# Business Analytics

Hui Zeng *

Winter Semester 2020-2021

*All notes are summarized from the lecture and tutorial materials provided by Prof. Martin Bichler and his DSS team. Images are retrieved from the lecture as well as tutorial slides.

# Inhaltsverzeichnis

# 1. Formula Sheet

## 1.1. Basics

**Standard Deviation**

$$SD(x) = \sqrt{\frac{1}{n-1}\Sigma_i(x_i - \bar{x})^2}$$

**Variance**

$$Var(x) = \frac{1}{n-1}\Sigma_i(x_i - \bar{x})^2$$

**Covariance**

$$Cov(x, y) = \frac{1}{n-1}\Sigma_i(x_i - \bar{x})(y_i - \bar{y})$$

**Normal Distribution**

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

standardized:

$$Z = \frac{X - \mu}{\sigma}$$

**Probability density function (pdf): f(x)**

$$f(x) = P(X = x)$$

**Cumulative distribution fuction (cdf): F(x)**

$$F(x) = P(X \leq x)$$

$$P(X \geq x) = 1 - F(x)$$

$$P(x_1 \leq X \leq x_2) = F(x_2) - F(x_1)$$

**Confidence Interval**

$$CI = \left[\bar{X} - z_{(1-\frac{\alpha}{2})} \cdot \frac{\sigma}{\sqrt{n}}, \bar{X} + z_{(1-\frac{\alpha}{2})} \cdot \frac{\sigma}{\sqrt{n}}\right]$$

$$CI = \left[\bar{X} - t_{(1-\frac{\alpha}{2})} \cdot \frac{s}{\sqrt{n}}, \bar{X} + t_{(1-\frac{\alpha}{2})} \cdot \frac{s}{\sqrt{n}}\right]$$

## 1.2. Linear Regression

### 1.2.1. First Order Linear Model

$$Y = \beta_0 + \beta_1 X + \varepsilon$$

**OLS Estimator**

$$\hat{\beta}_1 = \frac{\frac{1}{n-1}\Sigma_i(x_i - \bar{x})(y_i - \bar{y})}{\frac{1}{n-1}\Sigma_i(x_i - \bar{x})^2} = \frac{Cov(x,y)}{Var(x)}$$

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \cdot \bar{x}$$

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x$$

### 1.2.2. Multiple Linear Regression Model

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_k X_k + \varepsilon$$

Formulation in **matrix form**:

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} = \begin{bmatrix} 1 & X_1 1 & X_1 2 & \dots & X_1 k \\ 1 & X_2 1 & X_2 2 & \dots & X_2 k \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & X_m 1 & X_m 2 & \dots & X_m k \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_k \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_m \end{bmatrix}$$

$$[m \times 1] = [m \times (k+1)] \cdot [(k+1) \times 1] + [m \times 1]$$

**OLS Estimator**

- Model:

$$RSS = e^T e = (y - X\hat{\beta})^T (y - X\hat{\beta}) \to \min$$
$$\to \frac{\partial RSS}{\partial \beta} = -2X^T y + 2X^T X\beta = 0$$

- Solution:

$$\hat{\beta} = (X^T X)^{-1} X^T y$$
$$\hat{y} = X(X^T X)^{-1} X^T y$$

### 1.2.3. Quality Metrics of Linear Regression Models

**Residual Sum of Squares (RSS)** An **unbiased** estimator of RSS of the population is given by

$$RSS = \Sigma_{i=1}^{n}(y_i - \hat{y}_i)^2$$

**Mean Squared Error (MSE)**

$$MSE = \frac{RSS}{N}$$

**Root Mean Squared Error (RMSE)**

$$RMSE = \sqrt{MSE}$$

**Total Sum of Squares (TSS)** sum of Explained Sum of Squares(ESS) and Residual Sum of Squares(RSS).

$$\Sigma(y-\bar{y})^2 = \Sigma(\hat{y}-\bar{y})^2 + \Sigma(y-\hat{y})^2$$
$$TSS = ESS + RSS$$

**R-squared $\mathbf{R^2}$** the proportional of explained variablity from the model

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

## 1.3. Logistic Regression & Poisson Regression

### 1.3.1. The Logistic Regression Model

The **binary** logistic regression model described in **log odds/logit**:

$$log\_odds = \ln(\frac{p(X)}{1-p(X)}), \quad \ln(\frac{p(X)}{1-p(X)}) = \beta_0 + \beta_1 X + \varepsilon$$

The logistic regression model described in **odds**:

$$odds = \frac{p(X)}{1-p(X)}, \quad \frac{p(X)}{1-p(X)} = e^{\beta_0+\beta_1 X}$$

### 1.3.2. The Logistic Function

$$Pr[Y|X] = p(X) = \frac{e^{\beta_0+\beta_1 X}}{1+e^{\beta_0+\beta_1 X}}$$

### 1.3.3. Multiple Logistic Regression Model

$$\ln(\frac{p(X)}{1-p(X)}) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_k X_k$$

The logistic function can also be described as a **sigmoid function** in form $S(x) = \frac{e^x}{1+e^x} = \sigma(x)$

$$P(X) = \sigma(\beta_0 + \beta_1 X)$$

### 1.3.4. The Likelihood Function and Maximum Likelihood Estimator

**likelihood function for Logistic Regression Model**

$$\begin{aligned} L &= \Pi_{i=1} p^{y_i}(1-p)^{1-y_i} \\ &= \Pi_{i=1} \sigma(\beta_0+\beta_1 X)^{y_i} \cdot (1-\sigma(\beta_0+\beta_1 X))^{1-y_i} \end{aligned}$$

**Log of Likelihood Function**

$$LL = \ln(L) = \Sigma_{i=1}(y_i \ln(p) + (1-y_i)\ln(1-p))$$

**Maximum Likelihood Estimator**

$$\begin{aligned}\beta = \arg\max_{\beta}(LL) &= \arg\max_{\beta}[\Sigma_{i=1}(y_i \ln(p) + (1-y_i)\ln(1-p))] \\ &= \arg\max_{\beta}[\Sigma_{i=1}(y_i \ln(\sigma(\beta_0+\beta_1 X)) + (1-y_i)\ln(1-\sigma(\beta_0+\beta_1 X)))]\end{aligned}$$

**Gradient(partial derivatives) of the LL-Function** : **chain rule**
with $z = \beta_0 + \beta_1 X$

$$\frac{\partial LL}{\beta_j} = \Sigma_{i=1}\frac{\partial LL}{\partial p}\cdot\frac{\partial p}{\partial z}\cdot\frac{\partial z}{\partial \beta_j}$$

$$\begin{aligned}\frac{\partial LL}{\partial p} &= \frac{y_i}{p} - \frac{1-y_i}{1-p} \\ \frac{\partial p}{\partial z} &= \sigma(z)\cdot(1-\sigma(z)) \\ \frac{\partial z}{\partial \beta_0} &= 1,\ \frac{\partial z}{\partial \beta_j} = x_j\end{aligned}$$

$$\begin{aligned}\frac{\partial LL}{\beta_0} &= \left[\frac{y_i}{p} - \frac{1-y_i}{1-p}\right]\sigma(z)\cdot(1-\sigma(z)) = [y_i - \sigma(X\beta)] \\ \frac{\partial LL}{\beta_j} &= \left[\frac{y_i}{p} - \frac{1-y_i}{1-p}\right]\sigma(z)\cdot(1-\sigma(z))\cdot x_j = [y_i - \sigma(X\beta)]\, x_j\end{aligned}$$

### 1.3.5. Poisson Regression Model

$$ln(\mu(x)) = \beta_0 + \beta_1 X_1 + \ldots \beta_j X_i$$

**random component(dependent variable)**

$$Pr(Y|X) = p(X) = \frac{e^{-\mu}\mu^y}{y!} = \frac{e^{\beta x y}e^{-e^{\beta x}}}{y!}$$

**likelihood function**

$$L(\beta|X,Y) = \Pi_{i=1}p = \Pi_{i=1}\frac{e^{\beta x_i y_i}e^{-e^{\beta x}}}{y_i!}$$

**The Maximum Likelihood Estimator**

$$\log L(\beta|X,Y) = \Sigma_{i=1}(\beta x_i y_i - e^{\beta x_i} - \log(y_i!))$$

## 1.4. Naive Bayes & Bayesian Network

### 1.4.1. Bayes Theorem

- single evidence:

$$Pr(h|e) = \frac{Pr(h \cap e)}{Pr(e)} = \frac{Pr(e|h) \cdot Pr(h)}{Pr(e)}$$

- multiple evidence:

$$\begin{aligned} Pr(h|e_1, e_2, \ldots, e_k) &= \frac{Pr(e_1|h) \cdot Pr(e_2|h) \ldots Pr(e_k|h) \cdot Pr(h)}{Pr(e_1, e_2, \ldots, e_k)} \\ &= \frac{\Pi_{i=1}^{k} Pr(e_i|h) \cdot Pr(h)}{Pr(e_1, e_2, \ldots e_k)} \end{aligned}$$

### 1.4.2. Pr(e)

- If the prior probability $Pr(e_i)$ is **known**:

$$Pr(e_1, e_2, \ldots, e_k) = Pr(e_1) \cdot Pr(e_2) \ldots Pr(e_k)$$

- If the prior probability $Pr(e_i)$ is **unknown**: law of total probability

$$Pr(e_1, e_2, \ldots, e_k) = Pr(e_1, e_2, \ldots, e_k|h) \cdot Pr(h) + Pr(e_1, e_2, \ldots, e_k|\neg h) \cdot Pr(\neg h)$$

### 1.4.3. Chain Rule

According to the **directed acyclic graph**, derive the **joint probability distribution**

$$Pr(e_1, e_2, \ldots, e_k) = \Pi_{i=1} Pr(e_i|e_{i-1}, \ldots, e_1) = \Pi_{i=1} Pr(e_i|\text{Parents}(e_i))$$

### 1.4.4. Conditional Independence

**conditional independence between hypothesis and evidence** the hypothesis $h$ is only dependent on $e_1, e_2, e_3$, not on $e_4$ (redundant), then

$$Pr(h|e_1, e_2, e_3, e_4) = Pr(h|e_1, e_2, e_3)$$

**conditional independence between hypotheses** if two hypotheses are **independent** from each other, then

$$Pr(h_1, h_2|e_1, e_2) = Pr(h_1|e_1, e_2) \cdot Pr(h_2|e_1, e_2)$$

## 1.5. Decision Tree

**Entropy** $\in [0, 1]$, measures how much **additional information required** in **bits**

$$\text{entropy}(p_1, \ldots, p_n) = -\Sigma_{i=1}^{n} p_i \cdot \log_2 p_i$$

**Information of Each Attribute Value**

$$\text{info}([c_1, \ldots, c_n]) = \text{entropy}(\frac{c_1}{C}, \ldots, \frac{c_n}{C})$$

**Information of the Attribute** the **weighted average** of the **information needed** from each attribute value.
Say an attribute has $m$ attribute values/branches,

$$\text{info}([c_1,\ldots,c_n]_1,\ldots,[c_1,\ldots,c_n]_m) = \Sigma_{i=1}^{m}\frac{C_m}{N}\cdot\text{info}([c_1,\ldots,c_n])_m$$

**Information Gain of the Attribute**

$$\text{Information\_Gain(attribute)} = \text{info(before split by attribute)} - \text{info(after split by attribute)}$$

**Intrinsic Information of a Attribute** s: size of a leaf from each branch

$$\text{intrinsic\_info}([s_1,\ldots,s_n]) = \text{info}([s_1,\ldots,s_n])$$

**Gain Ratio of a Attribute**

$$\text{Gain\_Ratio(attribute)} = \frac{\text{Gain(attribute)}}{\text{Intrinsic\_Info(attribute)}}$$

## 1.6. Evaluation

**Accuracy**

$$\text{Accuracy} = \frac{TP+TN}{N}$$

**Error Rate**

$$\text{Error Rate} = 1 - \text{Accuracy} = \frac{FP+FN}{N}$$

**True Positive Rate / Recall / Hit Rate**

$$\text{True Positive Rate/Recall} = \frac{TP}{TP+FN}$$

**True Negative Rate / Specificity**

$$\text{True Negative Rate/Specificity} = \frac{TN}{TN+FP}$$

**False Positive Rate / False Alarm Rate**

$$\text{False Positive Rate/False Alarm Rate} = 1 - \text{Specificity} = \frac{FP}{TN+FP}$$

**Precision**

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

## 1.7. Clustering: Expectation Maximization

**Expectation Step** calculate the probability for **all instances** in **each cluster**.

$$Pr(A|x) = \frac{Pr(x|A) \cdot Pr(A)}{Pr(x)}, \quad Pr(x|A) = \frac{1}{\sqrt{2\pi} \cdot \sigma_A} e^{-\frac{(x-\mu_A)^2}{2\sigma_A^2}}$$

$$Pr(B|x) = \frac{Pr(x|B) \cdot Pr(B)}{Pr(x)}, \quad Pr(x|B) = \frac{1}{\sqrt{2\pi} \cdot \sigma_B} e^{-\frac{(x-\mu_B)^2}{2\sigma_B^2}}$$

**Maximization Step** calculate the weighted mean and weighted variance using **all instances**.

$$w_{iA} = Pr(A|x), \quad w_{iB} = Pr(B|x)$$

$$\mu_A = \frac{w_{1A}x_1 + w_{2A}x_2 + \cdots + w_{nA}x_n}{w_{1A} + w_{2A} + \cdots + w_{nA}}$$

$$\sigma_A = \sqrt{\frac{w_{1A}(x_1 - \mu_A)^2 + \cdots + w_{nA}(x_n - \mu_A)^2}{w_{1A} + w_{2A} + \cdots + w_{nA}}}$$

analog to $\mu_B$ and $\sigma_B$

$$Pr(A) = \frac{\Sigma w_A}{\Sigma w_A + \Sigma w_B}, \quad Pr(B) = 1 - Pr(A)$$

## 1.8. Principal Component Analysis & Restoring Original Data

- Calculate the dimension means: $\bar{d}_j = \frac{1}{N}\sum_{i=1}^{N} d_{ij}$
- Subtract means: $x_j = d_j - \bar{d}_j$
- The covariance matrix: $\sum_x = \begin{bmatrix} var(x_1) & cov(x_1,x_2) & cov(x_1,x_3) \\ cov(x_2,x_1) & var(x_2) & cov(x_2,x_3) \\ cov(x_3,x_1) & cov(x_3,x_2) & var(x_3) \end{bmatrix}$
- Calculate the covariance matrix:

$$cov(x_{j_1}, x_{j_2}) = \frac{1}{N-1}\sum_{i=1}^{N}(x_{ij_1} - \bar{x}_{j_1}) \cdot (x_{ij_2} - \bar{x}_{j_2}) = \frac{1}{N-1}\sum_{i=1}^{N} x_{ij_1}\, x_{ij_2}$$

- Find the eigenvalues by solving the characteristic equation: $|\sum_x - \lambda \boldsymbol{I_p}| = 0$
- Calculate the eigenvectors: $(\sum_x - \lambda \boldsymbol{I_p})v = 0$
- Calculate the variance explained by each component: $\frac{\lambda_i}{\sum_{i=1}^{n}\lambda_i}$
- Projecting the transformed data: $Z = X\Phi$
- Restoring the original dataset: $D \approx Z\Phi^T + means$

## 1.9. Recommendation Systems: Association Rules

support of **a rule**: the support of all item sets it contains.

$$supp(A, B \Rightarrow C, D) = supp(\{A, B, C, D\})$$

**Confidence of a Rule** the probability that X and Y coexist given that X exists.

$$conf(R : X \Rightarrow Y) = \frac{supp(X \cup Y)}{supp(X)}$$

**Lift of a Rule** indicates **by how much (ratio)** the **confidence of a rule** surpasses the **expected value**.

$$Lift(R : X \Rightarrow Y) = \frac{conf(R)}{expConf(R)} = \frac{\frac{supp(X \cup Y)}{supp(X)}}{supp(Y)} = \frac{supp(X \cup Y)}{supp(X) \cdot supp(Y)}$$

## 1.10. Recommendation Systems: Collaborative Filtering

### Weighted Correlation

$$w_{a,u} = s_{a,u} \cdot c_{a,u}$$

$$c_{a,u} = \frac{Cov(r_a, r_u)}{\sigma_{r_a} \cdot \sigma_{r_u}}$$

$$Cov(r_a, r_u) = \frac{1}{m-1} \cdot \Sigma(r_a - \bar{r}_a)(r_u - \bar{r}_u)$$

### Rating Prediction

$$p_{a,i} = \bar{r}_a + \Sigma_{u=1}^{k} \frac{w_{a,u} \cdot (r_{u,i} - \bar{r}_u)}{\Sigma_{u=1}^{k} |w_{a,u}|}$$

## 1.11. Recommendation Systems: SVD

### Rating Prediction

$$r_{u,i} = \bar{r}_u + U(user) \cdot S \cdot V^T(item)$$

## 1.12. Neural Network

### Forward Pass

$$z^{[1]} = W^{[1]} \cdot a^{[0]} + b^{[1]} = W^{[1]} \cdot x + b^{[1]}$$
$$a^{[1]} = g^{[1]}(z^{[1]}) = \sigma(z^{[1]})$$
$$z^{[2]} = W^{[2]} \cdot a^{[1]} + b^{[2]}$$
$$a^{[2]} = g^{[2]}(z^{[2]}) = \sigma(z^{[2]})$$

**Loss Function** If we evaluate the model using **cross-entropy loss**: the calculation for $y \ln \hat{y}$ is a dot product(element-wise multiplication).

$$l(y, \hat{y}) = -[y \ln \hat{y} + (1 - y) \ln(1 - \hat{y})]$$

**Empirical Risk** **average** the loss.

$$\mathcal{L}(y, \hat{y}) = \frac{1}{n} \cdot \Sigma l(y, \hat{y})$$

**Backpropagation** example in updating layer 2:

$$W_{t+1}^{[2]} = W_t^{[2]} - \alpha \cdot dW = W_t^{[2]} - \alpha \cdot \frac{\partial L}{\partial W^{[2]}}$$
$$b_{t+1}^{[2]} = b_t^{[2]} - \alpha \cdot db = b_t^{[2]} - \alpha \cdot \frac{\partial L}{\partial b^{[2]}}$$

$$\frac{\partial L_n}{\partial W} = \frac{\partial L_n}{\partial a_n} \cdot \frac{\partial a_n}{\partial z_n} \cdot \frac{\partial z_n}{\partial W}$$

$$L_n = \frac{1}{2}(y_n - g(w_{kl} a_{kn} + b_l))^2 = \frac{1}{2}(y_n - a_{ln})^2, \qquad \frac{\partial L_n}{\partial a_{ln}} = -(y_n - a_{ln})$$
$$a_{ln} = g(z_{ln}), \qquad \frac{\partial L_n}{\partial z_{ln}} = g'(z_{ln})$$
$$z_{ln} = w_{kl} a_{kn} + b_l, \qquad \frac{\partial L_n}{\partial w_{ln}} = a_{kn}$$

If the activation is a sigmoid activation: $\sigma(x) = \frac{e^x}{1+e^x}$, $\sigma'(x) = \sigma(x)(1 - \sigma(x))$.

$$dW^{[2]} = -(y - a^{[2]}) \cdot a^{[1]^T} = (a^{[2]} - y) \cdot a^{[1]^T}$$

**Gradient Descent**

**fixed step size**

$$\begin{bmatrix} x_n \\ y_n \end{bmatrix} = \begin{bmatrix} x_{n-1} \\ y_{n-1} \end{bmatrix} - \alpha \cdot \nabla f(x_{n-1}, y_{n-1})$$

**dynamic step size**

$$\begin{bmatrix} x_n \\ y_n \end{bmatrix} = \begin{bmatrix} x_{n-1} \\ y_{n-1} \end{bmatrix} - \alpha_n \cdot \nabla f(x_{n-1}, y_{n-1})$$

**Momentum**

$$d_n = \beta \cdot d_{n-1} + \alpha \cdot \nabla f(x_{n-1})$$
$$x_n = x_{n-1} - d_n$$

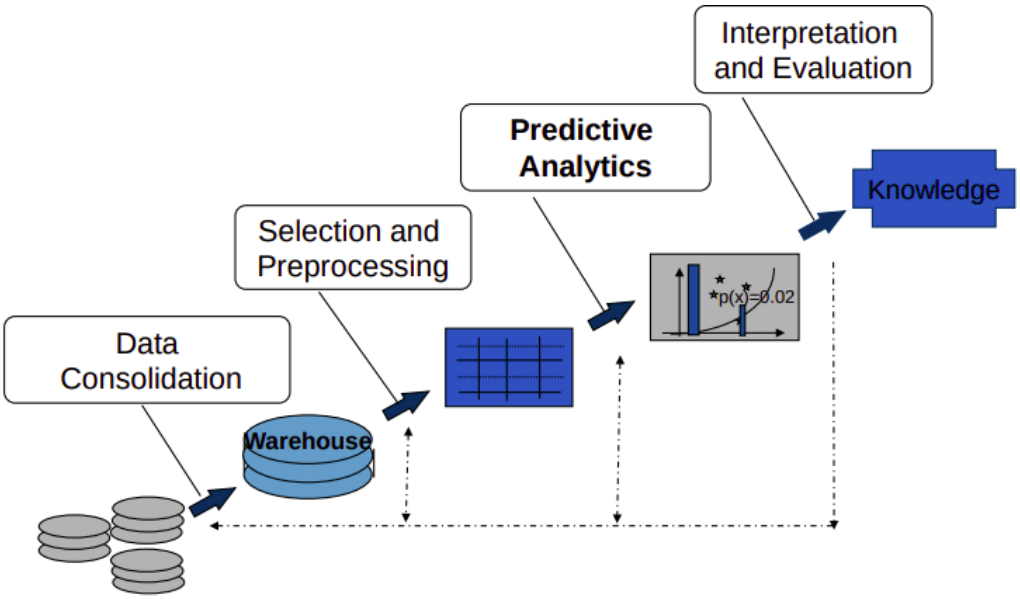
# Teil I.
# Introduction

## 2. Terminologies & Intros

### 2.1. Categories of Business Analytics

- descriptive analytics
  - data engineering(organizing data, queries) & statistics (mean, trend, standard deviation, test hypotheses)
- predictive analytics
  - machine learning & econometrics
  - learn the pattern of data
- prescriptive analytics
  - algorithm & optimization models

### 2.2. From Data to Information




### 2.3. Types of Analytic Exercises

- Numeric Prediction
  - **supervised** learning
  - input: a collection of data input + **known** output
    job: create a function $f(\text{input}_0) = \text{output}_0$
    output: **prediction** values to a **new** collection of data: $f(\text{input}_1) = ?$

  - example: linear regression

- Classification
  - **supervised** learning
  - input: a collection of data input + **known** label

    job: create a classifier $f(\text{input}_0) = \text{label}_0$

    output: **prediction** label to a **new** collection of data: $f(\text{input}_1) = ?$

- Clustering
  - **unsupervised** learning
  - input: a collection of data input

    job: identify "natural"grouping in data

    output: clustered data

- Association Rule Analysis
  - **unsupervised** learning
  - input: a collection of data list

    job: identify **relationships** in data from **co-occuring items**
  - grocery store purchases analysis

## 2.4. Machine Learning Terminology in categorizing analytic exercises

- **supervised learning**
  - a training set is given
  - find relationship between input & target attributes
  - examples: numeric prediction, classification
- **unsupervised learning**
  - only input data available, no training set
  - find regularities, irregularities, relationships, similarities among data points
  - examples: clustering, association rule analysis

## 2.5. Model

examples:

- linear model
- decision tree
- neural network

# 3. Statistics Recap

## 3.1. Categories

- descriptive statistics: **summary** of data
  - examples: mean, standard deviation
- inferential statistics: model **patterns** of data, accounting for **randomness** and drawing **inferences about a larger population**
  - estimation
  - hypothesis testing
  - forecasting
  - correlation
  - regression

## 3.2. Random Variables

$X$ is a **random variable** if:

- it represents a **random draw** from a population
- it's associated with a **probability distribution**
- either **discrete** or **continuous**
- example: a random variable that follows a **normal distribution** $N(\mu, \sigma^2)$ has a probability density function of

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

## 3.3. Normal Distribution & Standard Normal Distribution

A random variable $X$ follows a normal distribution $N(\mu, \sigma^2)$
To **"standardize"** a random variable $x$ to **standard normal distribution** $N(0, 1)$:

$$Z = \frac{X - \mu}{\sigma}$$

## 3.4. Probability density function & Cumulative density function

**Probability density function (pdf): f(x)**

$$f(x) = P(X = x)$$

**Cumulative distribution fuction (cdf): F(x)**

$$F(x) = P(X \leq x)$$

$$P(X \geq x) = 1 - F(x)$$

$$P(x_1 \leq X \leq x_2) = F(x_2) - F(x_1)$$

## 3.5. Statistical Estimation

**Statistical estimation**:
The parameters of a population are **unknown**. However, we can **estimate** the parameters by **drawing a random sample** out of the population, analyzing this random sample and getting the statistics. The statistics should infer the parameters.

Requirements of random sampling:

- each variable $X$ from the population is a random variable.
- each combination of $n$ sample points has an **equal** chance of being selected.

$\longrightarrow$ a random sample is a set of **independent, identically distributed (i.i.d)** random variables.

**Categories:**

- Point Estimate
  - sample mean
  - sample proportion
- Interval Estimate
  - **confidence interval** for sample mean
  - **confidence interval** for sample proportion

Point estimate is always within the interval estimate.

### 3.5.1. Point Estimate: Population Mean and Its Estimation

- **Expected Value of X and the Population Mean**

  The expected value of a probability **weighted average** of $X$, $E(X)$, is the mean/expected value of the probability distribution of $X$.

  $f(x_i)$ is the (discrete) probability that $X = x_i$.

  $$\mu_x = E(X) = \Sigma_{i=1}^{n} x_i f(x_i)$$

  or

  $$\mu_x = E(X) = \int_{-\infty}^{+\infty} x f(x) dx$$

  If this random sample is a set of i.i.d random variables, the **expected value of X** is the **population mean**(unknown).

- **Estimation of the Population Mean by Sample Mean**

  **Sample Mean $\bar{X}$:** the **random variable** for the arithmetic mean of the sample. $\bar{x}$ is the mean of a **particular realization** of a sample.

$$\bar{X} = \frac{\Sigma X_i}{n}$$

This is a random variable, because a lot of samples are drawn repeatedly, the arithmetic mean of the sample has also a probability distribution. **The mean(center) of this distribution should estimate the mean of the whole population.**

- Requirements for an estimator: **unbiased**

  Example:

  $$E(\bar{X}) = \mu_x$$

- **Standard Error of the Sample Mean**:
  - **standard error of the sample mean** $SE_{\bar{X}}$ : an estimate of how far the **sample mean** is likely to be **from the population mean**.

    When $n \to \infty$, $SE_{\bar{X}} \to \sigma_{\bar{X}}$ (**true standard deviation of sample mean**).

    $$SE_{\bar{X}} = \frac{s}{\sqrt{n}}$$

    $$\sigma_{\bar{X}} = SD(\bar{X}) = \sqrt{Var(\bar{X})} = \frac{\sigma}{\sqrt{n}}$$

  - **sample standard deviation** $s$ : the degree to which **individuals within the sample** differ from the **sample mean**

    $$s = \sqrt{\frac{1}{n-1}\Sigma_{i=1}^{n}(X_i - \bar{X})^2}$$

- **Law of Large Numbers**

  If $n \to \infty, \bar{X}_n \to \mu$.

  If the size of the random sample is large enough, then the arithmetic mean converge to the real population mean.

  $$\lim_{n\to\infty} P(|\bar{X}_n - \mu| > \epsilon) = 0$$

- **Central Limit Theorem**

  If $n \to \infty$, the **average** $\bar{X}$ of any population of **i.i.d.** random variables $X_i$ with the population mean $\mu_X$ and population variance $\sigma^2$ **follows asymptotically a normal distribution** $\bar{X} \sim N(\mu_X, \frac{\sigma^2}{n})$.

  $$\bar{X} = \frac{X_1 + X_2 + \cdots + X_n}{n}$$

  The **standardize average** $Z \sim N(0, 1)$:

  $$Z = \frac{\bar{X} - \mu_X}{\frac{\sigma}{\sqrt{n}}}$$

### 3.5.2. Interval Estimate:

- **Confidence Interval of Sample Mean:**
  - Assumption: samples drawn from a population that follows a normal distribution $N(\mu_X, \sigma^2)$.

    eg: The sample mean $\bar{X}$ follows asymptotically a normal distribution $N(\mu_X, \frac{\sigma^2}{n})$
  - a level of confidence$(1-\alpha)$ is given.
  - **two-sided** $\rightarrow z_{(1-\frac{\alpha}{2})}$ / $z_{(1+\frac{\alpha}{2})}$
  - z value: **standardized**. **Find z value from cdf-table given an** $\alpha$.
  - if population standard deviation $\sigma$ is given,

$$CI = \left[\bar{X} - z_{(1-\frac{\alpha}{2})} \cdot \frac{\sigma}{\sqrt{n}}, \bar{X} + z_{(1-\frac{\alpha}{2})} \cdot \frac{\sigma}{\sqrt{n}}\right]$$

$$Pr(\bar{X} - z_{(1-\frac{\alpha}{2})} \cdot \frac{\sigma}{\sqrt{n}} < \mu_X < \bar{X} + z_{(1-\frac{\alpha}{2})} \cdot \frac{\sigma}{\sqrt{n}}) = 1 - \alpha$$

  - if population standard deviation $\sigma$ is unknown,
    - n is **small**: use **sample standard deviation** $s$ and **t-distribution**

$$CI = \left[\bar{X} - t_{(1-\frac{\alpha}{2})} \cdot \frac{s}{\sqrt{n}}, \bar{X} + t_{(1-\frac{\alpha}{2})} \cdot \frac{s}{\sqrt{n}}\right]$$

    - n is **large**: use **sample standard deviation** $s$ and **normal distribution**

$$CI = \left[\bar{X} - z_{(1-\frac{\alpha}{2})} \cdot \frac{s}{\sqrt{n}}, \bar{X} + z_{(1-\frac{\alpha}{2})} \cdot \frac{s}{\sqrt{n}}\right]$$

    If $n \rightarrow \infty$, the T-Distribution converges to a Normal Distribution.

- Effects on Confidence Intervals:
  - **sample size** n: n ↑, interval size ↓

    (the larger the size, more precise is the estimation)
  - **confidence level** $(1-\alpha)$: confidence level ↑, interval size ↑

    (given the same sample size, the higher the confidence level, the more values need to be included.)
  - **population standard deviation** $\sigma$: $\sigma$ ↑, interval size ↑

    (the more spreaded the population, the more values need to be included to achieve same confidence level.)

## 3.6. Statistical Tests

### 3.6.1. Process

- number of samples
  - 1 sample:
    - $\sigma$ known: **Z-Test**
    - $\sigma$ unknown: **T-Test**
  - 2 samples:
    - dependent: **Paired T-Test**
    - independent: **Welch-Test**
- Formulate **null and alternative hypothesis** ($H_0$ and $H_1$), $H_1$ is the hypothesis we want to test.
- Choose an $\alpha$ **level**. (type I error: probability of falsely rejecting $H_0$)
- Find corresponding **distribution**, calculate **test statistic**,
- Find the **critical value** in the table and corresponding **p-value**
- Conclusion:
  - $p \leq \alpha$, reject $H_0$
  - $p > \alpha$, reject $H_1$

  Interpretation of p-value: the probability of having the other mean $\bar{x}$, given that $H_0$ is true.

### 3.6.2. Two-sided or One-sided Test

Three possible alternative hypotheses $H_1$:

| Hypothesis | $H_0$ | $H_1$ |
|---|---|---|
| Two-sided | $\mu_x = \mu_0$ | $\mu_x \neq \mu_0$ |
| One-sided | $\mu_x \leq \mu_0$ | $\mu_x > \mu_0$ |
| One-sided | $\mu_x \geq \mu_0$ | $\mu_x < \mu_0$ |

Critical Value:

- two-sided test: $z_{\frac{\alpha}{2}}, z_{1-\frac{\alpha}{2}}$ / $t_{\frac{\alpha}{2}}, t_{1-\frac{\alpha}{2}}$
- one-sided test: $z_{1-\alpha}$ / $t_{1-\alpha}$

### 3.6.3. Z-Test

- Requirements: **1 sample,** $\mu$**,** $\sigma$ **known** (population mean and population standard deviation)
- Distribution: **standardized normal distribution**
- test statistic:

$$z = \frac{\bar{X} - \mu_0}{\frac{\sigma}{\sqrt{n}}}$$

- critical value: $z^c_{1-\frac{\alpha}{2}}$ / $z^c_{1-\alpha}$
- $H_0$ Rejection region/ $H_1$ Acceptance region :

| $H_1$ | Rejection Region | test variant |
|---|---|---|
| $\mu\mu \neq \mu_0$ | $\lvert z\rvert \geq z^c_{1-\frac{\alpha}{2}}$ | two-sided |
| $\mu\mu > \mu_0$ | $z \geq z^c_{1-\alpha}$ | one-sided |
| $\mu\mu < \mu_0$ | $z \leq -z^c_{1-\alpha}$ | one-sided |

### 3.6.4. Single Sample T-test

- Requirements: **1 sample,** $\sigma$ **unknown**
- Distribution: Student **t-Distribution**
- Degree of Freedom($df$): determines how spread the distribution is.

$$df = n - 1$$

- test statistic:

  $s$ is the standard error(empirical)

$$t(df) = \frac{\bar{X} - \mu_0}{\frac{s}{\sqrt{n}}}$$

- critical value: $t^c_{1-\frac{\alpha}{2},df}$ / $t^c_{1-\alpha,df}$
- $H_0$ Rejection region/ $H_1$ Acceptance region :

| $H_1$ | Rejection Region | test variant |
|---|---|---|
| $\mu\mu \neq \mu_0$ | $\lvert t\rvert \geq t^c_{1-\frac{\alpha}{2},df}$ | two-sided |
| $\mu\mu > \mu_0$ | $t \geq t^c_{1-\alpha,df}$ | one-sided |
| $\mu\mu < \mu_0$ | $t \leq -t^c_{1-\alpha,df}$ | one-sided |

### 3.6.5. Paired Sample T-test

- Requirements: **2 samples, $\sigma$ unknown, dependent**

  eg: means obtained in 2 conditions(time,places,etc.) by a single group of participants
- Distribution: **T-Distribution**
- Test of relationship between 2 linked samples (eg: Difference)
- Degree of Freedom($df$):
$$df = n - 1$$
- Example null hypothesis: $H_0 : \mu_d = \mu_1 - \mu_2 = \Delta_0$
- test statistic:
$$t = \frac{\bar{d} - \Delta_0}{\frac{s}{\sqrt{n}}}$$
- critical value: $t^c_{1-\frac{\alpha}{2},df}$ / $t^c_{1-\alpha,df}$
- $H_0$ Rejection Region/ $H_1$ Acceptance Region:

| $H_1$ | Rejection Region | test variant |
|---|---|---|
| $\mu_d \neq \Delta_0$ | $\lvert t\rvert \geq t^c_{1-\frac{\alpha}{2},df}$ | two-sided |
| $\mu_d > \Delta_0$ | $t \geq t^c_{1-\alpha,df}$ | one-sided |
| $\mu_d < \Delta_0$ | $t \leq -t^c_{1-\alpha,df}$ | one-sided |

### 3.6.6. Independent T-Test/ Welch-Test

- Requirements: **2 samples, $\sigma$ unknown, independent**

  eg: credit card debt difference between urban and rural households/ shopping expenses between male and female
- Distribution: **T-Distribution**
- Test of relationship between 2 independent samples (doesn't need to be same size)
- Degree of Freedom($df$): round to **nearest integer** number
$$df = \frac{(\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2})^2}{\frac{(\frac{s_1^2}{n_1})^2}{n_1 - 1} + \frac{(\frac{s_2^2}{n_2})^2}{n_2 - 1}}$$

- test statistic:

$$t = \frac{(\bar{x}_1 - \bar{x}_2) - \mu_0}{s_{\bar{x}_1 - \bar{x}_2}}$$

$$s_{\bar{x}_1 - \bar{x}_2} = \sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}$$

- $H_0$ Rejection Region/ $H_1$ Acceptance Region:

| $H_1$ | Rejection Region | test variant |
|---|---|---|
| $\mu_d \neq \Delta_0$ | $\lvert t \rvert \geq t^c_{1-\frac{\alpha}{2},df}$ | two-sided |
| $\mu_d > \Delta_0$ | $t \geq t^c_{1-\alpha,df}$ | one-sided |
| $\mu_d < \Delta_0$ | $t \leq -t^c_{1-\alpha,df}$ | one-sided |

- Confidence Interval of both samples: if both confidence intervals overlap each other $\rightarrow$ **cannot reject** $H_0$

### 3.6.7. Using Confidence Intervals in Significance Tests

Find confidence intervals for $\mu_x$, which – under $H_0$ – contains the true value $\mu_x$ with a probability of at least $1 - \alpha$:

- $\sigma$ known: **Normal Distribution**

$$CI = [\bar{x} - z^c_{(1-\frac{\alpha}{2})} \cdot \frac{\sigma}{\sqrt{n}}, \bar{x} + z^c_{(1-\frac{\alpha}{2})} \cdot \frac{\sigma}{\sqrt{n}}]$$

- $\sigma$ unknown: **t-Distribution**

$$CI = [\bar{x} - t^c_{(1-\frac{\alpha}{2},n-1)} \cdot \frac{s}{\sqrt{n}}, \bar{x} + t^c_{(1-\frac{\alpha}{2},n-1)} \cdot \frac{s}{\sqrt{n}}]$$

Conclusion:

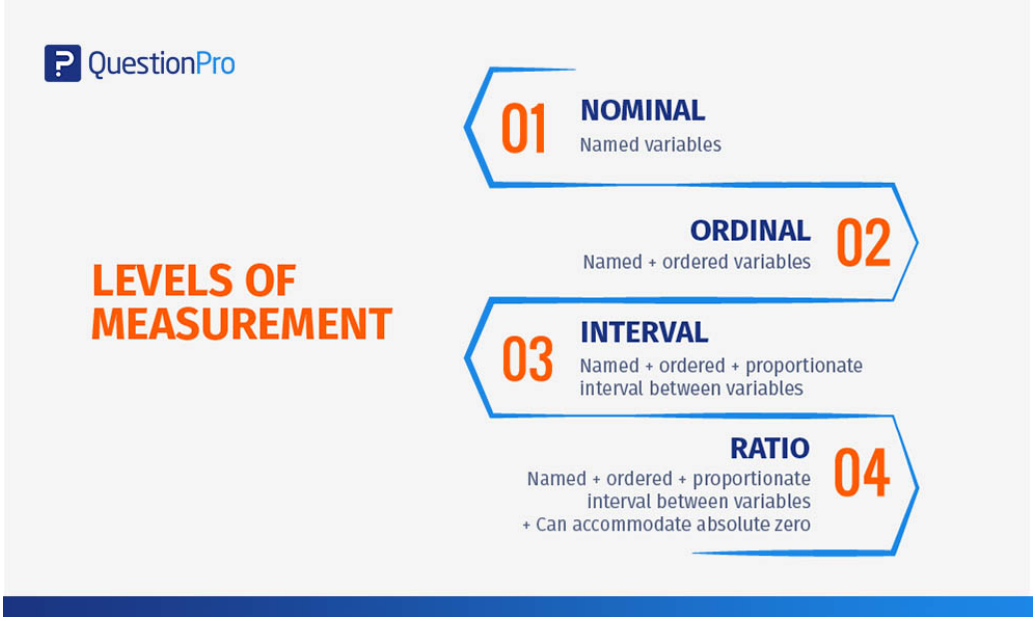
- Accept $H_0$: if $\mu_0$ lies **within** the confidence interval
- Rejection $H_0$: if $\mu_0$ lies **outside** the confidence interval

### 3.6.8. Other Tests

- Parametric Tests: eg: T-tests, F-test(comparing variance of 2 samples)
- Non-parametric Tests: eg: Wilcoxon signed-rank test(2 paired i.i.d. samples), Mann-Whitney-U test(2 independent i.i.d. samples)
- Test of Probability Distribution: Kolmogorov-Smirnov test, Chi-square test

# 4. Description of a Dataset

- **Dependent** and **independent** variables
- Scales of measurement of the variables




  - **Nominal**: **categorical** variable scale. A scale used for labeling variables into distinct classifications, **doesn't involve quantitative value or order.**
    - eg: Gender, City, Nationality, Jobs
  - **Ordinal**: scale to depict **order** of the variables for non-mathematical ideas. It maintains a **descriptional** quality and the **difference between variables can't be calculated**.
    - eg: Frequency(high, medium, low), Happiness, Satisfaction, Pain level, Cloth size (S, M, L), rank of Unis
  - **Interval**: numeric scale where **order** of the variables as well as the **difference between variables are known**. There exists **no true 0**. Variables can be added and subtracted, but **not multiplied or divided**.
    - eg: GPA, GRE, Celsius, Fahrenheit (20°C is 10°C higher than 10°C, it doesn't mean 2 times warmer.)
  - **Ratio**: numeric scale that's ordered, difference between variables known, and there **exists true 0**. Variables can be added, subtracted, multiplied and divided.
    - eg: weight, height, time, Kelvin temperature, money
- **Cross-sectional, time series, panel data** (see 7.1.5)

# Teil II.
# Regression Analysis

## 5. Definition & Terminology

Regressions identify **relationship between dependent and independent variables**.

- Association between dependent & independent variables,
- Impact of independent variables on dependent variables.
- Formulation of association/impact in functional form.
- Used for
  - descriptive analysis
  - numerical prediction
  - time series forcasting

### Terminology

- **measurement tuples**: data streams $(x_1, y_1), \ldots, (x_n, y_n)$
- **predictor (independent variable, feature, regressor, covariate)**: $x_i$
- **response (dependent variable, outcome)**: $y_i$
- **regression function**: $\eta(x) = E(y|x)$

## 6. Linear Regression

### 6.1. First Order Linear Model

$$Y = \beta_0 + \beta_1 X + \varepsilon$$

- Y: response variable (from measurement tuples)
- X: predictor variable (from measurement tuples)
- $\beta_0$: y-Axis intercept, **unknown, to be estimated**
- $\beta_1$: slope, **unknown, to be estimated**
- $\varepsilon$: **residual**, random error

## 6.2. Multiple Linear Regression Model

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_k X_k + \varepsilon$$

Formulation in **matrix form**:

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} = \begin{bmatrix} 1 & X_1 1 & X_1 2 & \dots & X_1 k \\ 1 & X_2 1 & X_2 2 & \dots & X_2 k \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & X_m 1 & X_m 2 & \dots & X_m k \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_k \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_m \end{bmatrix}$$

$$[m \times 1] = [m \times (k+1)] \cdot [(k+1) \times 1] + [m \times 1]$$

**OLS Estimator** for multiple linear regression model: minimize the RSS

- Model:

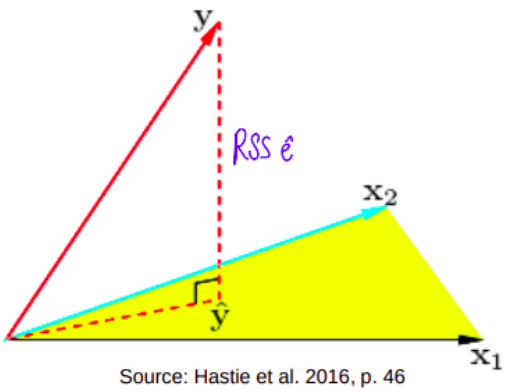
$$RSS = e^T e = (y - X\hat{\beta})^T (y - X\hat{\beta}) \to \min$$
$$\to \frac{\partial RSS}{\partial \beta} = -2X^T y + 2X^T X\beta = 0$$

- Solution:

$$\hat{\beta} = (X^T X)^{-1} X^T y$$

$$\hat{y} = X(X^T X)^{-1} X^T y$$

- Projection: minimizing RSS $\to \vec{e}$ is orthogonal to the subspace spanned by all independent variables



Source: Hastie et al. 2016, p. 46

## 6.3. Estimation of Coefficients: Ordinary Least Square Estimator

- input: data streams $(x_i, y_i)$
  model: linear regression $Y = \beta_0 + \beta_1 X + \varepsilon$
  output: $\hat{\beta}_0$, $\hat{\beta}_1$

- Goal of OLS estimator: **minimize the sum of squared residuals (RSS)**
  Outlier will have larger value input when residual squared.

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x$$

$$\min \Sigma_i e_i^2 = \min \Sigma_i (y - \hat{y})^2 = \min \Sigma_i (y - (\hat{\beta}_0 + \hat{\beta}_1 x))^2$$

## 6.4. Quality Metrics of the Model

To measure how well the model fits the data, you can analyze the quality model-wise or coefficient-wise.

- Model-wise:
  - Residual Sum of Squares (RSS)
  - Total Sum of Squares (TSS)
  - $R^2$
- Coefficient-wise:
  - Significance-Test on each coefficient in the model

### 6.4.1. Residual Sum of Squares (RSS)

An **unbiased** estimator of RSS of the population is given by

$$RSS = \Sigma_{i=1}^{n} (y_i - \hat{y}_i)^2$$

- RSS = 0: the model fits 100% of the data
- RSS $\neq$ 0: distance between true $y$ and $\hat{y}$ : residual $e$
- RSS $\downarrow$, model-fit quality $\uparrow$

### 6.4.2. Mean Squared Error (MSE)

$$MSE = \frac{RSS}{N}$$

### 6.4.3. Root Mean Squared Error (RMSE)

$$RMSE = \sqrt{MSE}$$

### 6.4.4. Total Sum of Squares (TSS)

**Total Sum of Squares(TSS)** is the sum of Explained Sum of Squares(ESS) and Residual Sum of Squares(RSS).

$$\Sigma (y - \bar{y})^2 = \Sigma (\hat{y} - \bar{y})^2 + \Sigma (y - \hat{y})^2$$
$$TSS = ESS + RSS$$

### 6.4.5. $R^2$

$R^2$ measures the proportion of the variation in y that is explained by the variation in x. → the **proportion of Explained Sum of Squares(ESS).**

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

- range of $R^2$: [0,1]
  - $R^2 = 0$: ESS = 0, RSS = $\infty$. No linear relationship between x and y.
  - $R^2 = 1$: ESS = TSS, RSS = 0. Perfect match between x and y.

### 6.4.6. Significance Test of the coefficients: T-Test

- Test the significance of the coefficients in alternative hypothesis($H_1$):

$$H_0 : \beta_1 = 0$$

$$H_1 : \beta_1 \neq 0$$

- Distribution: T-Distribution
- Degree of Freedom: if **error variable** is **normally distributed**, then

$$df = n - 2$$

- test statistic:

$$t = \frac{\hat{\beta}_1 - 0}{SE(\hat{\beta}_1)} = \frac{\hat{\beta}_1}{\sqrt{\frac{RSS}{\Sigma_{i=1}^{n}(x_i - \bar{x})^2} \cdot \frac{1}{n-2}}}$$

- Conclusion:

  two-sided test, reject $H_0$, if $|t| > t^c_{1-\frac{\alpha}{2}}$ or $p < \alpha$

### 6.4.7. Other Metrics

- Adjusted $R^2$: allows models **with different number of variables** to be compared
- F-statistic: indicates linear relationship between y and **at least one** of the xs
- T-test of each partial regression coefficient: significance of a single coefficient while controlling others

## 6.5. Model Specification

The process of developing a regression model. It's a repeated process. You need to try different combinations or test the significance of variables to find an optimal regression model.

- selection of an appropriate functional form (linear, quadratic, log-linear, interaction terms, etc.)
- choosing which variables to include (might include irrelevant or omit relevant variables)

### 6.5.1. Functional Form of Linear Model

- standard normal linear model(first-order/multiple)
- nominal variables(categorical): 0 or 1
- quadratic models: $y = \beta_0 + \beta_1 x_1 + \beta_2 x_2^2 \rightarrow z_2 = x_2^2$
- interaction terms: $y = \beta_0 + \beta_1 x_1 + \beta_2 x_1 x_2 \rightarrow z_2 = x_1 x_2$
- exponential terms into logarithm terms: $y = \alpha x^\beta \varepsilon \rightarrow \ln(y) = \ln(\alpha) + \beta \ln x + \ln(\varepsilon)$

### 6.5.2. Choosing Variables to Include

- Idea: The initial model might include large set of irrelevant variables or omit some relevant variables.
- Goal: find an optimal combination that explains variation in Y with **a small and meaningful predictors Xs** $\rightarrow$ feature selection
- Methods:
  - **Best Subset**:

    Test **all combinations** ($2^n$) and find out best subset
  - **Backward Elimination**: (top-down)

    Start with **full model with all variables**, test the significance(t-Test) of the variables.

    Consider the predictor with lowest t-statistic/highest p-value: remove the variable if $p > \alpha$ (can't reject $H_0$)
  - **Forward Selection**: (buttom-up)

    Start with **only one variable**, test the significance(t-Test) of the variable.

    Only consider the variable with highest t-statistic/lowest p-value: add the variable if $p < \alpha$ (reject $H_0$)
  - **Stepwise Regression**: combination of forward/backward selection.

## 6.6. Model Interpretation

Result of a model fitting can be retrieved by "summary(model)"

- Hypothesis:
  - Parameters:

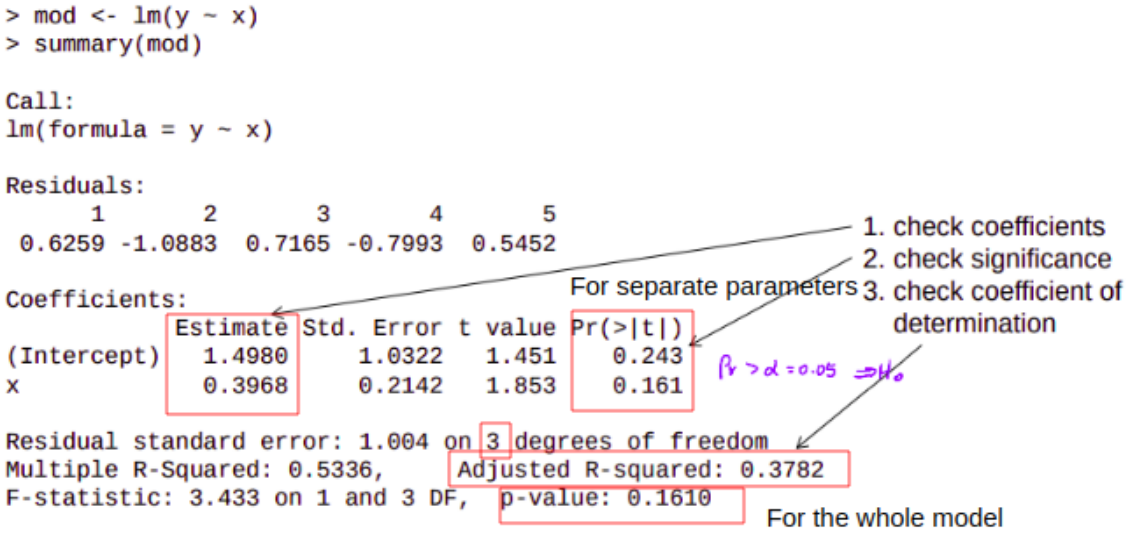
    $H_0$: $\beta_i = 0$

    $H_1$: $\beta_i \neq 0$
  - Whole Model: $H_0$: All predictors are not able to explain the model.

    $H_1$: The whole model is statistically significant.

- Conclusion:

  for an $\alpha = 0.05$ level, $p > \alpha, \rightarrow$ cannot reject $H_0$, the parameter x is not significant from 0.




### Interpretation

- **predictors, intercepts – coefficient & p-value**:
  - Significance: significant? at a significance level of $\alpha = ?$
  - $\beta_0$: intercept, the value when all predictors are 0.
  - $\beta_i$: slope, coefficient positive/negative influence of predictors on response?
  - **Transformed** predictors/response
- **whole model – F-statistic & p-value**: significant? at a significance level of $\alpha = ?$
- **Explanatory power of the model – Adjusted** $R^2$: high/low? why?

# Teil III.
# Regression Diagnostics

## 7. Gauss-Markov Theorem

The Gauss-Markov theorem states that in a linear regression model, where the errors

- have expectation 0
- are uncorrelated
- have equal variances

the **Best Linear Unbiased Estimator (BLUE)** of the coefficients is given by the **Ordinary Least Square (OLS)** estimator.

Properties of the BLUE:

- Unbiased: $E(\hat{\beta}) = \beta$
- Consistent: $n \uparrow$, $var(\hat{\beta}) \downarrow$
- Efficient: $var(\hat{\beta}) < var(\tilde{\beta})$, it gives the **lowest variance** compared to other linear unbiased estimators.

## 8. Gauss-Markov Assumptions/Requirements

The OLS estimator is the best linear unbiased estimator (BLUE), iff

| Property | What does it mean? | Why do we need that? | How can we test that? |
|---|---|---|---|
| Linearity | Regression linear in the coefficients $\beta$ | Core assumption of **linear** regression | Do not transform $\beta$, only the covariates |
| No Multicollinearity | • $rank(\mathbf{X}) = p$<br>• No high correlation between covariates | • Impossible to estimate coefficients<br>• Non-significant coefficients | Variance Inflation Factor |
| Homoskedasticity | $Var(\varepsilon_i \mid \boldsymbol{X}) = \sigma^2 \; \forall i$ | • Some observations have more „weight“<br>• Biased standard errors | • White Test<br>• Breusch-Pagan Test |
| No Autocorrelation | $\mathrm{Cov}(\varepsilon_i, \varepsilon_j) = 0 \;\; \forall i, j$ | • Omitted variables<br>• Functional misfit<br>• Measurement errors | Durbin-Watson Statistic |
| Exogeneity | $\mathrm{E}(\varepsilon_i \mid \boldsymbol{X}) = 0 \;\; \forall i$ | • Omitted variables<br>• Measurement errors | Instrument Variables |

## 8.1. Linearity

- Definition: Linear relationship in **coefficients** $\beta_i$.
- Why: core assumption of linear regression.
- Solution to non-linearity: Transform the **predictors or response**(logarithmic, interval-wise)
- Influence Factor: **outliers**.

  Reasons for outlier:
  - error in recording the value
  - point doesn't belong to the sample
  - no error, it's an valid observation

  Solution to outlier:
  - identify outliers → exclude
  - apply "robust" regression

## 8.2. No Multicollinearity

- Definition: **No** linear dependency between the **predictors** $X_i$

  → rank(X) = p (the data matrix has **full rank** = number of columns)

  → **No high correlation** between predictors (though full rank).
- Testing:
  - **Correlation Coefficient** between predictor **pairs**
  - **Variance Inflation Factor (VIF)**: correlation among **multiple predictors**

$$VIF = \frac{1}{1 - R_k^2}$$

**Interpretation of VIF** When the predictor $X_k$ is set as the dependent variable, $R_k^2$ of the variance in the predictor $X_k$ can be explained by the rest of other predictors.

eg: VIF = 10. $R_k^2 = 90\%$. 90% of the variance in the predictor $X_k$ can be explained by the rest of other predictors.

**Rule** VIF ↑, Multicollinearity ↑. Remove predictor $X_k$ if **VIF > 10**

- Consequences of Multicollinearity: Non-significance of the coefficients

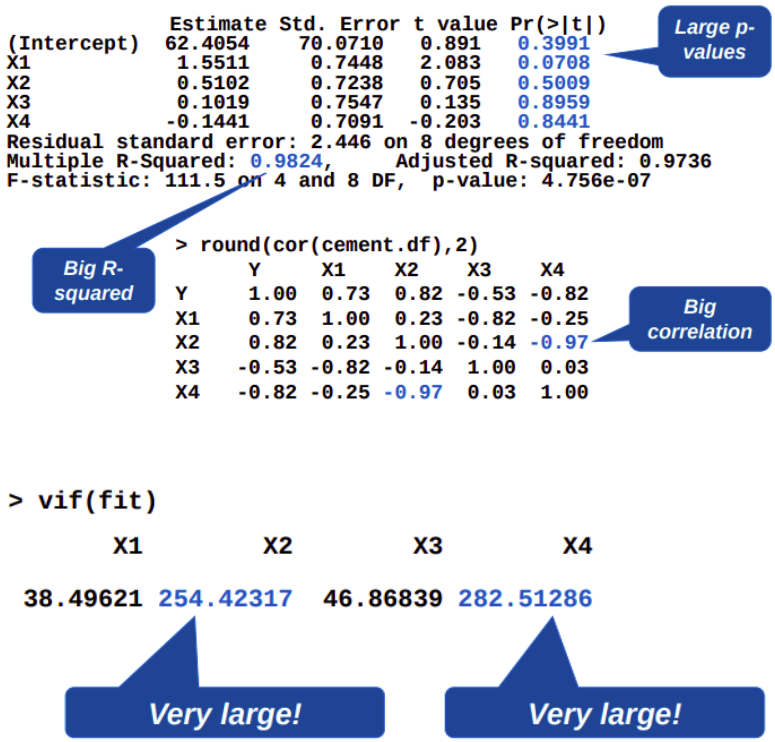
  The coefficient has a small t-value/large p-value (can't reject $H_0$)
  - small VIF: predictor $X_i$ is not related to response

    → remove the variable $X_i$

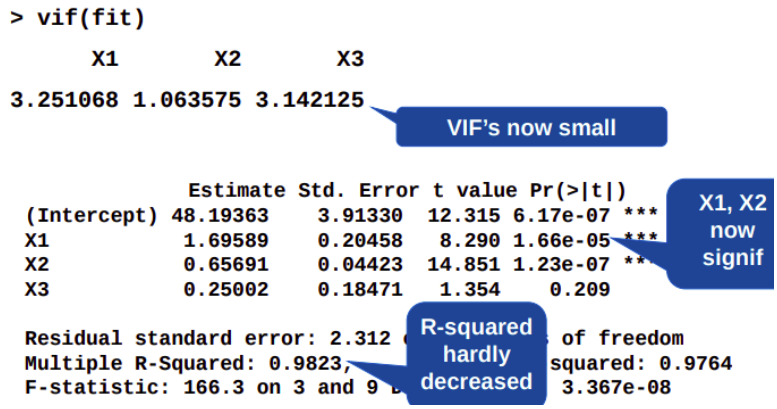
– large VIF: predictor $X_i$ is highly correlated to some other predictors.

→ **correlation matrix**: remove one of the highly correlated variables (near 1 or -1)

- Example R-Interpretation:




large p-value, large VIF → high correlation among predictors. Call correlation matrix. Find out and remove one of the highly correlated predictors.




## 8.3. Homoscedasticity

- Definition: **each residual** $\sigma_i$ of predictor $X_i$ exhibit **constant** variance.

→ the spread of residual in different predictors remains nearly the same.

→ no systematic development(grows larger/smaller) of residuals – **Heteroscedasticity**

- Testing:
  - **Breusch-Pagan Test**
  - **White Test**
    - $H_0$: all variances $\sigma_i$ are equal (homoscedasticity)
      $H_1$: heteroscedasticity
    - Distribution: $\chi^2$-Distribution
    - reject $H_0$ if $p < \alpha$
- Consequence of Heteroscedasticity:
  - estimated variance of coefficients $Var(\hat{\beta})$ is **biased**.
  - OLS Estimator no longer efficient.
  - Some predictors has more "weight" than others → higher sensitivity

## 8.4. No Autocorrelation

- Definition: no correlation between the $i^{th}$ and $j^{th}$ **overall residual**
  → $Cor(\varepsilon_i, \varepsilon_j) = 0$
  → **no pattern** of residuals should be observed over time, in case of **time series data**.
- Testing:
  The significance test of coefficients might say they are significant from 0. However, Autocorrelation detected.
  - visualize residuals against time
  - **Durbin-Watson statistic [0,4]**: test for first-order autocorrelation

$$DW = \frac{\Sigma_{i=2}^{n}(e_i - e_{i-1})^2}{\Sigma_{i=1}^{n} e_i^2}$$

    **Interpretation of DW-statistic**
    - DW = 2: **no** autocorrelation
      Rule of thumb: DW ∈ [1.5, 2.5] → no serial correlation
    - DW = 0: perfect **positive** autocorrelation
    - DW = 4: perfect **negative** autocorrelation
- Consequences for autocorrelation:
  - an important predictor is omitted (which explains the pattern over time)
  - functional misfit
  - measurement error in predictors
- Solution to autocorrelation: Model the missing predictor

- overall trend in time: t
- dummy variable for seasonal indexes $Q_1$, $Q_2$, $Q_3$

  number of dummy variables: **number of choices -1** → avoid multicollinearity

Example Model:

$$y = \beta_0 + \beta_1 \cdot t + \beta_2 \cdot Q_1 + \beta_3 \cdot Q_2 + \beta_4 \cdot Q_3$$

## 8.5. Exogeneity

- Definition: the expected value of **residual vector** given all predictors is 0.
  → $E(\varepsilon|X) = 0$, $Cov(\varepsilon, X) = 0$
- Consequences for Endogeneity(not exogene):
  - measurement error
  - predictors and response effect each other mutually
  - **important predictors are omitted** → bias in estimation of coefficients
- Testing for individual effects: **Lagrange Multiplier Test**, in R "plmtest(model)"
  - $H_0$: No individual effects
- Testing for fixed or random effect model when Lagrange Multiplier Test fails: **Hausman Test**
  - $H_0$: random effect estimator is consistent & efficient → **random effect model**

    $H_1$: **fixed effect model** needed.
- Solution to endogeneity due to omitted variable bias:
  according to types of data:

  **cross-section data** data observing many objects at the same time

  difficult to find out the confounding variables → **no solution**

  **panel data** repeated observations on the same objects over time. Mostly unbalanced panel data, where some individuals are not recorded in all time period.

  → individual-specific panel data structure

  → find out the omitted individual-specific effects on the response.

  Solution:
  - **Fixed Effects Model**: Individual/Entity-specific effects are **correlated** to other predictors

    → $\lambda_i$ is constant, can be seen as **an additional intercept** for each individual $i$ in regression model.

$$y_{it} = (\beta_0 + \lambda_i) + \beta_1 x_{1it} + \beta_2 x_{2it} + \cdots + \beta_k x_{kit} + \varepsilon_{it}$$

Estimators for the fixed effect models: first differences, within, least square dummy variable

– **Random Effects Model**: Individual/Entity-specific effects are **uncorrelated** to other predictors

$\rightarrow \lambda_i$ is drawn independently, can be seen as **an element of residual** for each individual in regression model.

$$y_{it} = \beta_0 + \beta_1 x_{1it} + \beta_2 x_{2it} + \cdots + \beta_k x_{kit} + \underbrace{(\lambda_i + u_{it})}_{\varepsilon_{it}}$$

# Teil IV. Classification Algorithms

**Classification**

- Input: a database $D = x_1, x_2, \ldots, x_n$ of tuples, a set of classes $C = C_1, C_2, \ldots, C_m$
- Output: a **mapping** $f : D \rightarrow C$, where each $x_i$ is assigned to a class.

## 9. Generalized Linear Models

### 9.1. Component of GLM

- Random component: identifies **dependent variable** $\mu$ and **its probability function**
- Systematc component: identifies the set of **explanatory variables – predictors** $(X_1, \ldots, X_k)$
- Link function: a **linear** function that links the dependent variable and all explanatory variables.

$$g(\mu) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_k X_k$$

### 9.2. Common Link Functions

- **Identity Link: linear regression**

$$g(\mu) = \mu = X\beta$$

- **Logit Link: logistic regression**

$$g(\mu) = \ln(\frac{\mu}{1-\mu}) = X\beta$$

- **Log Link: Poisson regression**

$$g(\mu) = \log(\mu) = X\beta$$

# 10. Logistic Regression: Binary Classification

- Idea:
  - Gauss-Markov assumptions need to be fulfilled to implement an OLS-Estimator
    → more **generalized models** needed to relax the assumptions for OLS
  - Predicting **categorical dependent variables**: a **classification** problem.
  - Limitation from linear regression in classification:
    * prediction values $\hat{y}$ should range within [0, 1], linear regression model prediction **exceeds this range**.
    * **violation of Homoscedasticity**: residuals $e_i$ doesn't have constant variance since the true Y only takes two values(0/1). The distribution of the residuals is no longer a normal distribution.
    * **violation of No Autocorrelation**: overall residuals of the model follows a systematic pattern, it's positive on one side and negative on the other side → Autocorrelation

## 10.1. The Logistic Regression Model

The **binary** logistic regression model described in **log odds/logit**:

$$\text{log odds} = \ln(\frac{p(X)}{1-p(X)})$$

$$\ln(\frac{p(X)}{1-p(X)}) = \beta_0 + \beta_1 X + \varepsilon$$

- Modeling Input: $X_i$, 0/1
  Output: a Logit-model
- p(X): probability that Y = 1 given X.
- **log odds / logit**: range $(-\infty, +\infty)$, the log-ratio of Y = 1 to Y = 0 given X

The logistic regression model described in **odds**:

$$odds = \frac{p(X)}{1-p(X)}$$

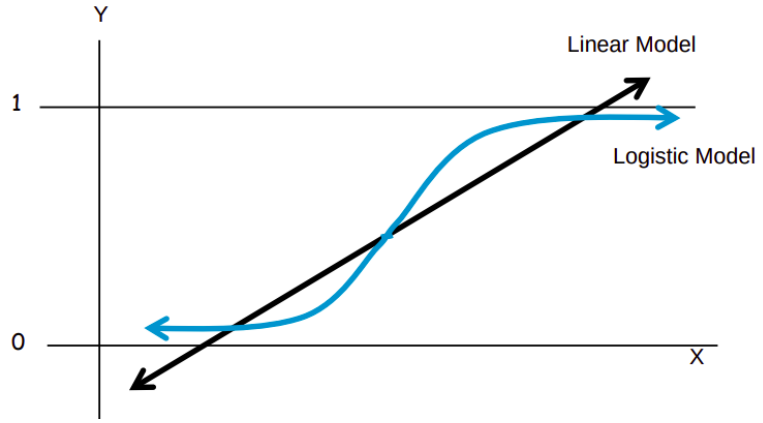
$$\frac{p(X)}{1-p(X)} = e^{\beta_0+\beta_1 X}$$

- **odds**: range $[0, +\infty)$, the ratio of Y = 1 to Y = 0 given X
  - p(X) = 0.5, odds = 1
  - p(X) < 0.5, → 0, odds → 0
  - p(X) > 0.5, → 1, odds → $+\infty$

## 10.2. The Logistic Function

$$Pr[Y|X] = p(X) = \frac{e^{\beta_0+\beta_1 X}}{1+e^{\beta_0+\beta_1 X}}$$

- Prediction Input: $X_i$, $\beta_i$
  **Output:** $p(X)$
- Range p(X): [0,1]
  - $\beta_0 + \beta_1 X = 0$: p(X) = 0.5
  - $\beta_0 + \beta_1 X \uparrow$: p(X) $\rightarrow$ 1
  - $\beta_0 + \beta_1 X \downarrow$: p(X) $\rightarrow$ 0
- $\beta_0$: regression constant, moves the curve **left/right**
- $\beta_1$: regression slope, defines **steepness** of the curve. $\beta_1 \uparrow$, steepness $\uparrow$
- This can be reformed into the logistic regression model.
- Comparison Linear Model & Logistic Model(the logistic function):




Comparison p(X) & odds & log-odds:

| $p(X)$ | $\frac{p(X)}{(1-p(X))}$ | Logit |
|---|---|---|
| 0 | 0 | -∞ |
| 0,1 | 0,11 | -2,20 |
| 0,2 | 0,25 | -1,39 |
| 0,3 | 0,43 | -0,85 |
| 0,4 | 0,67 | -0,41 |
| 0,5 | 1,00 | 0,00 |
| 0,6 | 1,50 | 0,41 |
| 0,7 | 2,33 | 0,85 |
| 0,8 | 4,00 | 1,39 |
| 0,9 | 9,00 | 2,20 |
| 1 | ∞ | ∞ |

## 10.3. Multiple Logistic Regression Model

$$\ln(\frac{p(X)}{1-p(X)}) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_k X_k$$

- Interpretation of Coefficients $\beta_j$:
  - **while keeping all other $x_j$ constant**, if $x_{ij}$ increase by 1, the **log-odds** will increase/decrease by $\beta_j$, or the **odds** will increase/decrease by $\mathbf{e}^{\beta_j}$.
- Test of Multicollinearity: VIF / correlation coefficient
- Test of irrelevant variables: Wald-Test (significance of coefficients)

## 10.4. Estimation of Coefficients: Maximum Likelihood Estimator

### 10.4.1. Intro

The probability of one data point $x_i$ can be modeled as **Bernoulli trial**:

$$p^{y_i}(1-p)^{1-y_i}$$

The **likelihood function**: the **joint probability** of observing the dependent variable values of random samples. $\rightarrow$ the product of all Bernoulli trials

$$L = \Pi_{i=1} p^{y_i}(1-p)^{1-y_i}$$

The logistic function can also be described as a **sigmoid function** in form $S(x) = \frac{e^x}{1+e^x} = \sigma(x)$

$$P(X) = \sigma(\beta_0 + \beta_1 X)$$

### 10.4.2. The Likelihood Function and Maximum Likelihood Estimator

The **likelihood function for Logistic Regression Model**:

$$\begin{aligned} L &= \Pi_{i=1} p^{y_i}(1-p)^{1-y_i} \\ &= \Pi_{i=1} \sigma(\beta_0 + \beta_1 X)^{y_i} \cdot (1 - \sigma(\beta_0 + \beta_1 X))^{1-y_i} \end{aligned}$$

The **Maximum Likelihood Estimator**: **maximizes the joint probability** of observing the set of dependent variables of the random samples.

Process:

- take the log:

$$LL = \ln(L) = \Sigma_{i=1}(y_i \ln(p) + (1-y_i)\ln(1-p))$$

- **maximize** LL:

$$\begin{aligned} \beta = \arg\max_{\beta}(LL) &= \arg\max_{\beta}[\Sigma_{i=1}(y_i \ln(p) + (1-y_i)\ln(1-p))] \\ &= \arg\max_{\beta}[\Sigma_{i=1}(y_i \ln(\sigma(\beta_0 + \beta_1 X)) + (1-y_i)\ln(1-\sigma(\beta_0 + \beta_1 X)))] \end{aligned}$$

- Method: **Gradient Ascent**

$k = 1$, feasible start point $\beta^{(1)} \in \mathbb{R}^n$, parameter $\varepsilon > 0$
While ( $\|\nabla L(\beta^{(k)})\| \geq \varepsilon$ ) {
- Choose step size $\alpha > 0$
- Set $\beta^{(k+1)} = \beta^{(k)} + \alpha^* \nabla L(\beta^{(k)})$
- $k++$

}

  - initial start point
  - select a step size $\alpha$
  - compute **partial derivatives** and **maximizes the function** $f(x^{(k)} + \alpha \bigtriangledown f(x^{(k)}))$

- **Gradient**(partial derivatives) of the LL-Function: **chain rule**
  with $z = \beta_0 + \beta_1 X$

$$\frac{\partial LL}{\beta_j} = \Sigma_{i=1} \frac{\partial LL}{\partial p} \cdot \frac{\partial p}{\partial z} \cdot \frac{\partial z}{\partial \beta_j}$$

$$\frac{\partial LL}{\partial p} = \frac{y_i}{p} - \frac{1 - y_i}{1 - p}$$
$$\frac{\partial p}{\partial z} = \sigma(z) \cdot (1 - \sigma(z))$$
$$\frac{\partial z}{\partial \beta_0} = 1, \frac{\partial z}{\partial \beta_j} = x_j$$

$$\frac{\partial LL}{\beta_0} = \left[\frac{y_i}{p} - \frac{1 - y_i}{1 - p}\right] \sigma(z) \cdot (1 - \sigma(z)) = [y_i - \sigma(X\beta)]$$
$$\frac{\partial LL}{\beta_j} = \left[\frac{y_i}{p} - \frac{1 - y_i}{1 - p}\right] \sigma(z) \cdot (1 - \sigma(z)) \cdot x_j = [y_i - \sigma(X\beta)]\, x_j$$

## 10.5. Quality Metrics of the Model

- **Null Model**: all predictors $x_i$ has no impact. The model is explained only by the intercept.
- **Fitted Model**: the model is explained by p predictors and 1 intercept.

### 10.5.1. Null Deviance

It measures how well the response is explained by **only the intercept (no predictors)**.

$$\text{null deviance} = -2\ln(L(null))$$

### 10.5.2. Residual Deviance

$$\text{residual deviance} = -2\ln(L(fitted))$$

- residual deviance ↓, model quality ↑
- difference between null and residual deviance ↑, model quality ↑

### 10.5.3. AIC

Additional penalizing term to get a balance between the **goodness of fit** and **simplicity of model**.

$$AIC = \text{residual deviance} + 2 \cdot \#\text{parameters in model}$$

- AIC ↓, model quality ↑

### 10.5.4. McFadden $R^2$

the ratio of improvement from null model to fitted model.

$$R^2_{McFadden} = 1 - \frac{LL(fitted)}{LL(null)} = 1 - \frac{\text{residual deviance}}{\text{null deviance}}$$

- model quality ↑ , LL(fitted) $\ll$ LL(null), $R^2 \to 1$
- model quality ↓ , LL(fitted) $\approx$ LL(null), $R^2 \to 0$
- rule of thumb: $> 0.2$ acceptable, $> 0.4$ ok.

### 10.5.5. Likelihood Ratio Test

Does the fitted model explain significantly more variance than null model?

- $H_0$: The fitted model explains **no more variance** than null model
  $H_1$: The fitted model explains **significantly more variance**.
- test statistic:
$$D = -2\ln(\frac{L(null)}{L(fitted)})$$
- Distribution: $\chi^2$-Distribution

### 10.5.6. Significance Test of Coefficients: Wald Test

- $H_0$: $\beta_i = 0$
  $H_1$: $\beta_i \neq 0$

### 10.5.7. Error Rates

comparing the predicted classification and the actual classification:

- percentage of correct Yes
- percentage of correct No
- overall percentage of correct predictions

## 10.6. Model Interpretation

### 10.6.1. R-Result Interpretation

```
> model <- glm( diabetes ~ glucose + mass + age, data = diabData, family = binomial)
> summary(model)

Call:
glm(formula = diabetes ~ glucose + mass + age, family = binomial,
    data = diabData)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.6030  -0.6666  -0.3815   0.6765   2.4804

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -9.677346   1.041873  -9.288  < 2e-16 ***
glucose      0.036266   0.004906   7.391 1.45e-13 ***
mass         0.077860   0.020120   3.870 0.000109 ***
age          0.054075   0.013236   4.085 4.40e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 498.10  on 391  degrees of freedom
Residual deviance: 354.37  on 388  degrees of freedom
AIC: 362.37

Number of Fisher Scoring iterations: 5
```

- significance of coefficients: significant if $p < \alpha$ in $\alpha$-level
- model: difference between null and residual deviance, AIC

### 10.6.2. Interpretation of Coefficients

Effect of change in $x_{ij}$ in **one unit**:

| $\beta_j$ | $ln(\frac{p}{1-p})$ (A) | $\frac{p}{1-p}$ (B) | $p$ (C) |
|---|---|---|---|
| $\beta_j > 0$ | increases by $\beta_j$ | increases by a factor of $e^{\beta_j}$ | Magnitude of increase unknown |
| $\beta_j < 0$ | decreases by $\beta_j$ | decreases by a factor of $e^{\beta_j}$ | Magnitude of decrease unknown |

- $\beta_j > 0$:
  - If $x_{ij}$ increase by 1, then the **log odds** will **increase** by $\beta_j$, or the **odds** will **increase** by $\mathbf{e}^{\beta_j}$.

  - If $x_{ij}$ increase by 1, then the **log odd ratio** is $\beta_j$, or the **odd ratio** is $e^{\beta_j}$.

- $\beta_j < 0$:
  - If $x_{ij}$ increase by 1, then the **change of log odds** will **decrease** by $\beta_j$, or the **change of odds** will **decrease by a factor** of $\mathbf{e^{\beta_j}}$.
  - If $x_{ij}$ increase by 1, then the **log odd ratio** is $\beta_j$, or the **odd ratio** is $e^{\beta_j}$.

# 11. Poisson Regression: Binary Classification for Count Data

- Idea:
  - **count variables (non-negative integers)** as dependent variables
  - limitation of linear regression / OLS-Estimator
    * linear model **predicts negative values**
    * count data is often **highly skewed**: #crimes committed – most are 0.

      → violates normality assumption (residuals follows normal distribution) of OLS-Estimator

- Assumption: observed count follows a **Poisson distribution**.

$$Pr(y|\mu) = \frac{e^{-\mu}\mu^y}{y!}$$

  - $\mu$: expected count and expected variance $E(Y) = Var(Y) = \mu$
  - y: observed count

- Limitation:
  - **Overdispersion**: $E(Y) = Var(Y) = \mu$ not met in real data.

    → underestimation of standard errors, potential overconfidence in result.

    → Alternative: negative binomial regression
  - **Zero-inflation**: highly skewed observed data/predictions in 0.

    this can't be changed even with negative binomial regression.

## 11.1. Poisson Regression Model

$$ln(\mu(x)) = \beta_0 + \beta_1 X_1 + \dots \beta_j X_i$$

## 11.2. Estimation of Coefficients: Maximum Likelihood Estimator

The random component(dependent variable) is:

$$Pr(Y|X) = p(X) = \frac{e^{-\mu}\mu^y}{y!} = \frac{e^{\beta xy}e^{-e^{\beta x}}}{y!}$$

The **likelihood function**:

$$L(\beta|X,Y) = \Pi_{i=1} p = \Pi_{i=1} \frac{e^{\beta x_i y_i} e^{-e^{\beta x}}}{y_i!}$$

The Maximum Likelihood Estimator:

$$\log L(\beta|X,Y) = \Sigma_{i=1}(\beta x_i y_i - e^{\beta x_i} - \log(y_i!))$$

- Method: **Gradient Ascent**

## 11.3. Model Interpretation

R-Result interpretation is same as logistic regression.

### 11.3.1. Interpretation of Coefficients

Effect of change in $X_{ij}$ in **one unit**:

| $\beta_j$ | $ln(\mu(x_i))$ (A) | $\mu(x_i)$ (B) |
|---|---|---|
| $\beta_j > 0$ | increases by $\beta_j$ | **increases by a factor of** $e^{\beta_j}$ |
| $\beta_j < 0$ | decreases by $\beta_j$ | **decreases by a factor of** $e^{\beta_j}$ |

- $\beta_j > 0$: If $x_{ij}$ increases by 1, then the **log-incidence rate** will **increase** by $\beta_j$, or the **incidence rate** will **increase by a factor** of $e^{\beta_j}$.
- $\beta_j < 0$: If $x_{ij}$ increases by 1, then the **log-incidence rate** will **decrease** by $\beta_j$, or the **incidence rate** will **decrease by a factor** of $e^{\beta_j}$.

# 12. Naive Bayes : Multinomial Classification with Independence

## 12.1. Naive Bayes Classifier

- it takes **all attributes/predictors** into account
- Assumptions:
  - all attributes are **equally important**
  - all attributes are **independent** → no correlation
- Difference Regression & Naive Bayes:
  - regression models the importance of different attributes (coefficents $\beta_i$)
  - attributes from regression can be correlated → VIF detection necessary

## 12.2. Bayes Theorem

**prior / unconditional probability $\mathbf{Pr(e)}$** the probability of a single event $e$

**posterior / conditional probability $\mathbf{Pr(e|h)}$** the probability of a single event $e$ given we know $h$

**probability distribution $\mathbf{Pr(E)}$** the probability distribution of the random variable $E$, with all possible values $e_i$

### Bayes Theorem

#### single evidence

- Input:
  - $Pr(e)$: prior probability of **single evidence** $e$ (eg: weather = windy)
  - $Pr(h)$: prior probability hypothesis (eg: play = true, test = positive)
  - $Pr(e|h)$: conditional probability of hypothesis
- Output: posterior conditional probability $Pr(h|e)$

$$Pr(h|e) = \frac{Pr(h \cap e)}{Pr(e)} = \frac{Pr(e|h) \cdot Pr(h)}{Pr(e)}$$

or

$$Pr(e|h) = \frac{Pr(h \cap e)}{Pr(h)} = \frac{Pr(h|e) \cdot Pr(e)}{Pr(h)}$$

- If prior probability $Pr(e)$ **unknown**: law of total probability

$$Pr(e) = Pr(e|h) \cdot Pr(h) + Pr(e|\neg h) \cdot Pr(\neg h)$$

$$Pr(h|e) = \frac{Pr(e|h) \cdot Pr(h)}{Pr(e)} = \frac{Pr(e|h) \cdot Pr(h)}{Pr(e|h) \cdot Pr(h) + Pr(e|\neg h) \cdot Pr(\neg h)}$$

#### multiple evidences

- Input:
  - $Pr(e_1, e_2, \ldots, e_k)$: prior probability of **multiple evidences** $e_i$
- Ouput: posterior conditional probability $Pr(h|e_1, e_2, \ldots, e_k)$

$$Pr(h|e_1, e_2, \ldots, e_k) = \frac{Pr(e_1, e_2, \ldots, e_k|h) \cdot Pr(h)}{Pr(e_1, e_2, \ldots, e_k)}$$

Since every attribute/evidence $e_i$ is **equally important & independent**:

$$Pr(h|e_1, e_2, \ldots, e_k) = \frac{Pr(e_1|h) \cdot Pr(e_2|h) \ldots Pr(e_k|h) \cdot Pr(h)}{Pr(e_1, e_2, \ldots, e_k)}$$
$$= \frac{\Pi_{i=1}^{k} Pr(e_i|h) \cdot Pr(h)}{Pr(e_1, e_2, \ldots e_k)}$$

- If the prior probability $Pr(e_i)$ is **known**:

$$Pr(e_1, e_2, \ldots, e_k) = Pr(e_1) \cdot Pr(e_2) \ldots Pr(e_k)$$

- If the prior probability $Pr(e_i)$ is **unknown**: law of total probability

$$Pr(e_1, e_2, \ldots, e_k) = Pr(e_1, e_2, \ldots, e_k|h) \cdot Pr(h) + Pr(e_1, e_2, \ldots, e_k|\neg h) \cdot Pr(\neg h)$$

## 12.3. Possible Problems in Prediction

### 12.3.1. Zero Frequency Problem in Dataset

- Definition: for the prediction of new instance ,there exists a **0-frequency** of attribute values **from the instance attribute**.

  eg: predict whether to play when Outlook = overcast: Pr(Outlook = overcast |¬ play)=0

  predict whether to play when Outlook = sunny: **no** zero-frequency problem here.

- Solution:
  - add 1 to the numerator for **every attribute value-class combination**.

    → the prior probability of the result class $Pr(h), Pr(\neg h)$ **remain the same** though adding 1.

    → for small data, significant bias possible
  - assign equal/unequal weights to the numerator, as long as $\Sigma w_i = 1$

### 12.3.2. Missing Value in New Instance

- Definition: there exists **missing values** for the attributes in the **new instance** for prediction.

| Outlook | Temp. | Humidity | Windy | Play |
|---|---|---|---|---|
| ? | Cool | High | True | ? |

- Solution: **omit** the attribute with missing value in prediction calculation.

  → take the **maximum** of $Pr(\text{play}|e_2, e_3, e_4)$ and $Pr(\neg\text{play}|e_2, e_3, e_4)$

### 12.3.3. Numeric Attributes in Dataset

- Definition: instead of nominal attributes (eg: Outlook = sunny, overcast, cloudy), **attribute is numeric** (eg: Temperature = 87, 90)

#### Assumption: Attribute Follows Normal Distribution

- Solution:
  - Assumption: numeric attributes follows **normal distribution** $e_i \sim N(\mu, \sigma^2)$

$$f(x) = \frac{1}{\sqrt{2\pi} \cdot \sigma} \cdot e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

  - calculate the **mean** and **standard deviation** for **each result class**.
  - conditional probability: **insert** the numeric instance into the **probability density function** f(x).

#### Assumption: Attribute Follows Unknown Distribution

- If the numeric data follows a **unknown distribution f(x)** (normal distribution not applied)

  → probability density distribution **estimation**.

- Solution: kernel density estimation
  - Estimator: Rosenblatt-Parzen Kernel-Density Estimator
  - f(x) is not a normal distribution, but each sample follows a normal distribution.

    → f(x) is the **sum** of normal distribution at each data point $x_i$

## 12.4. Prediction using 0-Rule & 1-Rule

- Input: a dataset $D = x_1, x_2, \ldots, x_n$ tuples (evidence, result class), a set of classes $C = C_1, C_2, \ldots, C_m$, a **new instance** with evidence $e_1, \ldots, e_k$
- Output: classification prediction result

### 12.4.1. 0-Rule

Process:

- count **absolute frequency** for each class (classification labels)
- Prediction result: class with **maximum** absolute frequency

### 12.4.2. 1-Rule

Process:

- build **frequency tables** for each evidence/attribute $e_i$ → absolute frequency
- pick the **most frequent class** as classification result for **each attribute value**

- calculate **overall error rate of the evidence/attribute** according to the classification result

| Outlook | Play = yes | Play = no | Error |
|---|---|---|---|
| Sunny | 2 | 3 | 2/5 |
| Overcast | 4 | 0 | 0/4 |
| Rainy | 3 | 2 | 2/5 |
| **Σ** | **9** | **5** | **(2+0+2)/(5+4+5)=4/14** |

- Prediction result: choose a **single attribute** with **smallest overall error rate**, pick the **most frequent class** of that evidence/attribute value.

Evaluation:

- uses only a **single** attribute for the classification
- no prediction result possible if missing value for the attribute found in new instance.
- if numeric values in dataset, discretization of the numeric values though possible, but increase the class complexity.

## 12.5. Prediction using Bayes Theorem: Maximum A Posteriori Classification

Process:

- **sort** the hypothesis/classification labels, get **prior probability of hypothesis** $Pr(h), Pr(\neg h)$
- build **frequency tables** for each evidence/attribute $e_i \rightarrow$ **absolute frequency**

| *Outlook* | No | Yes |
|---|---|---|
| Sunny | 3 | 2 |
| Overcast | 0 | 4 |
| Rainy | 2 | 3 |

- check if there is **zero frequency problem** for the **attributes from new instance**, resolve by **adding 1**.

  check for **numeric** attributes, calculate the **mean** and **standard deviation** for each result class.

- build **likelihood tables** for each evidence/attribute $e_i \rightarrow$ **relative frequency** $Pr(e_i|h)$

| Outlook | No | Yes |
|---|---|---|
| Sunny | 3/5 | 2/9 |
| Overcast | 0/5 | 4/9 |
| Rainy | 2/5 | 3/9 |

- find $\Pi_{i=1}^{k} Pr(e_i|h) \cdot Pr(h)$ and $\Pi_{i=1}^{k} Pr(e_i|\neg h) \cdot Pr(\neg h)$, **omit** the attribute if **missing value** in instance.
- **normalize** the result:

$$Pr(h|e_1, \ldots, e_k) = \frac{\Pi_{i=1}^{k} Pr(e_i|h) \cdot Pr(h)}{Pr(e_1, \ldots, e_k)}$$

$$Pr(\neg h|e_1, \ldots, e_k) = \frac{\Pi_{i=1}^{k} Pr(e_i|\neg h) \cdot Pr(\neg h)}{Pr(e_1, \ldots, e_k)}$$

with

$$Pr(e_1, \ldots, e_k) = \Pi_{i=1}^{k} Pr(e_i|h) \cdot Pr(h) + \Pi_{i=1}^{k} Pr(e_i|\neg h) \cdot Pr(\neg h)$$

- Prediction result: take the **maximum**.

$$\text{result} = \max \{Pr(h|e_1, \ldots, e_k), Pr(\neg h|e_1, \ldots, e_k)\}$$

## 12.6. Evaluation of Naive Bayes

- Complexity:
  - calculation of conditional probability: $\mathcal{O}(n)$,
    n: number of instances
  - calculation of class: $\mathcal{O}(c \cdot p)$,
    c: number of classes, p: number of attributes
- Advantages:
  - multinomial classification
  - works well, even if independence assumption is sometimes violated.
- Disadvantages:
  - takes all attributes with equal weight, could be **redundant**.
  - many numeric attributes are actually **not normally distributed**.

# 13. Bayesian Network: Multinomial Classification with Denpendency

- Idea:
  - Naive Bayes assumption too restrictive: **all** attributes are conditionally independent and equally important.
  - Attributes are often **correlated/dependent with each other**
  - Some attributes are **redundant** to the classification result.

  → conditional independence among **subset** of attributes.

## 13.1. Representation of Bayesian Network: Directed Acyclic Graph

- nodes: attributes
- edges: end node is dependent on start node / start node has direct influence on end node.
  - start: trigger/cause, evidence node
  - end: result/effect.


Abbildung 1: DAG - Naive Bayes(left) vs. Bayesian Network(right)

## 13.2. Probability Law in Bayesian Network

### 13.2.1. Chain Rule

According to the **directed acyclic graph**, derive the **joint probability distribution**

$$Pr(e_1, e_2, \ldots, e_k) = \Pi_{i=1} Pr(e_i|e_{i-1}, \ldots, e_1) = \Pi_{i=1} Pr(e_i|\text{Parents}(e_i))$$

eg: $Pr(A, B, C, D, E) = Pr(A) \cdot Pr(B) \cdot Pr(C|A, B) \cdot Pr(D|A, B, C) \cdot Pr(E|A, C, D)$

### 13.2.2. Conditional Independence

**conditional independence between hypothesis and evidence** the hypothesis $h$ is only dependent on $e_1, e_2, e_3$, not on $e_4$ (redundant), then

$$Pr(h|e_1, e_2, e_3, e_4) = Pr(h|e_1, e_2, e_3)$$

**conditional independence between hypotheses** if two hypotheses are **independent** from each other, then

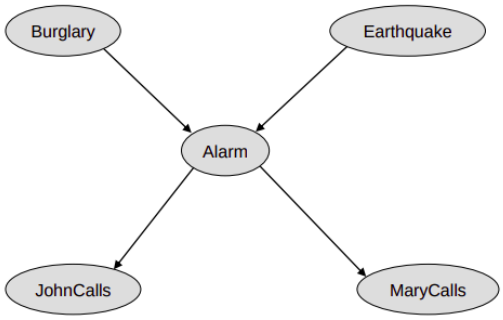
$$Pr(h_1, h_2|e_1, e_2) = Pr(h_1|e_1, e_2) \cdot Pr(h_2|e_1, e_2)$$

## 13.3. Inference in Bayesian Networks

- Idea: infer the probability of an event, given only observation of **a subset** of other attributes.

  → explain the **away effect** of attributes.

- Inference Rules: using **d-separation**




  Example:

  - alarm **not observed**: Burglary & Mary-calls **dependent**

    → if B, belief M ↑. if M, belief B ↑.

  - alarm **observed**: Burglary & Mary-calls **conditionally independent**.

    → no alarm. if Mary-calls, belief B −. if B, belief M −.

## 13.4. Evaluation of a Bayesian Network

- Quality Metrics:
  - To maximize the joint probability of training data, the Log-Likelihood of the training data.
  - Akaike Information Criterion (AIC)
- Advantages:
  - can handle dependencies among the attributes
- Disadvantages:
  - computationally expensive, given whether the network structure (DAG) is given, whether the attributes are observable.

# 14. Decision Tree Classifiers

- Idea of a Tree:
  - easy to read & interpret
  - robust, though lack of solid theoretical/statistical foundations
  - can use with Ensemble-Methods

## 14.1. Setup of a Decision Tree

- internal node: test on a **attribute**
- branch: outcome of the test (eg: true/false, red/green)
- leaf node: the **classification label/result**

Building an **Optimal** Decision Tree:

- Search Space: $2^{2^m}$ possible trees (m: # attributes, 2 result classes)
- Complexity: **NP-complete**
- Solution: **Greedy Algorithm** in **top-down** approach
  - All training data at the **root**.
  - Partition data **recursively** by choosing **one attribute** at each level.
  - Each split is assessed with a **measure**
  - Attribute with **best split** is chosen.
  - Repeat until all **leaf nodes** are pure (Not all attributes are necessary).

## 14.2. Quality Metrics of a Splitting Attribute

- Idea:
  - the path to classification **as easy as possible**. → **smallest** tree
  - good separation of classes → leaf nodes gives **one single class** → direct decision
  - the separation shouldn't affect class distribution.
- Evaluation function:
  - **information gain (ID3/C4.5)**
  - **information gain ratio**
  - **gini index (CART)**

### 14.2.1. Information Gain

- Idea: choose the attribute that result in **smallest tree** → **purest** nodes (one class)
  → choose the attribute with **greatest information gain**
  → information gain ↑, subset average purity ↑
- Parameters:
  - $c_i$: the **absolute frequency** of the training examples in the class $i$
  - $C$: the **total number** of training example at the **current stage/attribute value**.
  - $p_i$: the **relative frequency** of class $i$,

$$p_i = \frac{c_i}{C}$$

  - $N$: the **total number of training data**

**Process:**

① calculate **initial information** before any splits.

② calculate **information for each attribute value** using entropy

**Entropy** $\in [0, 1]$, measures how much **additional information required** in **bits**

$$\text{entropy}(p_1, \ldots, p_n) = -\Sigma_{i=1}^{n} p_i \cdot \log_2 p_i$$

- entropy = 0: pure
- entropy = 1: maximum impurity (for boolean)

**Information of Each Attribute Value**

$$\text{info}([c_1, \ldots, c_n]) = \text{entropy}(\frac{c_1}{C}, \ldots, \frac{c_n}{C})$$

③ calculate **information of the attribute**

**Information of the Attribute** the **weighted average** of the **information needed** from each attribute value.

Say an attribute has $m$ attribute values/branches,

$$\text{info}([c_1, \ldots, c_n]_1, \ldots, [c_1, \ldots, c_n]_m) = \Sigma_{i=1}^{m} \frac{C_m}{N} \cdot \text{info}([c_1, \ldots, c_n])_m$$

④ calculate the **information gain**

**Information Gain of the Attribute**

Information_Gain(attribute) = info(before split by attribute)−info(after split by attribute)

(5) choose the attribute with **maximum** information gain.

(6) continue to split.

**Attention**: the info **before** the split is the **info(attribute value)**, the information gain of the attribute changes to

eg: gain(Temperature) = **info(Outlook = sunny)** - info(high, mild, cool)

**Limitations**

- **biased** against **highly-branching** attributes (eg: IDs)
  → overfitting
  → Alternative: **Gain Ratio**

### 14.2.2. Gain Ratio

- Idea: modification of information gain, reduce bias on highly-branching attributes.
  → considers **number and size** of branches → **intrinsic information** of attribute
- **Intrinsic Information**, s: size of a leaf from each branch

$$\text{intrinsic\_info}([s_1, \ldots, s_n]) = \text{info}([s_1, \ldots, s_n])$$

  eg: 14 IDs, intrinsic_info([1,1,...,1]) = info([1,1,...,1]) = $14 \cdot (-\frac{1}{14} \cdot \log_2 \frac{1}{14}) = 3.807$ bits
- **Gain Ratio**:

$$\text{Gain\_Ratio(attribute)} = \frac{\text{Gain(attribute)}}{\text{Intrinsic\_Info(attribute)}}$$

- Process:
  - calculate the **information gain** of the attribute
  - calculate the **intrinsic information** of the attribute
  - calculate the **gain ratio**
  - choose the attribute that has **maximium** the gain ratio.

### 14.2.3. Gini Index

- Use-case: in Classification and Regression Tree (CART)
- Solution: select the split that **decreases** the Gini Index **the most**.
- **Gini Index**:

$$Gini(S) = 1 - P^2 - N^2$$
$$P = \frac{p}{p+n}, N = \frac{n}{p+n}$$

- a dataset S is split into $S_1, S_2$,

$$Gini_{split}(S_1, S_2) = \frac{p_1 + n_1}{p+n} \cdot Gini(S_1) + \frac{p_2 + n_2}{p+n} \cdot Gini(S_2)$$

  $\rightarrow$ select the attribute with **lowest** Gini-Index after split.

## 14.3. Evaluation of Decision Tree Algorithm

- Time Complexity: $\mathcal{O}(m \cdot n \log n)$
  (m: # attributes, n: # instances)
- Scalability for large data:
  - number of attributes ↑, tree size ↑, computation time ↑.
  - number of data instances ↑, memory ↑

## 14.4. Possible Problems in Prediction

### 14.4.1. Numeric Attributes in Dataset

- Solution: **binary split**
- Process:
  - an **initial split point** is either given or the middle of the sorted numeric values.
  - values are separated into 2 sections: **below($<$)** and **above($\geq$)** the split point.
  - calculate **information gain**
  - repeat **binary split**, choose the split with **maiximum information gain**.

### 14.4.2. Missing Values in Dataset

- Possible solutions:
  - **ignore** the instance/attribute with missing values.
  - treat missing value as **another nominal value**.
  - **estimate** missing values (regression, imputation)

- **follow the leader**: if an instance has missing attribute value, follow the the **branch with most instances**.
- **partition** the instance: send down instance **proportionally** to the number of instances.

  → classification result is **weighted**.

### 14.4.3. Overfitting of the Decision Tree

- Consequences in Overfitting:
  - decision tree to **large & complex**
  - **low bias** on training set, **high variance** on test set.
  - poor generalization to new data.
- Solution: Pruning

## 14.5. Pruning of Decision Trees

### 14.5.1. Prepruning

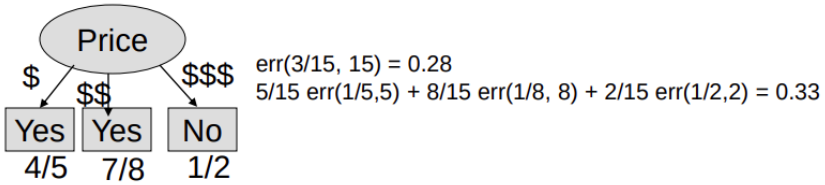
Process:

- define a **threshold** when to stop creating subtrees. This should be the same measure as determining attributes (eg: information gain)
- **stop** if the measure **no longer exceeds threshold**. (eg: infomation gain)
- leaf node: the **most frequent class**

Difficult to achieve high performance in practice.

### 14.5.2. Postpruning

- Process:
  - construct a **complete** decision tree.
  - prune back by **subtree replacement**, replacing the subtree with **a single leaf node**.
  - prune back criteria: **error rate** estimate for the **node** < **combined error rates** of the **children(weighted average)**




  → error rate of the node = 0.28 < combined error rate of the children = 0.33

  → prune back, the leaf node is the **most frequent class**.

- Use-case: C4.5, CART, however, computationally expensive
- Data for pruning:
  - hold-out set: an **independent** dataset from the training data. Best, but not practical when data is scarce.
  - training data: Used in C4.5, derive **outer bound of confidence interval** from data, use a **heuristic limit** for error rate. If the **error rate is outside the confidence interval** $\rightarrow$ **prune** back.

    $\rightarrow$ confidence limit c $\downarrow$ (25% -> 10%), the tree prunes **stronger**.

# 15. Ensemble Methods: Regression & Multinomial Classification

Ensemble methods can be applied widely, both in **regression** and **classification**. We have a slight focus of classification here.

- Definition: **combination of multiple models**. Build and output different "expert" models, let them vote for decision.
- Comparison over a **single model**:
  - a combination of several bad models sometimes achieves better result than a single good model. It ensembles different models with **low bias and high variance** and may
    - reduce overall variance
    - increase predictive performances
    - decrease expected error (bias + variance)
  - ensemble models tend to be **more stable**, a small change in input data doesn't necessarily change the final prediction.
- Types of Ensemble methods:
  - Bagging
  - Random Forest
  - Boosting
  - Stacking

## 15.1. Bagging

Bagging randomizes the **data** in training.

- idea: reduces **variance** of **low-bias models**.
- Training Models (Training)

  - randomly sample **m training subsets** of **size n** from the whole training data. (sample **with replacement**)
  - train one model for **each training subset independently**

- Classifying Instances (Testing)
  - each trained model predicts the test data independently **with equal weight**.
  - classification result: the **most frequent class**

- Advantages:
  - applied both in numeric prediction and classification
  - works well when data is **noisy**
  - improves performance if the learning scheme is **unstable** (eg: decision tree)
  - can be **parallelized**.

## 15.2. Random Forest

Random Forest **randomizes both data and feature selection**!!

- Definition/Process: **aggregates** full grown trees with **low bias but high variance**.

  → reduce the variance of the final predictor by **aggregating** all trees.

- Training Models:
  - define the $n$ number of trees, and the $m$ number of attributes to try.
  - for each tree, draw a **bootstrap sample of data** (random sample with replacement), **randomly select** $m$ **attributes**.
  - train each tree with their selected attributes **independently**.

- Classifying Instances:
  - each trained model predicts the test data independently (default: with equal weight).
  - classification result: aggregate the tree, result is the **most frequent class**.
  - **Weighting possible**.

## 15.3. Boosting

- Idea: combines **weak learners** into a **strong learner**.

  → reduces **bias** of **low-variance models**.

- Training Models (example: Adaboost):
  - Initialization: all training instances have **equal weight**.
  - First model: is trained and predicts **the training data instances** back.
  - Evaluation of prediction: **correct** prediction gets **lower** weights, **false** prediction gets **higher** weights.

  - **based on the last model**, repeat step 2-3. Until $m$ models are trained, always focus on traing data **with high weights** – misclassified instances.

- Classifying Instances:
  - each model is assigned **weight according to error rate** from training.
  - each trained model predicts the test data independently.
  - classification result: **weighted average** of classes.

- Algorithms: AdaBoost(weighting instances), XGBoost(trained on residual errors), etc.

## 15.4. Bagging VS. Boosting

| | Bagging | Boosting |
|---|---|---|
| Sequent | Two-step | Sequential |
| Partitioning data into subsets | Random | Give misclassified cases a heavier weight |
| Sampling method | Sampling with replacement | Sampling without replacement |
| Relations between models | Parallel ensemble: Each model is independent | Previous models inform subsequent models |
| Goal to achieve | Minimize variance | Minimize bias, improve predictive power |
| Method to combine models | Weighted average or majority vote | Majority vote |

## 15.5. Stacking & Meta-Learning

- Idea: different level models will be stacked on, predictions from **different classifiers(level-0 models)** are used as input into a **meta-learner(level-1 model)**.

- Training Models – level 0:
  - split the training data into **training subset** and **holdout subset**.
  - $m$ different models(NB, DT, etc.) are trained **independently** on the training subset.

    $\rightarrow$ level-0 classifiers

- Classifying Instances – level 0:
  - the level-0 classifiers predicts on **holdout subset**
  - the holdout set contains **only the prediction results of all level-0 classifiers**.

- Training Models – level 1:

  - training holdout serves as **training data** for a **single** level-1 model (normally simple, eg:regression).

- Classifying Instances – level 1:
  - classify the test data with the level-1 model.

# 16. Neural Networks: Regression & Multinomial Classification

supervised learning.

## 16.1. Terminology

**Training Example** has form $(x_n, y_n)$, with $x_n$ as input vector, $y_n$ expected/true output vector.

**Loss/Cost Function** maps values of one or more variables onto a number representing **loss/cost**.
Learning → minimizing a loss function.

**Risk Function** **expectation** of the **loss function**. In neural network, we minimizes our risk by **minimizing the empirical risk function – average loss of all training examples**.

**Activation Function**

- linear activation
- sigmoid activation: **focus of this lecture**, $\sigma(x) = \frac{e^x}{1+e^x}$
- Perceptron activation
- ReLU activation

**Epoch** one **forward pass** and one **backward pass** of **all** training examples. One pass = forward + backward pass.

- Forward Pass: calculate the output of **all training** through the neural network.
- Backward Pass: Backpropagation

**Backpropagation** calculate a **gradient** that is needed in **calculation of weights** to be used in network. It describes how a **single training example** starting from **output neurons** determines the goal for the neurons on the next layer and **steps backwards recursively**.

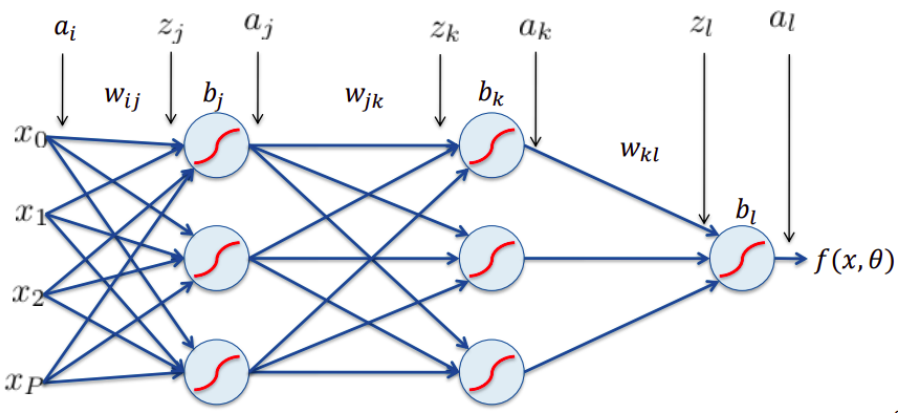
**Multi-Layer Feed-Forward Networks** represent arbitrary **non-linear** functions. It consists of

- input layer
- hidden layer with **activation**
- output layer with **activation**

**Weights and biases** need to be adapted in the neural network.
→ updated by **backpropagation (gradient descent)**.

## 16.2. Multi-Layer Feed-Forward Network

### 16.2.1. Setup of a Neural Network




### 16.2.2. Process

Goal of training: **minimizes loss function, minimizes empirical risk**.

Assume a network with 1 hidden layer, 1 output layer:

① **Forward Pass**: from input layer to output layer (with **sigmoid** activation).

If we are asked to perform, this calculation can done **matrix-wise**!! No need to separate each observation.

$$
\begin{aligned}
z^{[1]} &= W^{[1]} \cdot a^{[0]} + b^{[1]} = W^{[1]} \cdot x + b^{[1]} \\
a^{[1]} &= g^{[1]}(z^{[1]}) = \sigma(z^{[1]}) \\
z^{[2]} &= W^{[2]} \cdot a^{[1]} + b^{[2]} \\
a^{[2]} &= g^{[2]}(z^{[2]}) = \sigma(z^{[2]})
\end{aligned}
$$

② **Loss Function**: calculate the loss of the network output according to the loss function $l(y, \hat{y})$

If we evaluate the model using **cross-entropy loss**: the calculation for $y \ln \hat{y}$ is a dot product(element-wise multiplication).

$$l(y, \hat{y}) = -[y \ln \hat{y} + (1 - y) \ln(1 - \hat{y})]$$

③ **Empirical Risk**: calculate the empirical risk $\mathcal{L}$ by **averaging** the loss.

$$\mathcal{L}(y, \hat{y}) = \frac{1}{n} \cdot \Sigma l(y, \hat{y})$$

④ **Backpropagation**: readapt the **weights and biases** using **gradient descent**.
example in updating layer 2:

$$W_{t+1}^{[2]} = W_t^{[2]} - \alpha \cdot dW = W_t^{[2]} - \alpha \cdot \frac{\partial L}{\partial W^{[2]}}$$
$$b_{t+1}^{[2]} = b_t^{[2]} - \alpha \cdot db = b_t^{[2]} - \alpha \cdot \frac{\partial L}{\partial b^{[2]}}$$

- $\frac{\partial L}{\partial W^{[2]}}, \frac{\partial L}{\partial b^{[2]}}$: chain rule

$$\frac{\partial L_n}{\partial W} = \frac{\partial L_n}{\partial a_n} \cdot \frac{\partial a_n}{\partial z_n} \cdot \frac{\partial z_n}{\partial W}$$

$$L_n = \frac{1}{2}(y_n - g(w_{kl} a_{kn} + b_l))^2 = \frac{1}{2}(y_n - a_{ln})^2, \qquad \frac{\partial L_n}{\partial a_{ln}} = -(y_n - a_{ln})$$
$$a_{ln} = g(z_{ln}), \qquad \frac{\partial L_n}{\partial z_{ln}} = g'(z_{ln})$$
$$z_{ln} = w_{kl} a_{kn} + b_l, \qquad \frac{\partial L_n}{\partial w_{ln}} = a_{kn}$$

If the activation is a sigmoid activation: $\sigma(x) = \frac{e^x}{1+e^x}$, $\sigma'(x) = \sigma(x)(1 - \sigma(x))$.

$$dW^{[2]} = -(y - a^{[2]}) \cdot a^{[1]^T} = (a^{[2]} - y) \cdot a^{[1]^T}$$

### 16.2.3. Trainable Parameters

Number of trainable parameters: the **free parameters** of the neural network are **weights and biases**– $W^{[1]}, b^{[1]}, W^{[2]}, b^{[2]}, \ldots$.
→ define the **dimension** of these parameters, add up all possible trainable elements.
(eg: $W^{[1]}$ is 2x2-matrix, therefore 4 trainable parameters)

## 16.3. Gradient Descent for Backpropagation

- Goal: given any function f, find $x^* = \arg\min_x f(x)$
- **Gradient** at position x is defined as the **partial derivative**:

$$\nabla f(x) = \begin{bmatrix} \frac{\partial f(x)}{x_1} \\ \vdots \\ \frac{\partial f(x)}{x_d} \end{bmatrix}$$

– Interpretation: in d-dimensional space, gradient points in **direction of steepest ascent** of f at point x.

→ to **minimize loss function** → **descent**, the opposite direction $-\nabla \mathbf{f}(\mathbf{x})$.

### 16.3.1. General Process

① choose an **initial point**

② choose a **step size (either fixed or dynamic)**.

③ take a step in the **direction opposite the gradient**.

- fixed step size:

$$\begin{bmatrix} x_n \\ y_n \end{bmatrix} = \begin{bmatrix} x_{n-1} \\ y_{n-1} \end{bmatrix} - \alpha \cdot \nabla f(x_{n-1}, y_{n-1})$$

- dynamic step size:

$$\begin{bmatrix} x_n \\ y_n \end{bmatrix} = \begin{bmatrix} x_{n-1} \\ y_{n-1} \end{bmatrix} - \alpha_n \cdot \nabla f(x_{n-1}, y_{n-1})$$

④ repeat till convergence.

Convergence to optimum: depends on the **step size**.

- too small: would converge eventually, but takes long time.
- too large: value **oscillates**, doesn't converge.
- would stall if $\nabla f(x) = 0$.
- can stuck at saddle point.

Criteria:

- function f is **convex**
- step size $\alpha$ is square summable, but not summable.

→ Alternative: introduce **momentum**.

### 16.3.2. Process with Momentum Introduced

- Idea: uses an exponential averaging of gradients to **make sudden changes in direction less likely**.

$$d_n = \beta \cdot d_{n-1} + \alpha \cdot \nabla f(x_{n-1})$$
$$x_n = x_{n-1} - d_n$$

# 17. Causal Inference

Given a **treatment**, we want to know if there is a **causality** between the **treatment and outcome**.
Given a **control group** and **treatment group**, if we can observe the before and after treatment for both groups, we can find out different treatment effects:

- individual treatment effect: $Y_{1i} - Y_{0i}$
- average treatment effect: $E(Y_{1i} - Y_{0i})$
- subgroup treatment effect: $E(Y_{1i} - Y_{0i}|X)$

However, $Y_{1i}$ and $Y_{0i}$) can't be both observable for one group.
→ approximation

## 17.1. Data Collection in Causal Inference

**Golden Rule** **randomized controlled trials**. The treatment is controlled, individuals are assigned randomly to the treatment.
→ Sample selection bias is prevented.

### 17.1.1. Different Types of Experiments/Data Collections

- **Randomized Controlled Trials**: Treatment/Control groups separated. Subject is **randomly assigned** to the **treatment/control** group.

  → minimum selection bias
  - Lab Experiments
  - Field Experiments

- **Quasi-experiements**: **natural groups** pre-exist, no separation of control/treatment group beforehand. The independent variable(treatment variable) is **controlled**, subjects are **not randomly assigned**.

  → selection bias

- **Observational studies**: what we always have. The independent variable is **not controlled**, individuals **self-assigned**.

  → selection bias
  - cross-sectional study
  - longitudinal study
  - panel study
  - case-control study

## 17.2. Challenges to Quasi-Experiments & Observational Studies: Confounding Variables & Identification Strategies

### 17.2.1. Confounding Variables

For **Quasi-Experiments and Observational Studies**, confounding variables might exist, but **not observable, therefore omitted from the model**. In order to identify precise causal effects, we need to deal with confounding variables.

**Confounding Variables** an extraneous variable that is **unobservable**, which **correlates** with **dependent and independent variables**.
$\rightarrow Cov(\varepsilon, X) \neq 0 \rightarrow$ Endogeneity
$\rightarrow$ Consequence: biased results.

### 17.2.2. Combat Confounding Variables by Data Collection: Randomized Controlled Trials

Apply Randomized Controlled Trials: confounding variables **automatically removed**.
Ways to conduct RCTs:

- Lab experiment
- Field experiement

| | Randomized experiment (you assign them) | Quasi-experiment (they assign themselves) |
|---|---|---|
| **Field** | **High internal validity/ High external validity** | Low internal validity / High external validity |
| **Lab** | High internal validity/ Low external validity | Low internal validity/ Low external validity |

### 17.2.3. Indentification Strategy for Quasi-Experiments: Difference-in-Difference

- Idea: Observe the **effect of treatment** controlled by the researcher between **control & treatment group**, **over time**.
- Process:
  - Assume there exists an **overall trend** on both control & treatment group. We still want to estimate the treatment effect while not omitting confounding variable (eg: time).
  - treatment effect:

$$\text{Treatment effect} = (Y_{t2} - Y_{t1}) - (Y_{c2} - Y_{c1})$$

### 17.2.4. Indentification Strategy for Panel Studies: Fixed-Effect Models

- Idea: **Fixed influence** is omitted as one of the confounding variables in modeling the treatment effect on outcome.

  → model **fixed effect** to soak up **individual effects on model**.

- Fixed-Effect Model: the fixed effect is modeled as an **additional intercept** $\lambda_i$ for **each individual**.

### 17.2.5. Indentification Strategy for Observational Studies: Propensity Score Matching

- Idea: in cross-sectional data, find a **data section** where control & treatment group has the **maximum similarity** in covariate distribution.

  → resembles **randomized experiment**

- Process:
  - estimate **propensity score** by logistic regression for each individual in **treatment group**.
  - **match** the **control group** to the treatment group. Find subjects with **similar propensity score**.
  - evaluate quality of matching
  - evaluate **treatment effect** based on the **treatment and matched control group**.

### 17.2.6. Confounding Variable as Instrument Variables

**Instrument** attributes that **has causal effect** on **treatment variable**, but **no causal effect** on **outcome**.

- Modeling Process:
  - instruments **randomly assigned** to the treatment variable
  - model the relationship between the instrument and treatment variable.
  - model the relationship between the predicted treatment variable and outcome.
- Estimation: 2-stage least square.

# Teil V.
# Unsupervised Learning

## 18. Clustering

- Definition: given a database $D = t_1, \ldots, t_n$ of tuples and an integer value $k$, define a mappping $f : D \to 1, \ldots, k$ where each tuple $t_i$ is assigned to a cluster $K_j$.
- Input: a dataset with $n$ p-dimensional data instances.

  Output: a **natural partitioning** of the dataset into $k$ clusters and noise
- Clustering VS. Classification

| Characteristics | **Classification** | **Clustering** |
|---|---|---|
| Learning | supervised | unsupervised |
| Target | known | unknown, no dependent variables |
| Training | training data and training phase exists | no training data/training phase, no labels/true classes |

- key questions:
  - the right number of clusters k
  - identification of class membership between instances
- Issues:
  - interpreting results
  - evaluating results: high intra-similarity within cluster, low inter-similarity across cluster?
  - outlier
  - number of clusters k
  - scalability of algorithms

### 18.1. Hierarchical Clustering : Minimum Spanning Tree

- Input: a database D with tuples, **adjacency matrix** based on distances

  Output: **dendogram**
- Methods of building a dendogram:
  - top-down
  - bottom-up

- Algorithms: with adjacency matrix, **each instance** can be seen as **node**, **distance** to other instances can be seen as **weighted edges**

  → graph problem

  → compute **Minimum Spanning Tree**, bottom-up method.
  - Kruskal's algorithm: $\mathcal{O}(d \log(d))$
  - Prim's algorithm: $\mathcal{O}(d \log(n))$

  → each hierarchical level shares the same distance/weight.

- Distance measures in adjacency matrix:
  - **Euclidean distance** between instance $p_1$ and $p_2$

$$d_E(p_1, p_2) = \sqrt{(x_{p1} - x_{p2})^2 + (y_{p1} - y_{p2})^2 + \dots}$$

  - **Manhattan distance** between instance $p_1$ and $p_2$

$$d_M(p_1, p_2) = |x_{p1} - x_{p2}| + |y_{p1} - y_{p2}| + \dots$$

## 18.2. Partitional Clustering for Numeric Data: K-Means

- Input:
  - a database D with tuples
  - $k$ number of clusters

  Output: k partitioned clusters

- Process:
  - Initialization: randomly picked k centers
  - compute the distance between each instance to the centers, assign instance to the **nearest center**.
  - **update** the center of the clusters: **mean of assigned instances**
  - repeat step 2-3 until convergence.

- Advantages:
  - simple
  - items automatically assigned to clusters

  Disadvantages:
  - number of clusters $k$ must be predefined
  - result significantly depends on **initial choice of centers**

    → traps in **local minimum**

    → repeat algorithm by starting from different random centers (eg: Iterative Improvement)
  - sensitive to **outliers**

## 18.3. Probabilistic Clustering: Expectation Maximization

We only discuss the simplified case here: instance with single **numeric** attribute and 2 clusters A & B.

- Input: random assigned parameters for cluster A & B, assume **normal distribution**
  - A: $\mu_A, \sigma_A$, prior probability of instance in cluster A $Pr(A)$
  - B: $\mu_B, \sigma_B$, prior probability of instance in cluster B $Pr(B) = 1 - Pr(A)$

  Output: 2 clusters with assigned instances
- Process:
  - **Expectation** step: calculate the probability for **all instances** in **each cluster**: **Bayes Theorem**

$$Pr(A|x) = \frac{Pr(x|A) \cdot Pr(A)}{Pr(x)}, \quad Pr(x|A) = \frac{1}{\sqrt{2\pi} \cdot \sigma_A} e^{-\frac{(x-\mu_A)^2}{2\sigma_A^2}}$$

$$Pr(B|x) = \frac{Pr(x|B) \cdot Pr(B)}{Pr(x)}, \quad Pr(x|B) = \frac{1}{\sqrt{2\pi} \cdot \sigma_B} e^{-\frac{(x-\mu_B)^2}{2\sigma_B^2}}$$

    **no need to pick cluster here!**
  - **Maximization** step: update the parameters for cluster A & B. calculate the weighted mean and weighted variance using **all instances**.

$$w_{iA} = Pr(A|x), \quad w_{iB} = Pr(B|x)$$

$$\mu_A = \frac{w_{1A}x_1 + w_{2A}x_2 + \cdots + w_{nA}x_n}{w_{1A} + w_{2A} + \cdots + w_{nA}}$$

$$\sigma_A = \sqrt{\frac{w_{1A}(x_1 - \mu_A)^2 + \cdots + w_{nA}(x_n - \mu_A)^2}{w_{1A} + w_{2A} + \cdots + w_{nA}}}$$

    analog to $\mu_B$ and $\sigma_B$

$$Pr(A) = \frac{\Sigma w_A}{\Sigma w_A + \Sigma w_B}, \quad Pr(B) = 1 - Pr(A)$$

  - repeat expectation and maximization step until convergence.
- Limitation: can stuck in **local optimum**.

  $\rightarrow$ repeat algorithm by starting with **different initial parameters**.
- Extension of model:
  - multiple clusters: calculate k normal distributions
  - multiple attributes:
    - independent: multiply probabilities of all attributes
    - correlated: multivariate normal distribution
  - nominal attributes: create probability distribution

# 19. Association Rules Discovery

- Goal: discover **correlation among attributes** or other relationships in large databases.
- Use-case: Market Basket Analysis, cross/up-selling
- **Unsupervised learning**: no dependent variable defined, no labeled training data.

## 19.1. Terminology

**Rule** if $A$ and $B$ then $C$ and $D$. denote as $R : A, B \Rightarrow C, D$. It only describes **correlation**, not causality.

**Transaction Database** an instance/observation is a transaction. Each **attribute** in the database is converted to **binary flags 0/1**.

**Item** single element/attribute. eg: Milk/Bread

**Itemset** a set of items. eg: Milk, Bread, Butter

**Frequent Itemset** the itemset $I$ that meets the **minimum support**.

$$supp(I) \geq \min supp$$

**Support**

- support of **an item set**: **relative frequency** of the transactions that contain the item-set in **all transactions**
- support of **a rule**: the support of all item sets it contains.

  $$supp(A, B \Rightarrow C, D) = supp(\{A, B, C, D\})$$

  The **order, the arrow** of the rule **doesn't matter in computing support**.

  $$supp(\text{Milk} \Rightarrow \text{Bread}) = supp(\{\text{Milk, Bread}\}) = supp(\{\text{Bread} \Rightarrow \text{Milk}\})$$

- support **estimation**: lower bound + upper bound.
  - lower bound: the support of a subset is always higher than its superset. **subset property**, every subset of a frequent set is frequent.

    $$supp(\{B, C\}) \geq supp(\{A, B, C, D\})$$

  - upper bound: use Venn-Diagramm.

**Confidence of a Rule** the likeliness to apply to the dataset. → the probability that X and Y coexist given that X exists.

$$conf(R : X \Rightarrow Y) = \frac{supp(X \cup Y)}{supp(X)}$$

$$conf(\{\text{Milk,Bread}\} \Rightarrow \{\text{Butter}\}) = \frac{supp(\{\text{Milk, Bread, Butter}\})}{supp(\{\text{Milk, Bread}\})}$$

**Strong Rule** association rules with **minimum support & confidence**.

**Lift of a Rule** indicates **by how much (ratio)** the **confidence of a rule** surpasses the **expected value**.

$$Lift(R : X \Rightarrow Y) = \frac{conf(R)}{expConf(R)} = \frac{\frac{supp(X \cup Y)}{supp(X)}}{supp(Y)} = \frac{supp(X \cup Y)}{supp(X) \cdot supp(Y)}$$

**Interpretation of Lifts**

- lift $< 1$: X has **positive** effect on Y. Item-sets X and Y appears **more frequent than expected value**.
- lift $= 1$: X and Y are **independent**. X has **no effect** on Y.
- lift $> 1$: X has **negative** effect on Y. Item-sets X and Y appears **less frequent than expected value**.

## 19.2. A priori Algorithm: Generation of Itemsets and Rules

- Idea: if X is a frequent k-item set, then all (k-1)-item subsets of X have to be frequent item sets as well.

  → iteratively compute frequent item sets, compute k-item sets by merging (k-1)-item sets.

- Process:
  1. Generation of **Item sets**:
     - start with item sets in **size 1**.
     - only select those that **exceeds minimum support** → frequent.
     - iteratively build item sets in larger sizes based on previous sizes.

| 1-item set | supp |
|---|---|
| W | 0.6 |
| N | 0.7 |
| T | 0.7 |
| D | 0.7 |
| | |
| | |

| 2-item set | supp |
|---|---|
| W,N | 0.5 |
| W,T | 0.4 |
| W,D | 0.4 |
| N,T | 0.6 |
| N,D | 0.4 |
| T,D | 0.4 |

| 3-item set | supp |
|---|---|
| W,N,T | 0.4 |
| W,N,D | 0.3 |
| W,T,D | 0.2 |
| N,T,D | 0.3 |
| | |
| | |

(2) Generation of **Rules** based on frequent item sets:

- start with rules with only **1 item on the right**.
- rule $X \Rightarrow Y$ is different from $Y \Rightarrow X$. Compute **both directions**.
- only select rules that **exceeds minimum confidence**.
- evaluate rules containing **multiple items on the right** by checking whether single item on the right side. **Only expand if single rules exceeds minimum confidence.**

$$X \Rightarrow Y, Z \quad \text{bases on} \quad X \Rightarrow Y \quad \text{and} \quad X \Rightarrow Z$$

| 1 left, 1 right | Conf |
|---|---|
| W→ N | 5/6 |
| N→W | 5/7 |
| W→T | 4/6 |
| T→W | 4/7 |
| W→D | 4/6 |
| D→W | 4/7 |
| N→T | 6/7 |
| T→N | 6/7 |
| N→D | 4/7 |
| D→N | 4/7 |
| T→D | 4/7 |
| D→T | 4/7 |

| 2 left, 1 right | Conf |
|---|---|
| W,N→T | 4/5 |
| W,T→N | 4/4 |
| N,T→W | 4/6 |

| 0 left, 1 right | Conf |
|---|---|
| _ → W | 6/10 |
| _ → N | 7/10 |
| _ → T | 7/10 |
| _ → D | 7/10 |

Information from T1 implicates that these rules don't need to be considered

# 20. Recommendation Systems

- Approaches:
  - Association Rules: discover **correlations**
    - product association
    - user association
    - combination of both
  - Collaborative Filtering: discover **similarity**
  - Singular Value Decomposition

## 20.1. Collaborative Filtering

- Idea:
  - maintain a database of **user's rating** on items.
  - for a **given active user**, find other **similar users** whose **rating strongly correlates** with the active user.

    → recommend items highly rated by similar users, which is **not rated** by active user.

### 20.1.1. Process

(1) define **active user** $a$ and **other users** $u$.

(2) calculate **weighted correlation** $w_{a,u}$ based on **number of co-rated items m** .

- calculate average of the **co-rated items** $\bar{r}_a, \bar{r}_u$
- calculate the variance $\sigma^2_{r_a}, \sigma^2_{r_u}$ and standard deviation.
- calculate the covariance. **Don't forget the minus/plus symbol!!!**
- calculate the weighted correlation.

$$w_{a,u} = s_{a,u} \cdot c_{a,u}$$
$$c_{a,u} = \frac{Cov(r_a, r_u)}{\sigma_{r_a} \cdot \sigma_{r_u}}$$
$$Cov(r_a, r_u) = \frac{1}{m-1} \cdot \Sigma(r_a - \bar{r}_a)(r_u - \bar{r}_u)$$

(3) **rating prediction** for item i for active user.

- calculate **average rating for all rated items** $\bar{r}_a$ of active user a.
- calculate **average rating for all rated items** $\bar{r}_u$ of each other user u.
- $r_{u,i}$: other user u's rating on the i-th item.

$$p_{a,i} = \bar{r}_a + \Sigma_{u=1}^{k} \frac{w_{a,u} \cdot (r_{u,i} - \bar{r}_u)}{\Sigma_{u=1}^{k} |w_{a,u}|}$$

### 20.1.2. Limitation in Collaborative Filtering

- **Cold Start**: **enough users and ratings** are needed to generate recommendations.
- **Sparsity**: the user/rating matrix can be sparse even there are many users
  → hard to find **co-rated** items.
- **First Rater**: with a **new product**, there must first be consumers who test and evaluate it.
- **Popularity Bias**: cannot recommend items to users with **unique taste**. Tend to recommend popular items.

→ Alternative: **Content-Based Filtering**

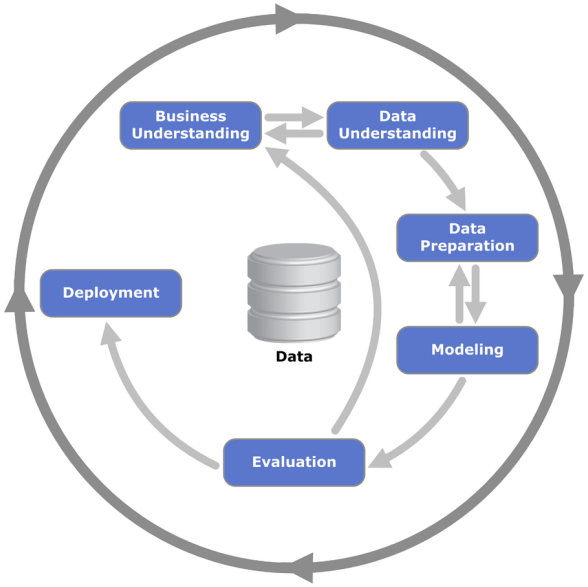
- idea: based on information of the content of items.
- solve:
  - combat popularity bias
  - combat first rater.
  - no need of user ratings → cold start + sparsity combated.

## 20.2. Singular Value Decomposition

- Idea: produce a low-dimensional representation of the customer-product space.
- Model:
$$A = U \cdot S \cdot V^T$$
  - A: the rating matrix, or the rating we want to predict.
  - U: maps **users to concepts**
  - S: strength of concepts/categories
  - $V^T$: maps **venues/products to concepts**
- **Rating Prediction**: rating for item i from user.
  - calculate/consider the average rating of user.
$$r_{u,i} = \bar{r}_u + U(user) \cdot S \cdot V^T(item)$$
- Interpretation of values:
  - User matrix(U):
    - positive: higher interest
    - negative: lower interest
    - 0: no interest
  - Product matrix($V^T$):
    - positive: **positively represented** in the i-th latent factor. Users having preference in i-th latent factor will **prefer items with positive value over items with negative values**.
    - negative: **negatively represented** in the i-th latent factor. Users having preference in i-th latent factor will **like item less**.

# Teil VI.
# CRISP-DM Process Model

The knowledge discovery process follows the following diagram: **Data Understanding** – **Data Preparation** – **Modeling** – **Evaluation**.




## 21. Data Understanding

### 21.1. Qualitative & Quantitative Understanding

Given a dataset description or an attribute table,

- Qualitative understanding:
  - format of the data: **tidy data**?
  - dependent & independent variables
  - scale of measurement of attributes: nominal/ordinal/interval/ratio
  - cross-sectional, time-series, panel data?
- Quantitative understanding
  - # instances, ideal: $> 5000$
  - # attributes, ideal start: $< 50$
  - # targets (balance of classes), ideal: $> 100$ each class

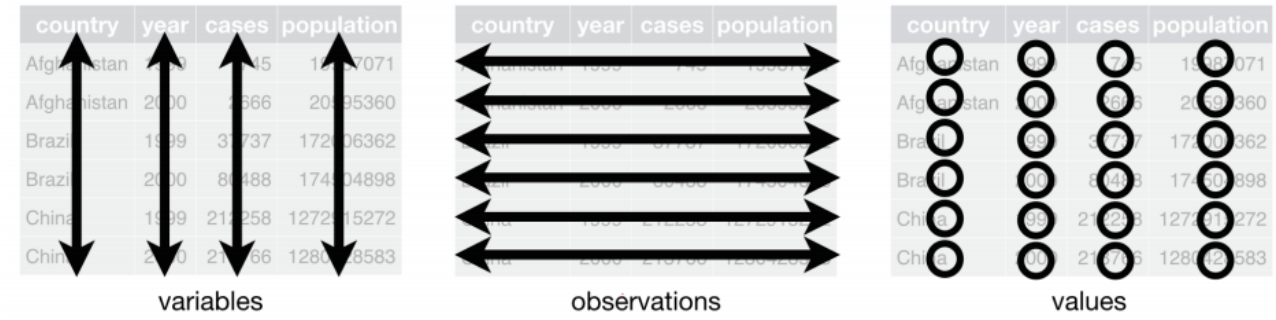
Ways to obtain understanding:

- Visualization: histogram, distribution, relationship between attribute and response
- Summary: mean, median, attribute relationships

## 21.2. Format of Data: tidy?

**Tidy Data** tabular data is tidy if

- each **variable/feature** is in **single column**
- each **observation/instance** is in **single row**
- each **value** is in **single cell**.




example of tidy data:

| Class | Year | #Students | #TAs |
|---|---|---|---|
| Business Analytics | 2019 | 650 | 5 |
| Business Analytics | 2020 | 800 | 6 |
| Data Mining Seminar | 2019 | 16 | 2 |
| Data Mining Seminar | 2020 | 21 | 3 |

**Wide Data** **same variable** is spanned into **multiple columns**. eg: year.

| Class | Students2019 | Students2020 | TAs 2019 | TAs 2020 |
|---|---|---|---|---|
| Business Analytics | 650 | 800 | 5 | 6 |
| Data Mining Seminar | 16 | 21 | 2 | 3 |

**Long Data** **multiple variables** is compressed into **one column**. eg: n_Students & n_TAs.

| Class | Year | Variable | Value |
|---|---|---|---|
| Business Analytics | 2019 | n_Students | 650 |
| Business Analytics | 2019 | n_TAs | 2 |
| Business Analytics | 2020 | n_Students | 800 |
| Business Analytics | 2020 | n_TAs | 3 |
| Data Mining Seminar | 2019 | n_Students | 16 |
| Data Mining Seminar | 2019 | n_TAs | 2 |
| Data Mining Seminar | 2020 | n_Students | 21 |
| Data Mining Seminar | 2020 | n_TAs | 3 |

# 22. Data Preparation

According to the data understanding, we distribute the data preparation also into these 3 parts.
But first of all, **tidy** the data if the format is long/wide. → pivoting.
The data preparation works on tidy data.

## 22.1. Instance: Missing Value?

Missing values need to be **standardized**:

- **ignore** instance with missing values
- treat missing values as **separate value**
- **imputation**: mean/median

## 22.2. Attributes: Conversion, Discretization & Feature Selection

### 22.2.1. Conversion

- ordinal → numbers, preserving **natural order**.
- nominal
  - #attribute values is small: nominal → numeric, **each attribute value** becomes a single **binary attribute** (number of binary classes: **(#attributevalues - 1)** per attribute).
  - #attribute values is large: ignore (eg: IDs)
- continuous numeric: Discretization, only if model requires(eg: naive Bayes).

### 22.2.2. Discretization / Binning

- Reasons for Discretization:
  - data: numeric data not normally distributed, data requires sorting frequently(decision tree).
  - model: model requires nominal data as input(naive Bayes).
- Reasons against Discretization:
  - in decision tree: equi-depth bins **doesn't maximize the information gain**. Through the algorithm(binary split) we can find better splits instead of binning.
  - will potentially **lose ordinal information**.
  - model-dependent: regression models requires numeric data, no discretization.
- Ways of Discretization:
  - equi-frequency/depth

  - class dependent: when decision tree classifier is used, **best split** according to **information gain**.
  - order-preserving: numeric → k nominal → $(k-1)$ binary attributes, explaining comparison between $(i-1)$ and $i$.

#### 22.2.3. Feature Selection

- Goal: choose the **most relevant subset**.
- Methods:
  - Best Subset (search all, select best)
  - Forward Selection (bottom-up)
  - Backward Elimination (top-down)
  - Stepwise Regression (combines forward/backward)
- Use-Case: linear regression, classification, dimensionality reduction, regularization
- Ideal: at most 50.

### 22.3. Targets: Balanced Train & Test Set

According to the distribution of targets, build up **balanced set** and then split into **balanced train set** and **balanced test set**.

- targets are balanced: each set gets **same amount** of targets
- targets are **unbalanced**: split **proportionally**.

This only applies to training & fitting models. It doesn't apply to statistical inference.

## 23. Model Selection

### 23.1. Bias-Variance Tradeoff & Sweet Spot

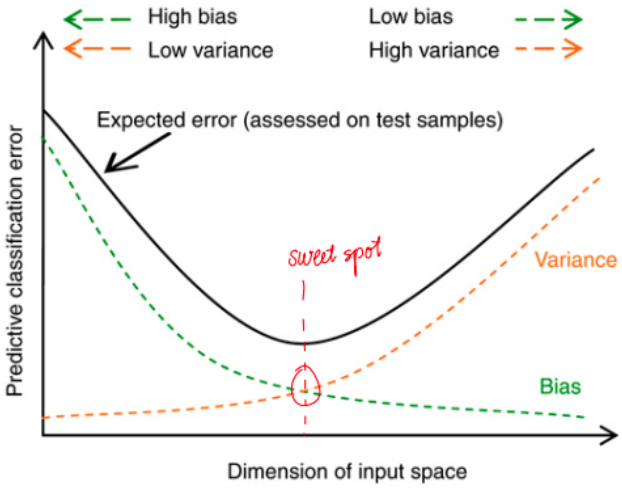
**Bias** error from **erroneous assumptions** in learning algorithm. Error can range from inaccurate assumption to simplification of model.

**Variance** error from **sensitivity to small fluctuations** in training set.

- Idea:
  - models **too simple** will have **high bias** on training data, **low variance** on test data. → **Underfitting**
  - models **too complex** will have **low bias** on training data, **high variance** on test data. → **Overfitting**

→ find the sweet spot → **low bias & low variance**

- **Goal** of Model Selection: **optimize** bias-variance tradeoff
- Criterion Metrics: **minimize** f(fitting error from given data) + g(model complexity)
  - **Akaike Information Criterion(AIC)**: $AIC = -2\ln(L) + 2 \cdot \#parameters$
  - **Minimum Description Length** given the same quality: Kolmogorov Complexity




# 24. Evaluation

## 24.1. Evaluation Methods of Model

- Goal of evaluation: how good is the model on **new data**?
- Evaluation methods: how to get the **test data**?
  - on training set
  - Holdout set (stratified / repeated)
  - k-Fold Cross-Validation (w/o stratified)
  - Leave-One-Out Validation
  - Bootstrap

### 24.1.1. Evaluation Directly on Training Set: Not Preferred

- might cause **overfitting**.
- evaluation too optimistic, the actual error rate is higher.

→ not preferred!!

### 24.1.2. Evaluation using Holdout Set

- reserve data from whole data. rule of thumb: $\frac{1}{3}$ of whole.
- holdout set method:
  - **stratified Holdout**: considers **distribution of classes**. split the training/test data **proportionally** according to the ratio of classification results.
  - **repeated Holdout**: **randomly** select holdout set **repeatedly** and estimate through **average error**.

### 24.1.3. Evaluation using (stratified) k-fold Cross Validation

Process:

- **partition** the data (**proportionally, if stratified**) into $k$ complementary subsets.
- **train** on $(k-1)$ subsets, **test** on 1 subset.
- repeat until each subset is tested once.
- calculate the **average error rate**.

### 24.1.4. Evaluation using Leave-One-Out Validation

- Use-case: when data is **scarce**.
- a **n-fold cross validation**: test on 1 instance, train on $(n-1)$ instances.
- Advantages:
  - maximum use of data for training, especially when data is scarce.
  - deterministic

  Disadvantages:
  - high computational cost
  - non-stratified samples

### 24.1.5. Evaluation using Boostrap

Process:

- draw $n$ **random samples with replacement** as test data.
- test and calculate the error rate.
- repeat the random sampling for many times.
- calculate the variance/confidence interval of the sample.

Comparison to k-fold cross validation: sampling **without** replacement.

### 24.1.6. Significance between Models: Paired T-Test

- Goal: compare the **error rate** of 2 models
  → see which model fits better to the **training data**
  → better model will predict on **test data**.
- Idea: results of a validation may be considered as **random chance**.
  → only **significant difference** counts! → significance test

**Paired T-test:**

**Significantly Different? two-sided Test**

- $H_0$: $\mu_d = 0$
  $H_1$: $\mu_d \neq 0$
- test statistic:
$$t = \frac{\bar{d} - \mu_d}{s_d/\sqrt{n}}$$
- critical value: $t_{1-\frac{\alpha}{2},n-1}$
- Reject $H_0$: $|t| \geq t_{1-\frac{\alpha}{2},n-1}$

**Significantly Better? one-sided Test**

- $H_0$: $\mu_{C1-C0} \leq 0$, classifier 1 is not significantly better than baseline classifier.
  $H_1$: $\mu_{C1-C0} > 0$, classifier 1 is significantly better than baseline classifier
- critical value: $t_{1-\alpha,n-1}$
- Reject $H_0$: $t \geq t_{1-\alpha,n-1}$

## 24.2. Quality Metrics of Model on Test Data

### 24.2.1. Confusion Matrix

| | | Predicted class | |
|---|---|---|---|
| | | Yes | No |
| **Actual class** | Yes | True positive (TP) | False negative (FN) (Type I error) |
| | No | False positive (FP) (Type II error) | True negative (TN) |

**Overall Diagonally**:

#### Accuracy

$$\text{Accuracy} = \frac{TP + TN}{N}$$

#### Error Rate

$$\text{Error Rate} = 1 - \text{Accuracy} = \frac{FP + FN}{N}$$

**Specific Horizontally**:

#### True Positive Rate / Recall / Hit Rate

$$\text{True Positive Rate/Recall} = \frac{TP}{TP + FN}$$

#### True Negative Rate / Specificity

$$\text{True Negative Rate/Specificity} = \frac{TN}{TN + FP}$$

#### False Positive Rate / False Alarm Rate

$$\text{False Positive Rate/False Alarm Rate} = 1 - \text{Specificity} = \frac{FP}{TN + FP}$$

**Specific Vertically**:

#### Precision

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

**Cost-Sensitive Learning**:

- Goal of general test data evaluation: minimize **overall error rate**
  → same weight on each prediction
- Idea in cost-sensitive learning:
  - unbalanced data
  - prediction has different cost
- Goal in cost-sensitive learning: minimize **cost**
- Solution:
  - **weighting** of instance according to cost
  - **resampling** of instance according to cost
  - **predict probabilities** instead of predicting classes. minimize the cost by selecting a better **cutoff-value** (default: 0.5)

  → the model **biased towards cost-sensitive** prediction.

  → eg: in churn prediction, better predict more churns (more false positives as false negatives)

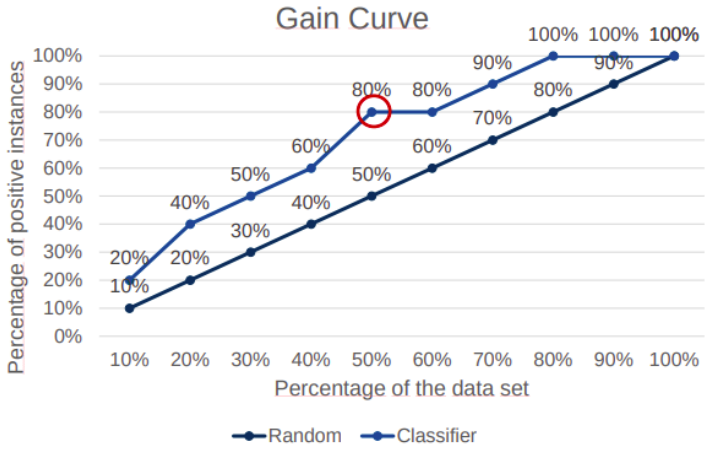
### 24.2.2. Gain Curve

- Idea: visualize results of **different cutoffs**.

  → the **gain most of the targets** by just taking **a percentage of the whole dataset**, no need to go through the whole.

  → evaluate models in **cost-sensitive learning**

- Process:
  - predict probabilities instead of classes.
  - **sort** instances by probability **in descending order**
- x-axis: percentage of the dataset
- y-axis: percentage of **actual true** instances in the whole given dataset

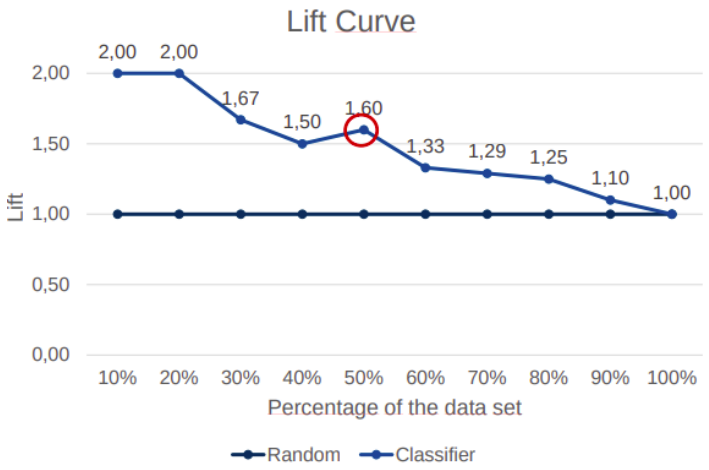



### 24.2.3. Lift Curve

- Idea: visualize **how much better** sorting and taking the q% of data set is than random sampling q%.
- x-Axis: percentage of the dataset $q$
- y-Axis: Ratio of sorting and taking the q% to random sampling

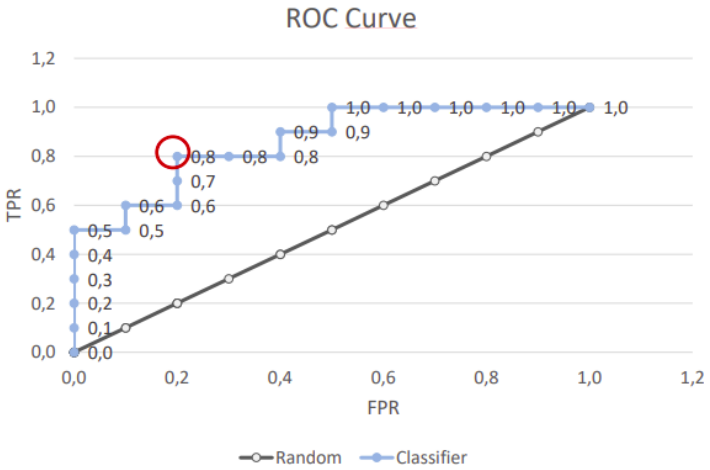
  → $\min(\text{Lift}) = 1$

$$\text{Lift}(q) = \frac{\text{Gain}(q)}{q}$$




### 24.2.4. ROC Curve

- Idea: go through all sizes of samples
- x-Axis: false positive rate
- y-Axis: true positive rate
- Process:
  - **sort** the predicted probability (the given table might be unsorted).
  - increase the sample size **step-wise** as a cutoff for positive prediction,
    - with one more true positive (+), **go up** one step.
    - with one more false positive(-), **go right** one step
- Choosing cut-off value: choose the **percentage that bends to the left the most**.
- Comparing 2 models: choose the model that **bends to the left** the most.
- Mark the cut-off value: find the corresponding **instance predicted at cut-off value**.



# 25. Dimensionality Reduction

- Reasons:
  - reduce a complex dataset to a **lower dimension**.
  - simplify data understanding, visualization and manipulation
  - reduce computation time
  - reveal hidden dynamics – latent variables, **multicollinearity**
  - the data lies on a lower dimensional subspace anyway.
- Techniques to Dimensionality Reduction (combat multicollinearity):
  - Subset Selection
    - Best Subset, Forward Selection, Backward Elimination, Stepwise Selection
  - Derived Input in Regression
    - Principal Component Regression
    - Partial Least Squares
  - Regularization (Coefficient Shrinkage)
    - Subset selection
    - Ridge Regression
    - Lasso Coefficient
- Comparison Linear Regression VS. Dimension-Reduction Techniques
  - Linear Regresion:
    - requires $n \geq p$, more observations than variables.
      $\rightarrow$ reality large observation too costly, variables too much
    - if number of variables too large $\rightarrow$ **Overfitting**
    - can't combat multicollinearity, separate test on VIF.
    - **unstable** to little variability on data in prediction results
  - Dimension-Reduction Techniques:
    - PCR: combats multicollinearity through computing linear uncorrelated principal components.
    - **more stable** to the variability on data if variables are correlated.
    - Regularization: through feature selection, introduce **bias** but **reduce variance (smaller MSE)**.

## 25.1. Principal Component Analysis

- Definition: converts a set of possibly **correlated** variables into a (possibly smaller) set of **linearly uncorrelated** variables – **Principal Components**.
- Goal: transform the data, such that the new dimensions are **linear uncorrelated** and we **maximize the variance** along the axes.
- Assumption: relationship among variables is **linear**.
- **Principal Components**: **explain most of the variability** in the original dataset.
  - the **eigenvectors** of the covariance/correlation matrix
  - the direction are those in feature space along which the original data is **highly variable**.
  - the **first PC** has **largest possible variance**. It's the direction of **maximum variance from origin**.
  - **subsequent PCs** are **orthogonal** to first PC. They describe **maximum residual variance**.
  - each element of eigenvector represents the **contribution of a variable** to the PC.
- PCA **Eigenvalues**: give the **proportion of variance explained** by the corresponding principal components.
  - $\lambda_1$ shows the proportion of variance explained by PC1. $\rightarrow$ the **spread of data** in PC1 direction.
- PCA **Scores**: Z, score of x are the coefficients of in each PC direction.

### 25.1.1. Process

(1) **center** the data, subtract the **mean** from each data dimension. $\rightarrow$ zero-mean dataset.

(2) compute **covariance/correlation matrix**:

- covariance matrix: variables in **comparable units**, **difference in variance** across variable **important**
- correlation matrix: variables in **different units**, **difference in variance** across variable **not important**

$$Var(x_j) = \frac{1}{N-1} \cdot \Sigma x_{ij}^2$$

$$Cov(x_{j1}, x_{j2}) = \frac{1}{N-1} \cdot \Sigma x_{ij1} x_{ij2}$$

(3) compute **eigenvalues and eigenvectors** of the covariance/correlation matrix. **Normalized** the eigenvectors.

(4) **order** eigenvectors according to **its eigenvalues in descending order** $\Phi$.

(5) compute variance explained by each principal component:

$$\text{variance explained} = \frac{\lambda_i}{\Sigma\lambda_i}$$

(6) Project the transformed data onto principal components: all principal components are **orthonormal basis**.

$$Z = X \cdot \Phi$$

(7) Compress: choose $k$ most important PCs. $\rightarrow$ slight **information loss**

### 25.1.2. Reconstruction of Original Data

$$D \approx Z\Phi^T + \text{means}$$

If **dimensionality reduced**, we **lose** those dimensions we choose to **discard**. The information loss is relatively small.

### 25.1.3. Computation Principal Component: Singular Value Decomposition

$$A = U \cdot S \cdot T^T$$

- Alternative to compute principal components.
- principal axes/components: columns of V
- principal component scores $U \cdot S$
- Use-case: recommendation system

## 25.2. Principal Component Regression

- Multiple Linear Regression VS. Principal Component Regression
  - PC combines the correlated variables into linear uncorrelated variables. It explain the most important variability of the model.

    $\rightarrow$ **combats multicollinearity and unstability** to minor change in data from linear regression models.
- PCR Model:

$$y = Z \cdot \gamma + \varepsilon, \quad \text{with} \quad Z = X \cdot \Phi$$

  - independent variables: principal components in Z
- works well when the first few principal components are sufficient to explain most of the variation.
- not a feature selection method.

### 25.2.1. Partial Least Squares

- identifies new features in a **supervised** way:
  - new features **approximate old features** and are **related to response**.
  - weights reflect the covariance structure between **predictors and response**
- requires more complicated iterative algorithms

## 25.3. Regularization: Ridge Regression

### 25.3.1. Regularization

- Goal: **introduce bias** into regression solution that can **reduce variance** relative to OLS solution.
- Objective function in regularization:

$$J(\theta) = L(\theta) + \Omega(\theta)$$

  - $L(\theta)$ : training loss, describes **model fit**
  - $\Omega(\theta)$: regularization, describes **model complexity**

### 25.3.2. Ridge Regression

- Goal: **minimizes** a **penalized** RSS
- Penality: $\mathbf{l_2}$ **penality**

$$\hat{\beta}^{ridge} = \arg\min_{\beta}(RSS + \lambda \cdot \Sigma\beta_j^2)$$

  - $\lambda \uparrow$ , coefficients $\rightarrow 0$
  - coefficents will never be exactly 0, but nearly 0.
- Evaluation: estimates **more biased** but have **lower variance** than OLS-Estimator.

## 25.4. Regularization: Lasso

- Goal: **minimizes quantity**
- Penality: $\mathbf{l_1}$ **penality**

$$\hat{\beta}^{lasso} = \arg\min_{\beta}(RSS + \lambda \cdot \Sigma|\beta_j|)$$

- Finding tuning parameter $\lambda$ : select a grid of values + cross-validation
- Evaluation & Comparison Ridge Regression:
  - has the effect of forcing some coefficients to be **exactly zero**, when $\lambda$ is large.
    $\rightarrow$ feature selection

- produces **simpler and more interpretable** models involving only **subset of predictors**
- similar behavior to ridge regression: $\lambda \uparrow$, variance $\downarrow$, bias $\uparrow$.
- generate **more accurate predictions**.