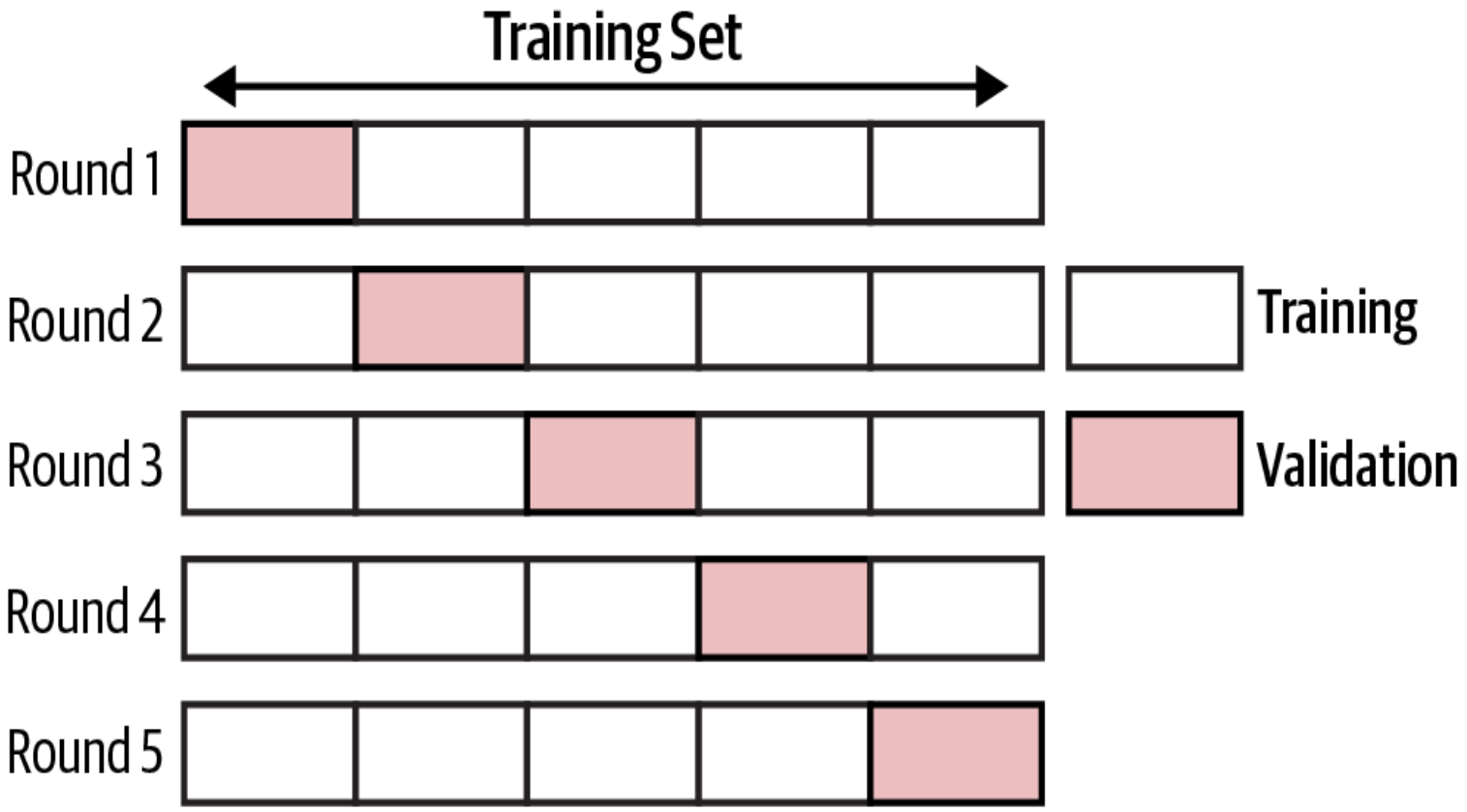


Figure 4-6. Cross validation

A potential drawback of cross validation is the computational cost, especially when paired with a grid search for hyperparameter tuning. Cross validation can be performed in a couple of lines using the sklearn package; we will perform cross validation in the supervised learning case studies.

Overfitting and Underfitting

- **Underfitting means that your model makes accurate, but initially incorrect predictions**. In this case, train error is large and val/test error is large too.
- Overfitting means that your model makes not accurate predictions. In this case, train error is very small and val/test error is large.
- The concepts of overfitting and underfitting are closely linked to *bias-variance trade-off*.
- *Bias* refers to the error due to overly simplistic assumptions or faulty assumptions in the learning algorithm. Bias results in underfitting of the data, as shown in the left-hand panel of Figure 4-5. A high bias means our learning algorithm is missing important trends among the features.
- *Variance* refers to the error due to an overly complex model that tries to fit the training data as closely as possible. In high variance cases, the model's predicted values are extremely close to the actual values from the training set. High variance gives rise to overfitting, as shown in the right-hand panel of Figure 4-5. Ultimately, in order to have a good model, we need low bias and low variance.

Figure 4-5. Overfitting and underfitting

There can be two ways to combat overfitting:

Using more training data

The more training data we have, the harder it is to overfit the data by learning too much from any single training example.

Using regularization

Adding a penalty in the loss function for building a model that assigns too much explanatory power to any one feature, or allows too many features to be taken into account.

Model Selection

- In the following section, we will go over all such factors, followed by a discussion of model trade-offs.

Factors for Model Selection

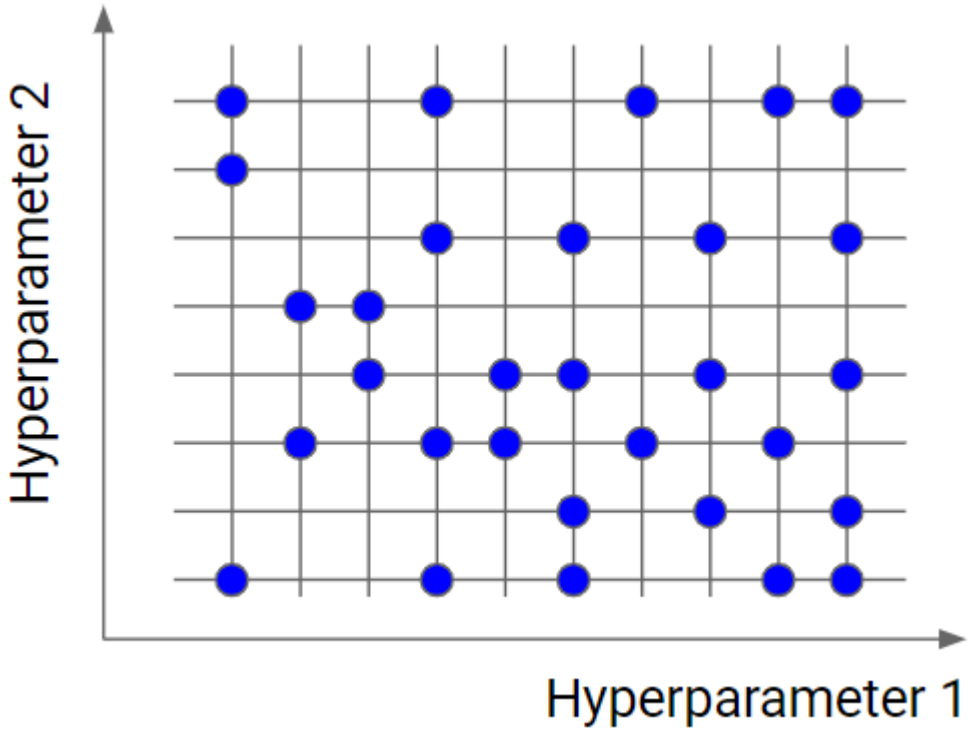
- The factors considered for the model selection process are as follows:
- **Simplicity:** The degree of simplicity of the model. Simplicity usually results in quicker, more scalable, and easier to understand models and results.
- **Training time:** Speed, performance, memory usage and overall time taken for model training.
- **Handle nonlinearity in the data:** The ability of the model to handle the nonlinear relationship between the variables.
- **Robustness to overfitting:** The ability of the model to handle overfitting.
- **Size of the dataset:** The ability of the model to handle large number of training examples in the dataset.
- **Number of features:** The ability of the model to handle high dimensionality of the feature space.
- **Model interpretation:** How explainable is the model? Model interpretability is important because it allows us to take concrete actions to solve the underlying problem.
- **Feature scaling:** Does the model require variables to be scaled or normally distributed?

The table is based on the advantages and disadvantages of different models discussed in the individual model section

	Linear regression	Logistic regression	SVM	CART	Gradient boosting	Random forest	Artificial neural network	KNN	LDA
Simplicity	✓	✓	✓	✓	✗	✗	✗	✓	✓
Training Time	✓	✓	✗	✓	✗	✗	✗	✓	✓
Handle non-linearity	✗	✗	✓	✓	✓	✓	✓	✓	✓
Robust to overfitting	✗	✗	✓	✗	✗	✓	✗	✓	✗
Large datasets	✗	✗	✗	✓	✓	✓	✓	✗	✓
Many features	✗	✗	✓	✓	✓	✓	✓	✗	✓
Model interpretation	✓	✓	✗	✓	✓	✓	✗	✓	✓
Feature scaling needed	✗	✗	✓	✗	✗	✗	✗	✗	✗

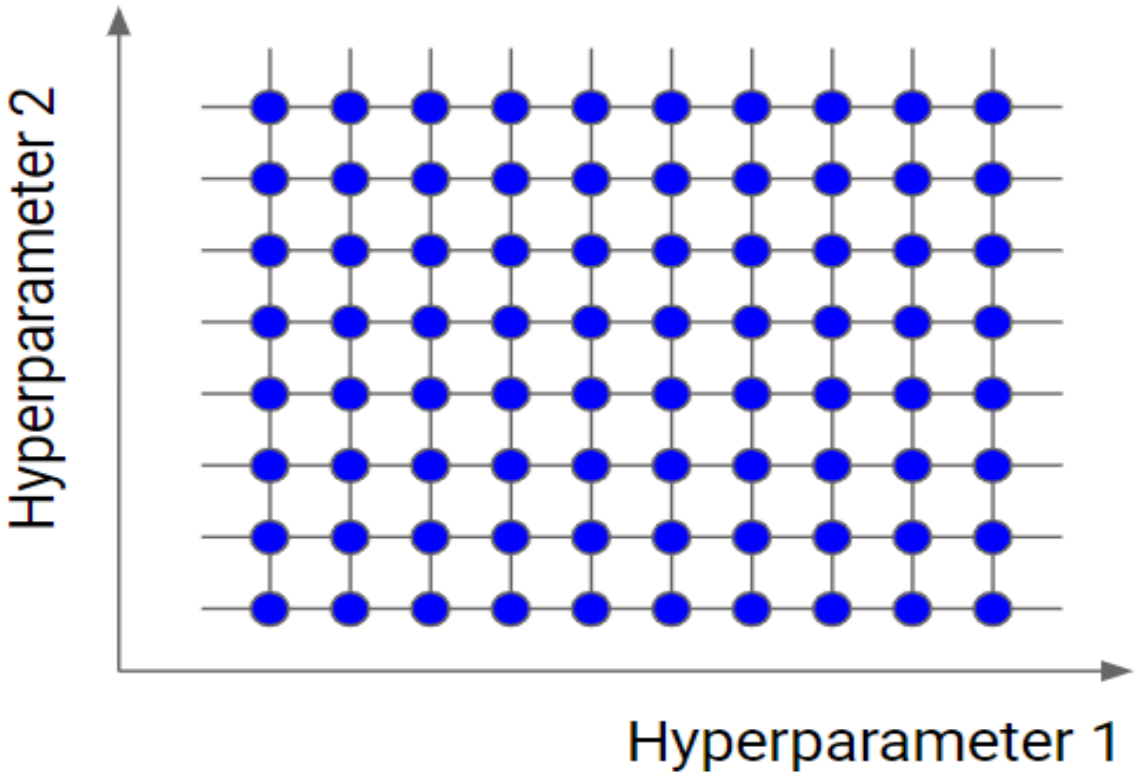
Testing Multiple Parameters by using Grid Search

- Hyperparameter tuning is one of the most important parts of a machine learning pipeline. A wrong choice of the hyperparameters' values may lead to wrong results and a model with poor performance.
- There are several ways to perform hyperparameter tuning. Two of them are grid search and random search



Grid search

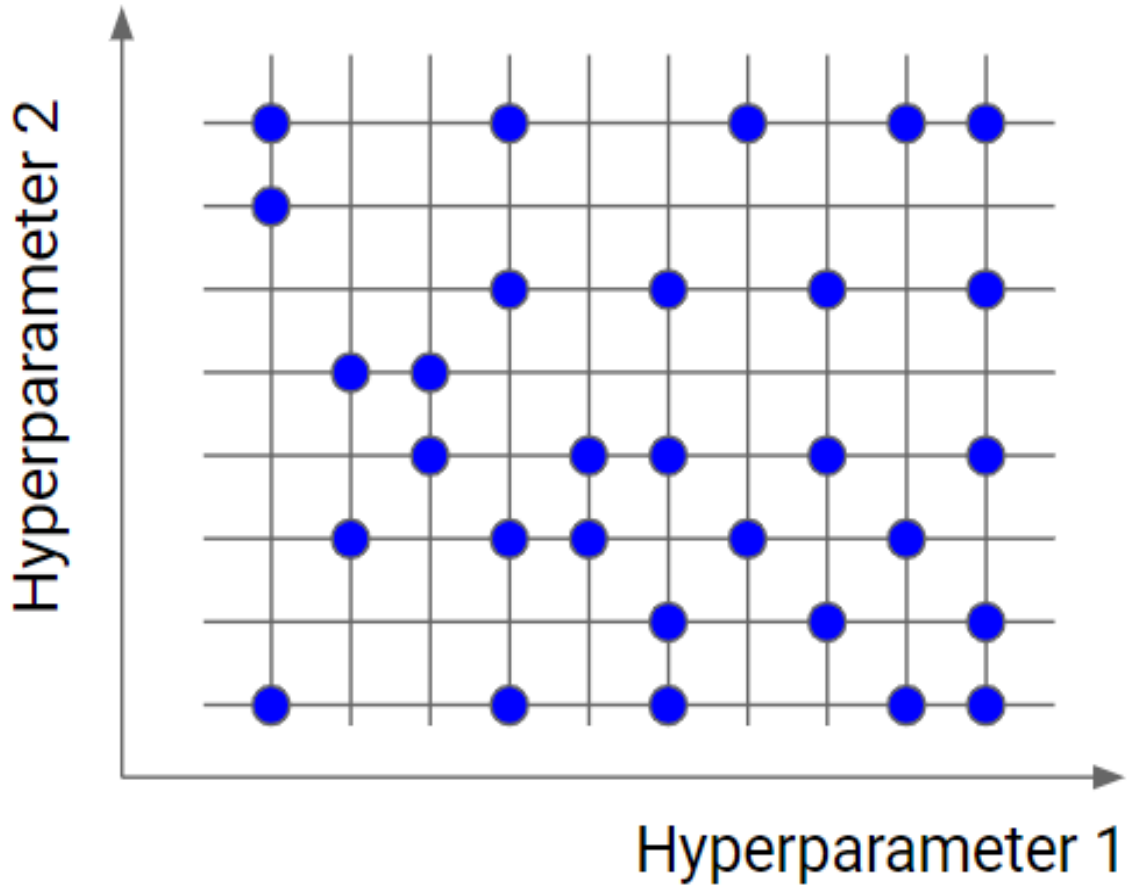
- Grid search is the simplest algorithm for hyperparameter tuning.
- Basically, we divide the domain of the hyperparameters into a discrete grid. Then, we try every combination of values of this grid, calculating some performance metrics using cross-validation.
- The point of the grid that maximizes the average value in cross-validation, is the optimal combination of values for the hyperparameters.
- Grid search is an exhaustive algorithm that spans all the combinations, so it can actually find the best point in the domain.
- The great drawback is that it's very slow.
- Checking every combination of the space requires a lot of time that, sometimes, is not available.
- Don't forget that every point in the grid needs k-fold cross-validation, which requires *k* training steps. So, tuning the hyperparameters of a model in this way can be quite complex and expensive.



Example of a grid search

Random search

- Random search is similar to grid search, but instead of using all the points in the grid, it tests only a randomly selected subset of these points.
- The smaller this subset, the faster but less accurate the optimization.
- The larger this dataset, the more accurate the optimization but the closer to a grid search.
- Random search is a very useful option when you have several hyperparameters with a fine-grained grid of values.
- Using a subset made by 5-100 randomly selected points, we are able to get a reasonably good set of values of the hyperparameters.
- It will not likely be the best point, but it can still be a good set of values that gives us a good model.



Example of random search

Principles of Data Science

Unit – VI

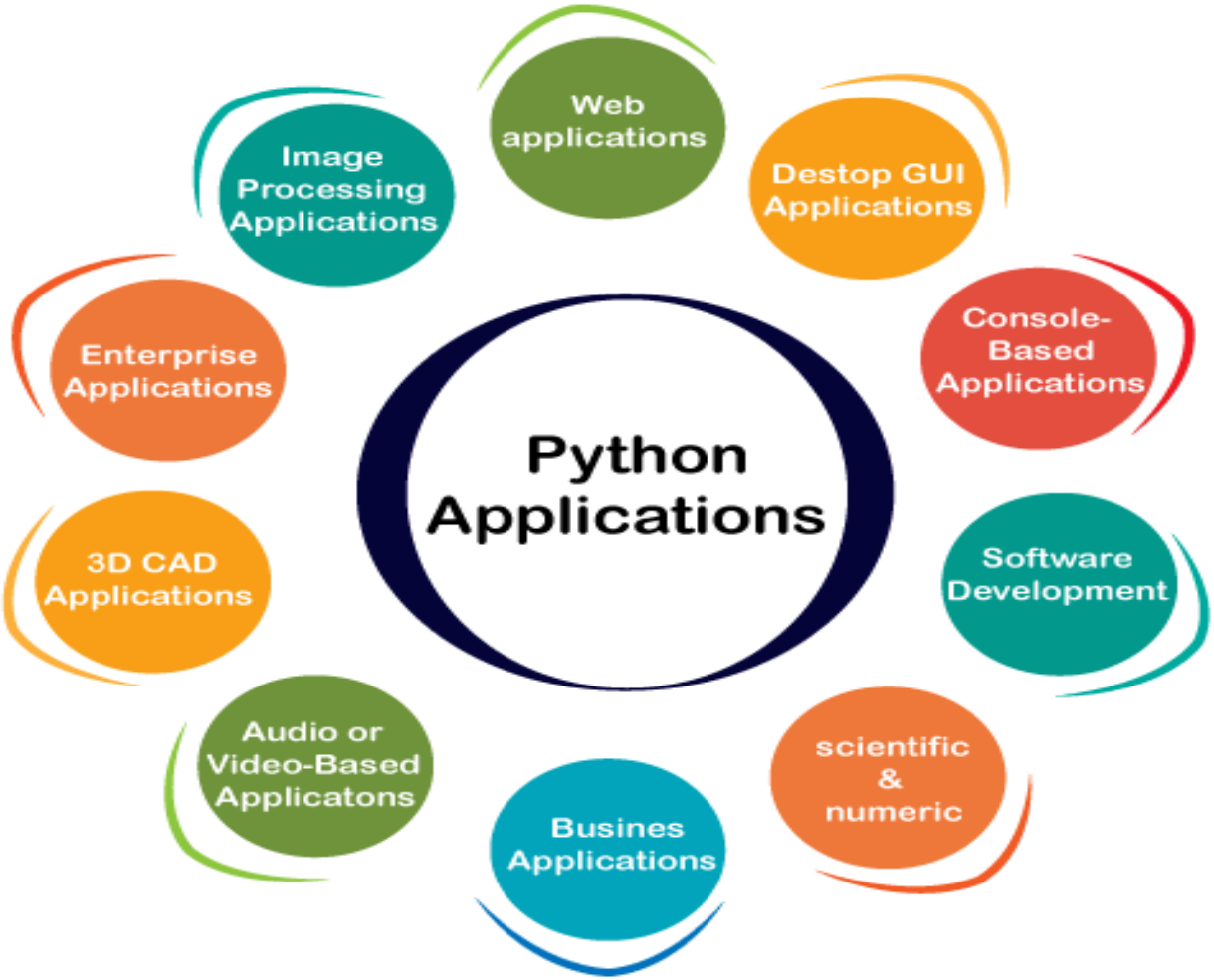
Data Analytics with Python

- Introduction to Python and its Basics
- Learn the basics of Python language
- Learn Regular Expressions in Python
- Learn Scientific libraries in Python
- NumPy, SciPy, Matplotlib and Pandas
- Effective Data Visualization
- Learn Scikit
- learn and Machine Learning.

Python

- Python was invented by **Guido van Rossum** in 1991 at CWI in Netherland. The idea of Python programming language has taken from the ABC programming language or we can say that ABC is a predecessor of Python language.
- **Python** is a general purpose, dynamic, **high-level,** and interpreted programming language. It supports Object Oriented programming approach to develop applications. It is simple and easy to learn and provides lots of high-level data structures.
- Python is ***easy to learn*** yet powerful and versatile scripting language, which makes it attractive for Application Development.
- Python's syntax and ***dynamic typing*** with its interpreted nature make it an ideal language for scripting and rapid application development.
- Python supports *multiple programming pattern*, including object-oriented, imperative, and functional or procedural programming styles.
- Python is not intended to work in a particular area, such as web programming. That is why it is known as *multipurpose* programming language because it can be used with web, enterprise, 3D CAD, etc.
- Python makes the development and debugging *fast* because there is no compilation step included in Python development, and edit-test-debug cycle is very fast.

Python Applications



Python Features

1) Easy to Learn and Use
2) Expressive Language
3) Interpreted Language
4) Cross-platform Language
5) Free and Open Source
6) Object-Oriented Language
7) Extensible
8) Large Standard Library
9) GUI Programming Support
10) Integrated
11. Embeddable
12. Dynamic Memory Allocation

Python Popular Frameworks and Libraries

- Python has wide range of libraries and frameworks widely used in various fields such as machine learning, artificial intelligence, web applications, etc. We define some popular frameworks and libraries of Python as follows.
  - **Web development (Server-side) -** Django Flask, Pyramid, CherryPy
  - **GUIs based applications -** Tk, PyGTK, PyQt, PyJs, etc.
  - **Machine Learning -** TensorFlow, PyTorch, **Scikit-learn**, Matplotlib, Scipy, etc.
  - **Mathematics -** Numpy, Pandas, etc.

Why Python is used by data scientists

- Python is not the only language used in data science and machine learning. R is another dominant option, and Java, JavaScript, and C++ also have their places. But Python's advantages have helped it earn its place as one of the most popular programming languages generally, and in data science and machine learning specifically.

These advantages include:

- Python is relatively **easy to learn**. Its syntax is concise and resembles English, which helps make learning it more intuitive.

- It has a large community of users. This translates into excellent **peer support and documentation**.

- Python is **portable** and allows you to run its code anywhere. This means a Python application can run across Windows, MacOS, and Linux without modifications to its source code (unless there are system-specific calls).

- Python is a **free, open-source, and object-oriented** programming language.

- Python makes it easy to **add modules from other languages**, such as C and C++.
- Finally, many of **Python's libraries** were literally made for data science and machine learning. We'll talk more about this advantage in the next section.

Scientific libraries in Python

Python's extensive collection of libraries enables all sorts of functionality, especially in data science and machine learning. Python has interactive libraries for data processing, data modeling, data manipulation, data visualization, machine learning algorithms, and more. Let's talk about seven of the top Python libraries for these fields.

1. NumPy

- **NumPy** is a popular open-source library for data processing and modeling that is widely used in data science, machine learning, and deep learning. It's also compatible with other libraries such as Pandas, Matplotlib, and Scikit-learn, which we'll discuss later.
- NumPy introduces objects for multidimensional arrays and matrices, along with routines that let you perform advanced mathematical and statistical functions on arrays with only a small amount of code. In addition, it contains some linear algebra functions and Fourier transforms.

2. SciPy

- **SciPy** is another open-source library for data processing and modeling that builds on NumPy for scientific computation applications. It contains more fully-featured versions of the linear algebra modules found in NumPy and many other numerical algorithms.
- SciPy provides algorithms for optimization, integration, interpolation, eigenvalue problems, algebraic equations, differential equations, statistics, and other classes of problems.
- It also adds a collection of algorithms and high-level commands for manipulating and visualizing data. For instance, by combining SciPy and NumPy, you can do things like image processing.

3. Pandas

- **Pandas** is an open-source package for data cleaning, processing, and manipulation. It provides extended, flexible data structures to hold different types of labeled and relational data.
- Pandas specializes in manipulating numerical tables and time series, which are common data forms in data science.
- Pandas is usually used along with other data science libraries: It's built on NumPy, and it's also used in SciPy for statistical analysis and Matplotlib for plotting functions.

4. Matplotlib

- **Matplotlib** is a data visualization and 2-D plotting library. In fact, it's considered the most popular and widely used plotting library in the Python community.
- Matplotlib stands out for its versatility. Matplotlib can be used in Python scripts, the Python and IPython shells, Jupyter notebooks, and web application servers. In addition, it offers a wide range of charts, including plots, bar charts, pie charts, histograms, scatterplots, error charts, power spectra, and stemplots.

5. Seaborn

- **Seaborn** is a data visualization library based on Matplotlib and closely integrated with NumPy and Pandas data structures. It provides a high-level interface for creating statistical graphics that assist greatly with exploring and understanding data.
- The data graphics available in Seaborn include bar charts, pie charts, histograms, scatterplots, and error charts.

6. TensorFlow

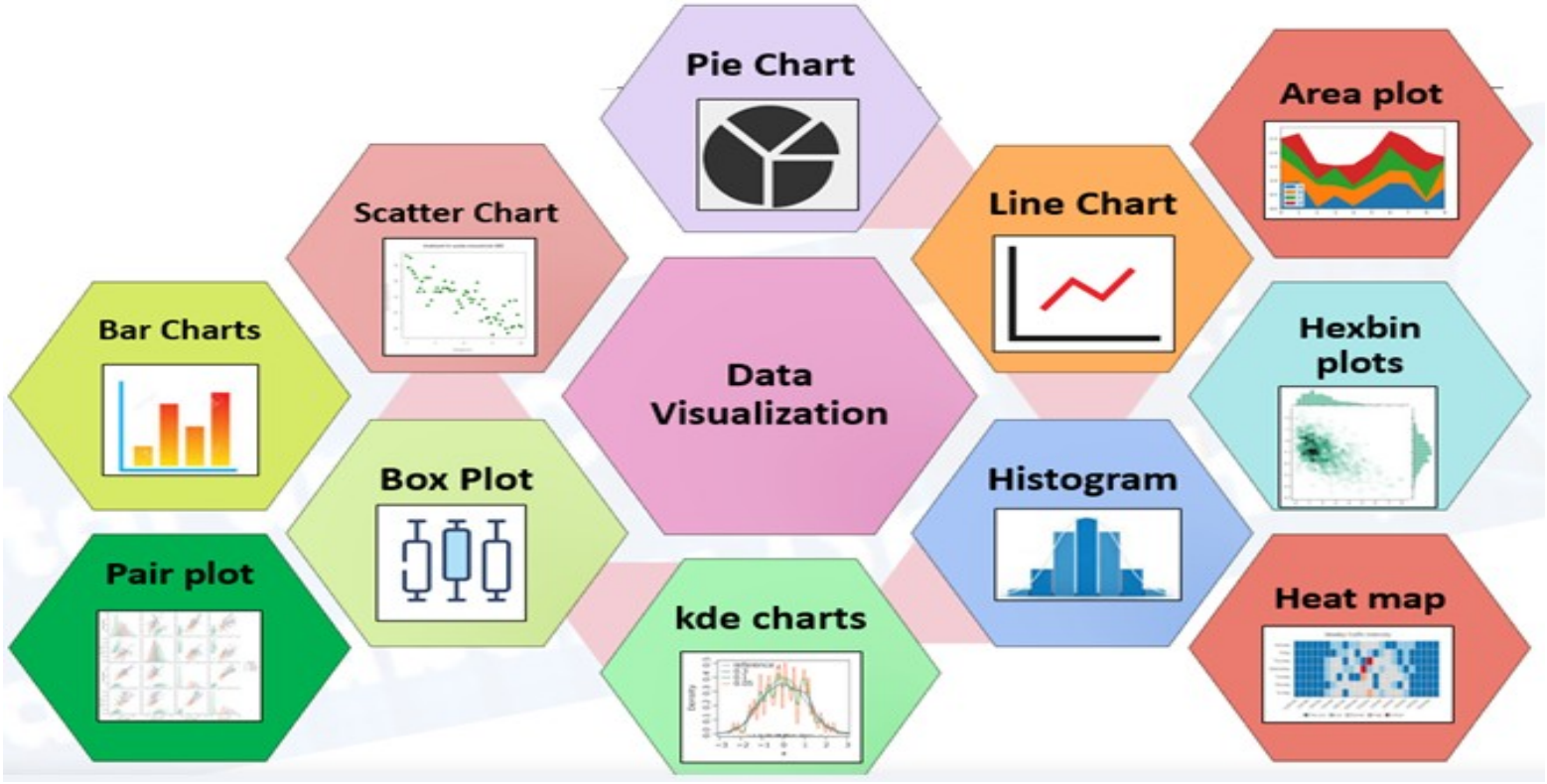
- **Tensor Flow** is a popular machine learning platform developed by Google. Its use cases include natural language processing, image classification, creating neural networks, and more.
- This platform provides a flexible "ecosystem" of libraries, tools, and user resources that are highly portable: You can train and deploy models anywhere, no matter what language or platform you use.
- TensorFlow lets you build and train high-level machine-learning models using the Keras API, a feature of TensorFlow 2.0. It also provides eager execution, allowing for immediate iteration and easier debugging.

7. Scikit-learn

- **Scikit-learn**, also called sklearn, is a library for learning, improving, and executing machine learning models. It builds on NumPy and SciPy by adding a set of algorithms for common machine-learning and data-mining tasks.
- Sklearn is the most popular Python library for performing classification, regression, and clustering algorithms. It's considered a very curated library because developers don't have to choose between different versions of the same algorithm.

Data Visualization

- Data Visualization techniques involve the generation of graphical or pictorial representation of DATA, form which leads you to understand the insight of a given data set. This visualisation technique aims to identify the Patterns, Trends, Correlations, and Outliers of data sets.

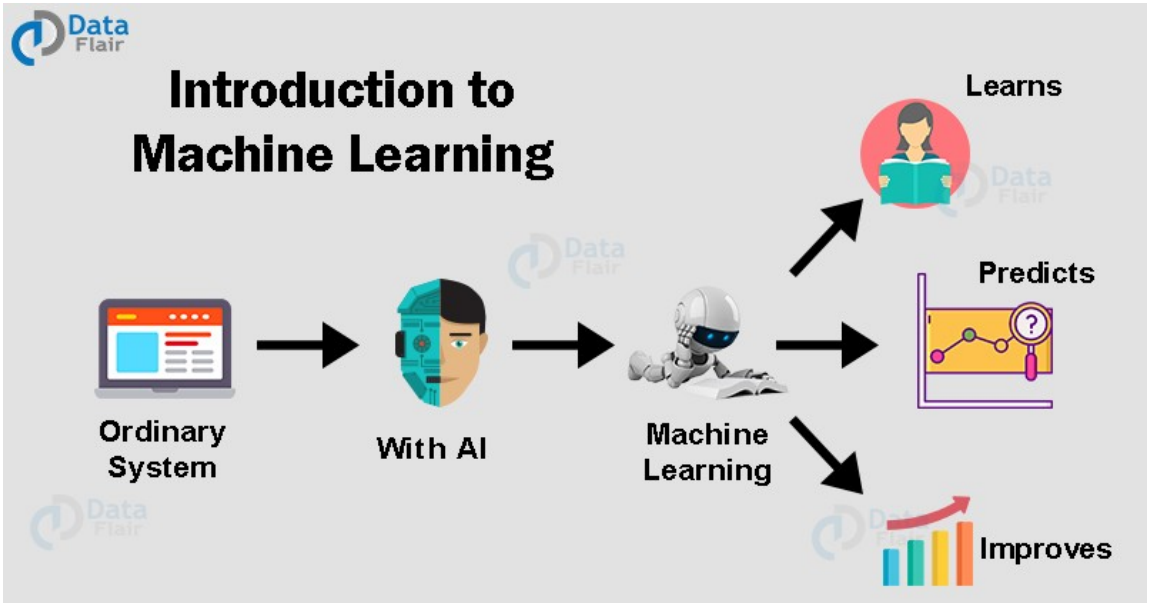


Benefits of Data Visualization

- **Patterns in business operations:** Data visualization techniques help us to determine the patterns of business operations. By understanding the problem statement and identifying the solutions in terms of pattering and applied to eliminate one or more of the inherent problems.
- **Identify business trends and relate to data:** These techniques help us identify market trends by collecting the data on Day-To-Day business activities and preparing trend reports, which helps track the business how influences the market. So that we could understand the competitors and customers. Certainly, this helps to long-term perspective.
- **Storytelling and Decision making:** Knowledge of storytelling from available data is one of the niche skills for business communication, specifically for the Data Science domain which is playing a vital role. Using best visualization this role can be enhanced much better way and reaching the objectives of business problems.
- **Understand the current business insights and setting the goals:** Businesses can understand the insight of the business KPIs, finding tangible goals and business strategy plannings, therefore they could optimize the data for business strategy plans for ongoing activities.
- **Operational and Performance analysis:**
- **Increase the productivity of the manufacturing unit:** With the help of visualization techniques the clarity of KPIs depicting the trends of the productivity of the manufacturing unit, and guiding were to improve the productivity of the plant.

MACHINE LEARNING

- Machine learning (ML) is **a type of artificial intelligence (AI)** that allows software applications to become more accurate at predicting outcomes without being explicitly programmed to do so. Machine learning algorithms use historical data as input to predict new output values.



•Machine Learning is the most popular technique of **predicting** the **future** or **classifying information** to help people in making necessary decisions.

•Machine Learning algorithms are trained over instances or examples through which they learn from **past experiences** and also **analyze** the **historical data**.

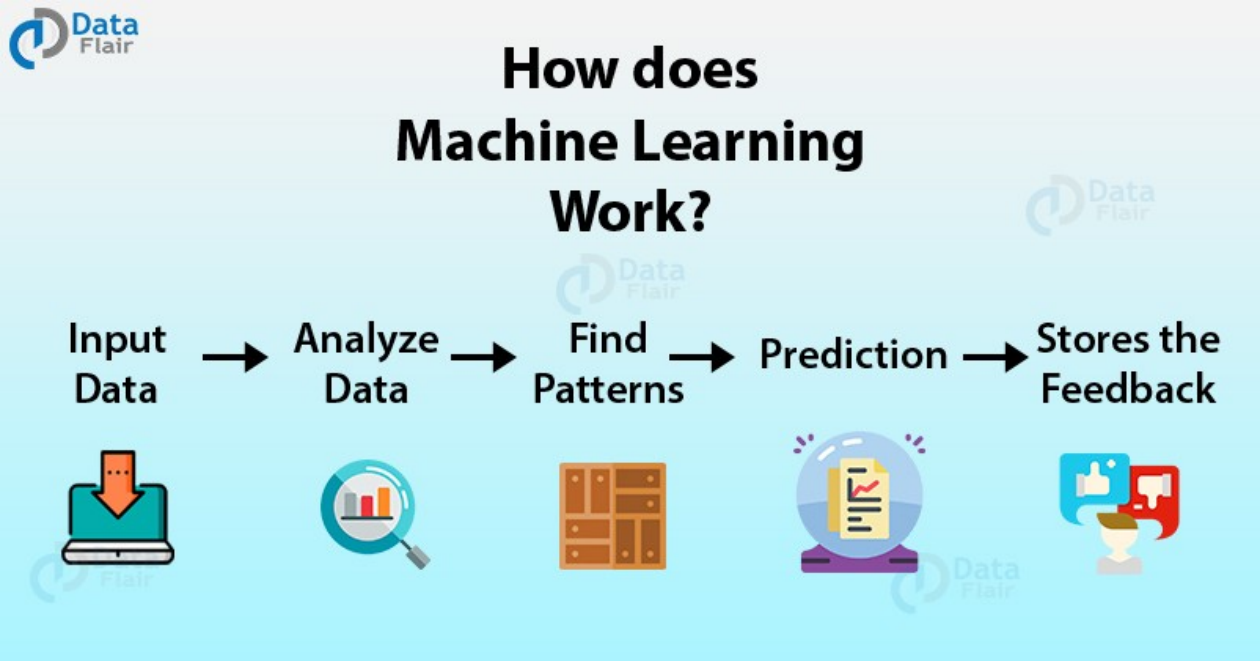
•Therefore, as it trains over the examples, again and again, it is able to identify patterns in order to make predictions about the future.

Why Machine Learning?

- Machine Learning has revolutionized industries like **medicine**, **healthcare**, **manufacturing**, **banking**, and several other industries. Therefore, Machine Learning has become an **essential part** of modern industry.
- Data is powerful and in order to harness the power of this data, added by the massive increase in computation power, Machine Learning has added another dimension to the way we perceive information. Example:
- The electronic devices we use is a **powerful machine learning algorithms**.
- Machine Learning example – **Google** is able to provide you with appropriate search results based on browsing habits.

How does machine learning works?

- With an exponential increase in data, there is a need for having a system that can handle this **massive load** of **data**.
- Machine Learning models like **Deep Learning** allow the vast majority of data to be handled with an **accurate generation** of **predictions**.
- Machine Learning has revolutionized the way we **perceive information** and the **various insights** we can gain out of it.



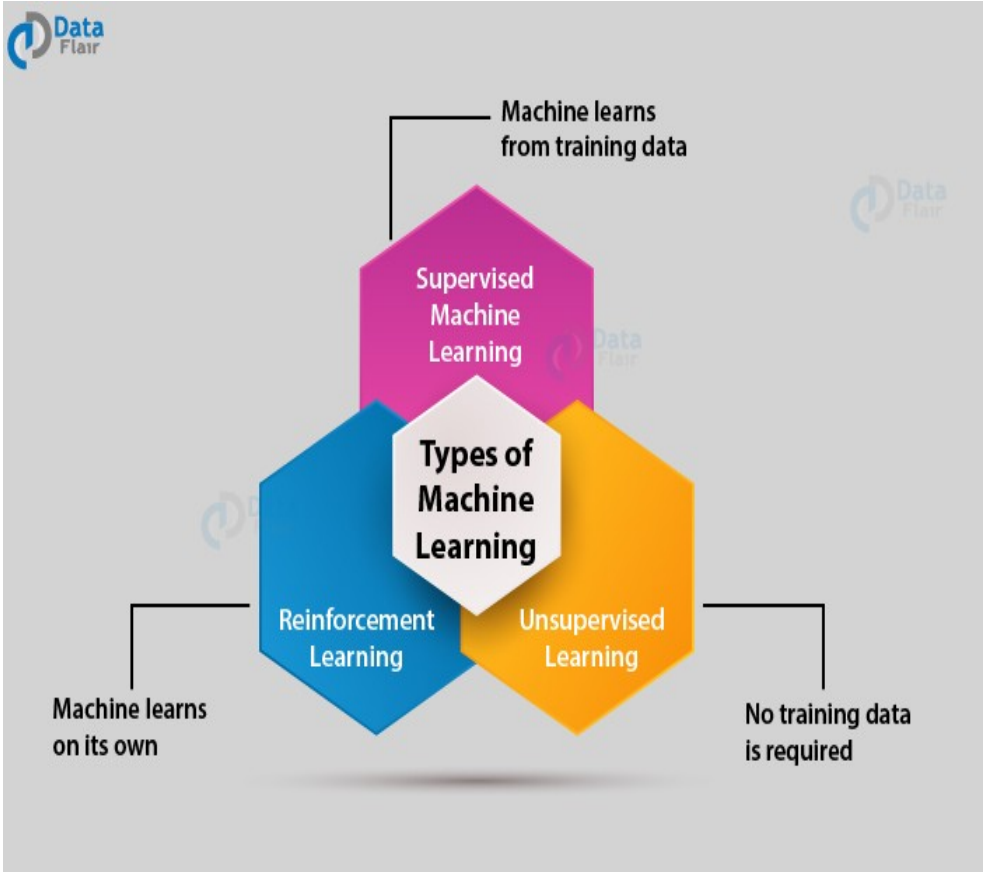
- These machine learning algorithms use the patterns contained in the training data to perform **classification** and **future predictions**.

- Whenever any new input is introduced to the **ML model**, it applies its learned patterns over the new data to **make future predictions**.

- Based on the final accuracy, one can **optimize** their models using various **standardized approaches**.

Types of ML

- **Supervised Learning**
- **Unsupervised Learning**
- **Reinforcement Learning**



Supervised Learning

- Supervised learning is that the machine learning task of learning a function that maps an **input** to an **output** supported example input-output pairs.
- In Supervised Learning, the dataset on which we train our model is **labeled**. There is a clear and **distinct mapping** of input and output. Based on the example inputs, the model is able to get **trained** in the **instances**.
- An example of supervised learning is **spam filtering**.

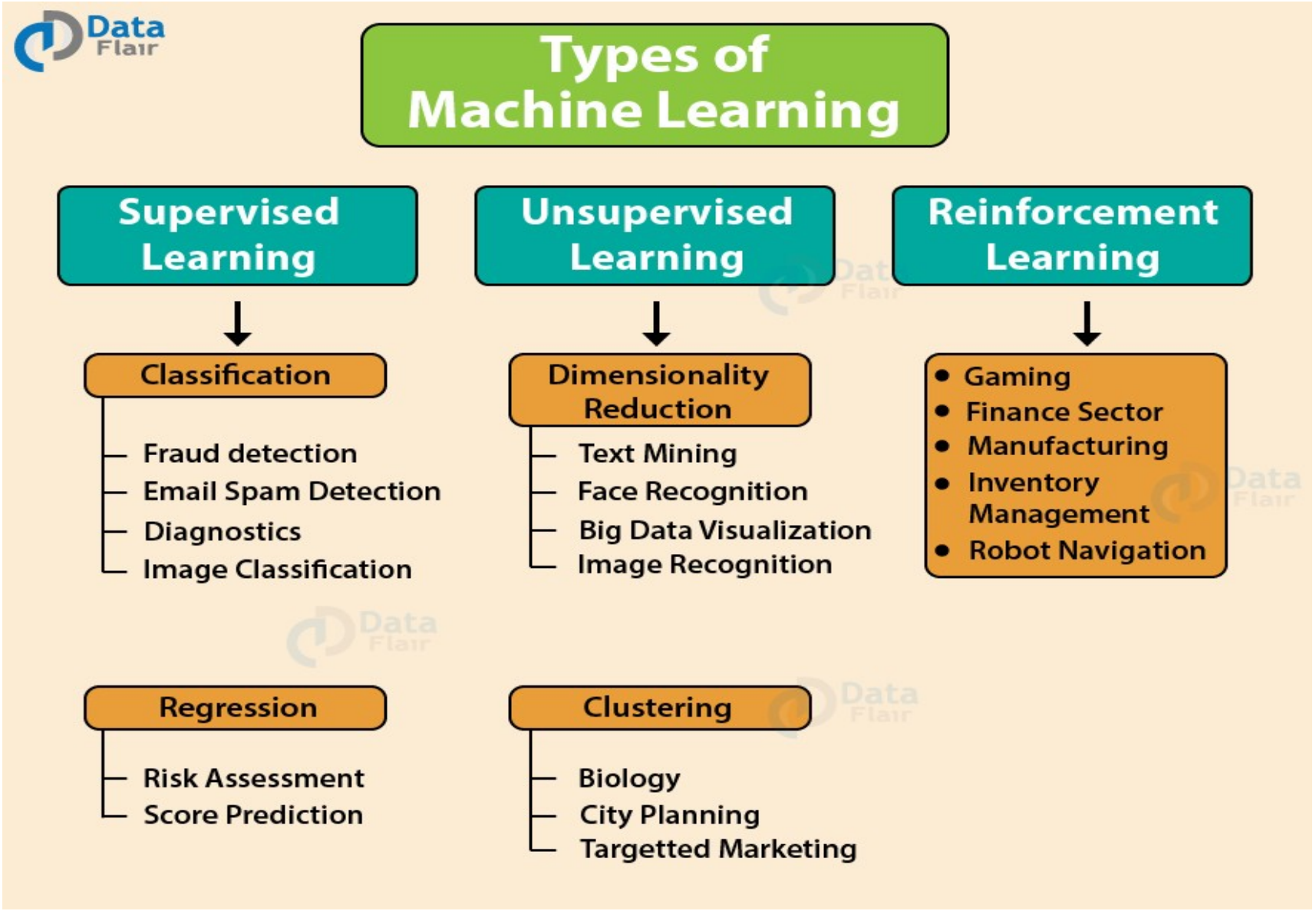
Unsupervised Learning

- It allows the model to figure on its own to get **patterns** and **knowledge** that was **previously undetected**. It mainly deals with the **unlabeled data**.
- In Unsupervised Learning, there is no labeled data. The algorithm identifies the **patterns** within the **dataset** and **learns** them. The algorithm groups the data into **various clusters** based on their **density.** Using it, one can perform **visualization** on **high dimensional data**.
- One example of this type of Machine learning algorithm is the **Principle Component Analysis**
- **K-Mean Cluster** is another type of Unsupervised Learning where the data is clustered in groups of a similar order. The learning process in Unsupervised Learning is solely on the basis of **finding patterns** in the **data**..

Reinforcement Learning

- Reinforcement learning is one among three basic machine learning paradigms, alongside supervised learning and unsupervised learning.
- Reinforcement Learning is an **emerging** and **most popular** type of Machine Learning Algorithm. It is used in various **autonomous systems** like **cars** and **industrial robotics**. The aim of this algorithm is to reach a goal in a **dynamic environment**. It can reach this **goal** based on several rewards that are provided to it by the system.

- It is most heavily used in **programming robots** to perform **autonomous actions**. It is also used in making **intelligent self-driving cars**.
- Let us consider the case of **robotic navigation**.
- Furthermore, the **efficiency** can be improved with further **experimentation** with the agent in its environment. This the main principle behind **reinforcement learning**.


Data Flair
Types of Machine Learning
Supervised Learning
Unsupervised Learning
Reinforcement Learning
Classification
Fraud detection
Email Spam Detection
Diagnostics
Image Classification
Dimensionality Reduction
Text Mining
Face Recognition
Big Data Visualization
Image Recognition
Gaming
Finance Sector
Manufacturing
Inventory Management
Robot Navigation
Regression
Risk Assessment
Score Prediction
Clustering
Biology
City Planning
Targetted Marketing