EFR summary

Seminar Data Analytics

2025 - 2026

Week 1-2

Deloitte.

DeNederlandscheBank
EUROSYSTEEM

Details

Subject: Seminar Data Analytics (AAC) 2025 - 2026

Teacher: M. Merlo, P. Breuer

Date of publication: 31.03.2026

Koffietje doen?

Start jouw carrière bij BDO

Maak kennis met BDO Accountants & Adviseurs, de beste plek om als toptalent aan de slag te gaan. De koffie staat voor je klaar. Vertellen wij je over wat jij kan bijdragen, en jij ons over je ambities.

Scan de QR-code en plan jouw koffiemoment in.

werkenbijbdo.nl ▶

BDO

Lecture 1: Anomaly Detection

Data analytics = transforming raw data into decision-relevant information.
It is an iterative and feedback-driven process. Insights reshape original questions.

Data is a core economic resource.
Organizations use analytics to: Optimize performance, Reduce risk (fraud detection, compliance) and Support strategic decisions (pricing, investments)

Accountants are evolving from 'scorekeeper' (recording past performance) to 'decision facilitator' (providing strategic insights)
→ More data-driven tasks, Risk assessment & predictive analysis, Collaboration with data specialists and Evaluating analytics

Anomaly Detection

Fraud and scandals often show abnormal patterns before collapse. For example:
- Enron: Stable profits despite rising off-balance-sheet obligations
- WorldCom: Inflated earnings by capitalized operating expenses

An outlier: "An observation which deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism." (Hawkins, 1980), Also referred to as anomaly.

Anomaly Detection: Identify observations that deviate from expected patterns, making them inconsistent with the rest of the data
- Most observations = normal behavior
- Anomalies = rare and generated by a different process

Examples: Fraudulent transactions, System errors, Data processing mistakes

Why Anomaly Detection Matters

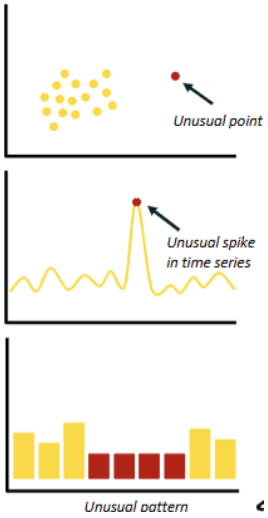
- **Improves Data Quality**: Identifies errors, duplicates, inconsistencies**.** Prevents distorted statistical results
- **Enhances ML Performance**: Prevents models from fitting noise. Detects structurally unusual observations
- **Supports Better Decision-Making**: Ensures reliable conclusions, Reduces misleading analysis
- **Reduces Financial & Reputational Risk**: Warning signals for fraud or failure. Protects investors, firms, auditors

Anomaly Detection → Cleaner Data → Better Models → Better Decisions → Lower Risk

Applications of Anomaly Detection

- **Fraud detection** (unauthorized transactions)
- **Network security** (abnormal logins)
- **Medical diagnosis** (rare disease patterns)
- **Manufacturing** (equipment failure)
- **Climate & environment** (extreme weather)

Types of Anomalies



1. Point Anomalies :
Single unusual observation. Example: One unusually large withdrawal

2. Contextual Anomalies
Unusual given a specific context. Example: Vendor payment at midnight

3. Collective Anomalies
Group of observations jointly unusual. Individual values may appear normal. Example: Sequence of small refunds

Challenges of Anomaly Detection

1. Conceptual Challenge

- Defining "normal" depends on benchmark (industry, firm size, time)
- Normal behavior evolves over time

2. Statistical Challenge

- Class imbalance (rare events)
- Limited/no labeled data

3. Decision Challenge

- Risk of misclassification

	Actual Normal	**Actual Anomaly**
Predicted Normal	True Negative	*False Negative*
Predicted Anomaly	*False Positive*	True Positive

Classes of Techniques

Statistical Methods

- Use numerical thresholds to detect outliers, examples: Z-Score, IQR

Visual Methods

- Spot unusual patterns through plots, example: Histogram

Machine Learning Methods

- Detect anomalies using algorithms,
- Approaches: Supervised, Semi-Supervised, Unsupervised

Note: No single method detects all outliers → use multiple approaches.

Statistical & Visual Methods

Z-Score

The Z-Score standardizes an observation relative to the sample distribution

$$z = \frac{x - \mu}{\sigma}$$

- x = observation
- μ = mean
- σ = standard deviation

Decision Rule:
Flag if: $|z| > k$
Common threshold: k = 3

Interpretation:

- Measures distance from mean in standard deviations
- Larger |z| = more extreme

Appropriate When:

- Numeric data
- Approximately normal distribution
- Mean & SD meaningful
- Detecting extreme values

Strengths:

- Simple
- Unit-free
- Fast screening tool

So, Z-score standardizes observations relative to mean and variance.
Z-score works best for symmetric, normally distributed numeric data.

Z-Score: Robustness Issues

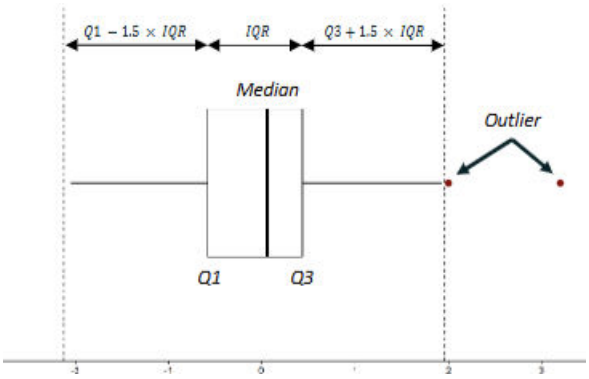
Sensitivity:

- Sensitive to skewness
- Sensitive to extreme outliers (mean & SD shift)

Masking Effect:

- Extreme outlier prevents other outliers from being detected

Interquartile Range (IQR)



- Q1 (25th percentile)
- Q3 (75th percentile)
- IQR = Q3 − Q1

Decision Rule:
Flag if: $x < Q1 - 1.5 \times IQR$ or $x > Q3 + 1.5 \times IQR$

Interpretation:

- Uses median as central measure
- Focuses on central 50%
- Less affected by extreme values

So, IQR method is more robust than Z-score.

Point anomaly can be detected using the IQR method

Appropriate When:

- Numeric data
- Skewed data
- Extreme values present
- Mean/SD unreliable

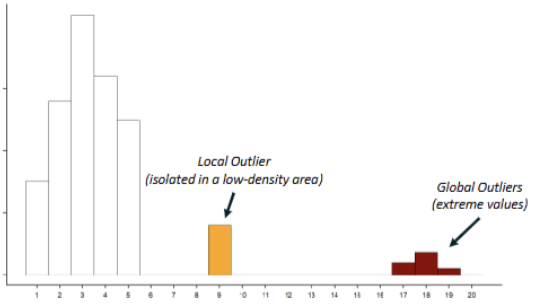
Strengths:

- Robust
- No normality assumption
- Simple

Limitations of Statistical Methods

- **Threshold-based**: Require predefined thresholds, Results depend on benchmark
- **Univariate** (one variable at a time): Cannot detect multivariate anomalies
- **Focus on extreme values**: May miss local anomalies

Histogram



Divides data into equal-width intervals (bins) and counts the number of observations per bin.
Anomaly Identification: Low-density regions may indicate anomalies.

Appropriate When:

- Numeric data
- Exploring distribution
- Detecting extreme observations
- Visualizing low-density (sparse) regions

Strengths:

- Intuitive
- Not limited to extreme values

Limitations:

- Sensitive to bin width
- No formal decision rule
- Univariate only

When to Use These Methods

Statistical Methods:

- Univariate
- Extreme outliers
- Quick screening

Visual Methods:

- Visualizing low-density areas

Both struggle with high-dimensional data.

- Statistical methods on individual columns miss cross-variable patterns
- Visual methods cannot display high-dimensional relationships

Solution: Dimensionality reduction or Multivariate anomaly algorithms (e.g., Isolation Forest)

Multivariate Anomaly Detection

Anomalies can be unusual combinations of normal values.

Example:

Person	Height	Weight
1	190 cm	80 kg
2	145 cm	45 kg
3	190 cm	45 kg

Individually plausible, jointly unusual (Person 3).

Limitation:
Z-score & IQR analyze variables separately.
Z-score fails to detect an anomaly in a 2D dataset because the anomaly is only extreme in combination of variables.

Machine Learning Advantage:

- Learn joint patterns
- Detect deviations in high-dimensional space

Machine Learning Methods

What is Machine Learning?

Artificial Intelligence (AI): Systems mimicking human intelligence.

Machine Learning (ML): Subfield of AI. Learns patterns from data. No explicit rule programming

Applications: Image recognition, Speech recognition, Recommendation systems

Types of Machine Learning

1. Supervised Learning

- Uses labeled data to accurately classify data or predict outputs
- Predict output (Y) from input variables (X)

Examples: Classification (spam vs. non-spam) or Regression (predict revenue)

2. Unsupervised Learning

- Uses unlabeled data
- Discovers hidden patterns and relationships in data from input variables (X)

Examples: Anomaly detection, Clustering or Dimensionality reduction

Why Anomaly Detection Is Usually Unsupervised

- Anomalies are rare
- Labels often unavailable
- Fraud patterns evolve over time

Supervised methods require labeled anomalies and stable fraud definitions
Therefore, focus on unsupervised algorithms that learn normal behavior and flag deviations

But Supervised Methods Are Not Useless

Supervised methods are powerful when labels exist and provide clear classification
Difference in questions:

- Supervised → *What predicts fraud?*
- Unsupervised → *What looks different from the rest?*

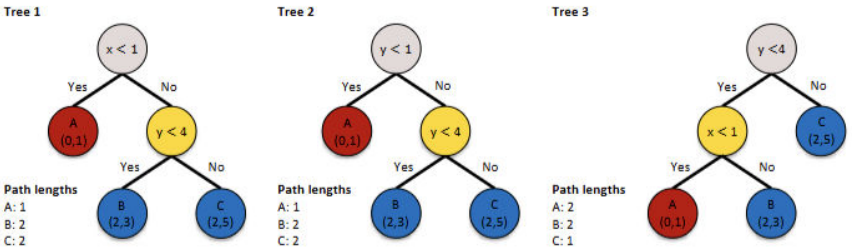
Isolation Forest

Isolation Forest is an unsupervised, tree-based ML method. It is designed for anomaly detection and works well on numeric tabular data. Goal: anomalies easier to isolate.

Assumption: Anomalies are rare & distinct → random partitions isolate them quickly.

"Is feature (x) larger/smaller than threshold (k)?"

Data Point	x	y
A	0	1
B	2	3
C	2	5



Fewer splits needed to isolate a point → more abnormal.

Isolation Forest: Algorithm

For each tree:

1. Randomly select a feature (e.g., x or y)
2. Randomly select a split value (between min and max)
3. Split data
4. Repeat until:
    - Only one observation in node OR
    - Maximum depth reached
5. Measure path length (h(x))
6. Repeat across many trees
7. Compute average path length
8. Convert to anomaly score

Anomaly Score

Formula:

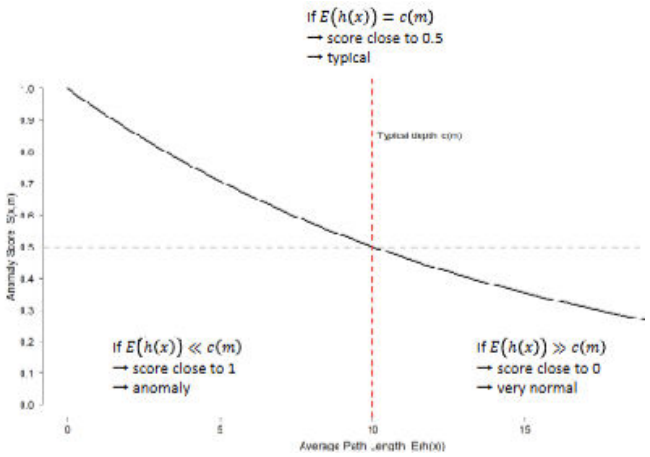
$$S(x,m) = 2^{-E(h(x))/c(m)}$$

, where
E(h(x)): average path length of data point x
c(m): expected average path length in a tree of size m
m: number of data points used to build each isolation tree

Output:

- Continuous score in $\in(0,1)$
- Not binary

So, Short path → high anomaly score.



Two main parameters:

1. Number of Trees (t)

- More trees → more stable
- Higher computation cost

2. Subsample Size (m)

- Default often 256
- Not necessarily full dataset

Why smaller samples? → Faster, Anomalies stand out more

Why Ensemble?

- Single tree: High variance and Random splits unstable
- Isolation Forest: Many trees, Average path length stabilizes results and Reduces randomness

So, Ensemble reduces variance and increases stability.

Isolation Forest: Strengths & Limitations

Strengths:

- Scales well to large datasets
- Works in high dimensions
- Robust to irrelevant features
- Fast to compute

Limitations:

- Limited interpretability (cannot clearly explain which features cause anomaly)
- Produces ranking, not classification (requires professional judgment)
- Stochastic (slightly different results per run)

Case Study: Financial Data

- Goal: Detect unusual firm-years.
- Dataset: 1,500 firm-years
- Variables: Assets, Sales, Net income, Debt, Operating cash flow, R&D, Industry
- Challenge: Financial data messy (missing values, scale differences)

Steps: 1. Prepare data, 2. Build model, 3. Extract scores, 4. Interpret results

Problem: Raw levels scale with firm size. Example:
SmallCo (Assets=100, NI=5), BigCo (Assets=10,000, NI=500) → Both ROA = 5%
Solution: Use ratios such as ROA, Leverage (Debt/Assets), Efficiency (Sales/Assets)
Ratios reflect economic behavior, not size.

Problem: Isolation Forest requires complete data.
If missing: Cannot assign branch, Tree inconsistent. Solutions:

- Missing indicator variable
- Impute numeric (mean/median)
- Impute categorical (mode)
- Drop rows only if rare (<1–2%)

Important: Missingness itself can signal anomalies.

Problem: Different variables have different ranges. Example:

- ROA: -20% to +20%
- Leverage: 0–90%
- Turnover: 0–5+

Isolation Forest splits between min and max. Large-range variables dominate splits.
Solution: Standardization reduces dominance.

Interpretation of the Model Output

The Isolation Forest model ranks firm-years by anomaly score. The highest-ranked observation (Firm 423 in 2024) has a score of 0.639, indicating it is relatively unusual. By examining the standardized financial variables, we see strong negative profitability, negative cash flow, very low leverage, and missing R&D data, which together explain why the model flags this firm-year as anomalous.

Variable	Standardized Value	Economic Value	Interpretation
ROA	-2.59	-11.4%	Strongly negative
Leverage	-2.05	1.0%	Very low
Margin	-2.34	-12.5%	Strongly negative
OCF/A	-1.54	-7.7%	Negative
Turnover	-0.03	0.91	Normal
R&D missing	1	NA	Missing

Standardized values = statistical abnormality
Economic values = business explanation

Lecture 2 – Unsupervised Learning

Recap: Lecture 1

Anomaly detection identifies observations that deviate from expected patterns in the data. Assumptions:

- Most observations represent normal behaviour
- Anomalies are rare
- Anomalies are generated by a different process

Methods for detecting anomalies

Method Type	Example	Type	Strengths	Limitations
Statistical	Z-score, IQR	Univariate	Simple, fast screening, easy to interpret	Requires thresholds, sensitive to distribution assumptions
Visual	Histogram	Univariate	Intuitive, helps understand distribution	No formal decision rule, sensitive to bin width
Machine Learning	Isolation Forest	Multivariate	Detects anomalies across many variables	Harder to interpret

Supervised vs Unsupervised Learning

Machine learning can learn from data in two ways.

Supervised Learning	Unsupervised Learning
Uses labeled data	Uses unlabeled data
Predicts outcome **Y**	Discovers patterns in **X**
Goal: Prediction	Goal: Structure / pattern

Example Supervised: Fraud prediction, Credit risk prediction, Revenue forecasting

Example Unsupervised: Anomaly detection, Dimensionality reduction, Clustering

Unsupervised Learning: Structure Discovery

Two main approaches:

1. Factor Analysis
Focuses on structure in variables.
Goal: Reduce dimensionality and identify underlying latent factors
Example idea: Many variables → fewer underlying factors.

2. Cluster Analysis
Focuses on structure in observations (objects).
Goal: Identify groups of similar observations

Dimensionality

High-dimensional data: A dataset may contain many variables (p).

Problems:

1. Correlated and noisy variables

- Ideally variables provide independent information
- In reality many variables are redundant

2. Computational complexity

- More variables → more computation time

3. Visualization difficulty

Solution: Dimensionality reduction
Transform high-dimensional data into fewer dimensions while keeping as much information as possible.

Example: Large firms tend to have **high values on variables like assets, sales, employees, etc.** Therefore: these variables may represent a **single underlying concept: Firm size.** Instead of 5 variables → possibly **1 dimension**.

Factor Analysis

Main idea: Given many interrelated variables, find a smaller number of latent variables (factors) that describe them.
Central concept: Factor analysis is based on correlations between variables.

Uses

- Dimensionality reduction
- Discover underlying factors

Example: Many financial variables → underlying factors like profitability or risk.

Types of Factor Analysis

1. Principal Component Analysis (PCA)
Purpose: Dimension reduction
Example: Combine 10 financial ratios → profitability component

2. Exploratory Factor Analysis (EFA)
Purpose: Discover latent constructs
Example: Personality traits from survey questions.

3. Confirmatory Factor Analysis (CFA)
Purpose: Test theoretical models
Example: Test if survey questions follow a one-factor model.
Also called: Structural Equation Modeling (SEM).

Principal Component Analysis

Tries to represent data with fewer variables.
Idea: Create new variables that are linear combinations of the original variables

The new variables are called **principal components**

- The components are uncorrelated
- Number of components ≤ number of variables

Usage

- Dimensionality reduction
- Preprocessing step before further analysis

Each principal component is a **linear combination** of variables:

$$Z_s = u_1X_1 + u_2X_2 + u_3X_3 + \ldots + u_pX_p$$

Where:

- (X) = original variables
- (u) = weights
- (Z) = principal component

Goal: Each component should explain as much variance as possible.

Eigenvalues: Measure the amount of variance explained.

- Larger eigenvalue → more information captured

Important: Variables are usually standardized before PCA.

Why standardize variables?

PCA gives more weight to variables with higher variance.
If variables use different units: Results become **biased**
Example: Height vs weight measured in different units.

Standardization formula:

$$X_{ij}^{s} = \frac{X_{ij} - \overline{x_j}}{s_j}$$

Where:

- subtract mean
- divide by standard deviation

Result: mean = 0, variance = 1

Important: Correlation ≠ causation.
When interpreting PCA components: Be careful not to overinterpret correlations.

PCA Strengths

- Reduces dimensionality: Summarizes correlated variables
- Simplifies datasets: Makes visualization easier
- Useful for feature engineering: Components can be used in prediction models

PCA Limitations

- Components may be hard to interpret: New variables are linear combinations of many variables
- Sensitive to scaling: Requires standardization
- Only captures linear relationships

Conducting PCA Steps:

1. Formulate the problem
2. Determine number of dimensions
3. Retain variables
4. Interpret dimensions
5. Use components in further analysis

PCA Example Dataset

Dataset: job satisfaction
Variables (rated 1–4): Supervisor, Salary, Career possibilities, Customers, Colleagues, Work content, Company, Work-life balance
Scale: 1 = very unsatisfied, 4 = very satisfied

PCA Step 1: Problem Definition

Goal: **Dimensionality reduction**
Question: Can we combine **8 job satisfaction variables** into **fewer components**?
Variables: Work content, Colleagues, Supervisor, Salary, Career possibilities, Customers, Company, Work-life balance. Each rated on a 4-point scale.

Factor analysis relies on **correlations between variables**. Expectation: Variables that are **highly correlated** likely belong to the **same dimension**. Correlation matrix →

		(1)	(2)	(3)	(4)	(5)	(6)	(7)
Work content	(1)	1						
Colleagues	(2)	0.210	1					
Supervisor	(3)	0.243	0.389	1				
Salary	(4)	0.153	0.127	0.235	1			
Career possibilities	(5)	0.350	0.184	0.288	0.417	1		
Customers	(6)	0.219	0.165	0.139	0.045	0.125	1	
Company	(7)	0.239	0.213	0.317	0.151	0.334	0.242	1
Work-life balance	(8)	0.101	0.098	0.172	0.075	0.158	0.114	0.309

Example correlations:

- Supervisor & colleagues → moderate correlation
- Career possibilities & salary → stronger correlation

Interpretation: Highly correlated variables may form the **same component**.

PCA Step 2: Determining the Number of Dimensions

When conducting PCA, we must decide how many components (dimensions) to keep. Important note: There is no universally accepted optimal rule. Approaches:

1. A priori determination

- The researcher decides the number of dimensions beforehand
- Based on **theory or prior knowledge**

2. Kaiser criterion

- Keep components with **eigenvalues > 1**
- Reason: each retained component should explain at least as much variance as one variable

3. Scree plot

- Plot **eigenvalues vs number of components**
- Look for the **"elbow" point** where the curve flattens

4. Proportion of variance

- Choose number of components that explain enough **cumulative variance**

Kaiser Criterion

Key rule: Keep components where: **Eigenvalue > 1**

Why? Because each retained component should explain **at least the variance of one original variable**.

Example Table

Dimension	Eigenvalue	Variance %	Cumulative Variance %
Dim 1	2.51	31.39	31.39
Dim 2	1.10	13.73	45.12
Dim 3	0.99	12.32	57.44
Dim 4	0.93	11.61	69.05
Dim 5	0.75	9.42	78.47

Interpretation: Only **Dim 1 and Dim 2** have eigenvalues > 1 → keep **2 dimensions**.

Scree Plot

A scree plot is a graphical way to determine the number of components.

- **X-axis:** number of dimensions
- **Y-axis:** eigenvalues

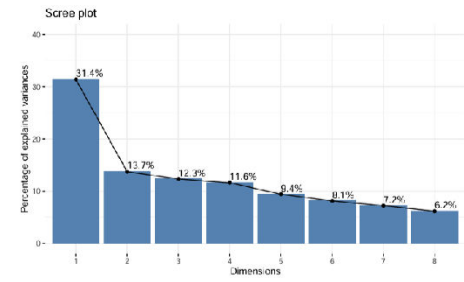
Eigenvalues are ordered from largest to smallest.

Interpretation: Look for the **"elbow" point** where the curve suddenly flattens.

Before elbow → important dimensions

After elbow → less useful dimensions

Limitation: The elbow is sometimes **hard to identify clearly**.



Here the elbow point is at 2 dimensions

Proportion of Variance

Another method is looking at **cumulative explained variance**.

Idea: Keep enough dimensions to explain a large portion of total variance.

Typical threshold in **social sciences: ≈ 60% cumulative variance**

Example

From the table:

- Dim 1 → 31.39%
- Dim 1 + 2 → 45.12%
- Dim 1 + 2 + 3 → 57.44%
- Dim 1 + 2 + 3 + 4 → 69.05%

Thus **4 dimensions** would pass the 60% threshold.

	Eigenvalue	Variance %	Cumulative Variance %
Dim. 1	2.51	31.39	31.39
Dim. 2	1.10	13.73	45.12
Dim. 3	0.99	12.32	57.44
Dim. 4	0.93	11.61	69.05
Dim. 5	0.75	9.42	78.47
Dim. 6	0.65	8.13	86.60
Dim. 7	0.58	7.23	93.82
Dim. 8	0.49	6.18	100.00

PCA Step 3: Retain the variables

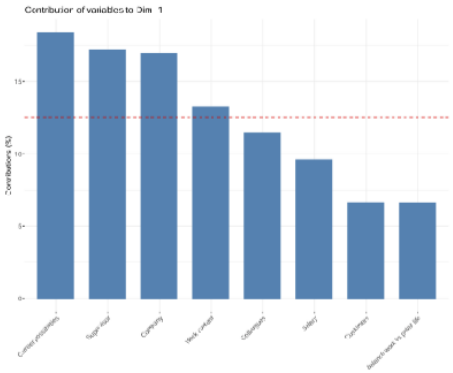
Next step: determine **which variables contribute most to each component**.
Contribution = percentage of variance of a dimension explained by a variable.
Interpretation: Some variables contribute strongly to **one specific dimension**.

	Dim. 1	Dim. 2
Work content	13.23	0.05
Colleagues	11.43	1.82
Supervisor	17.19	0.00
Salary	9.58	39.38
Career possibilities	18.39	15.48
Customers	6.63	22.29
Company	16.93	6.98
Work-Life Balance	6.61	14.00

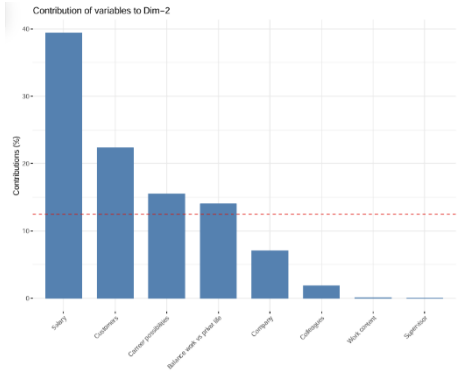
Contribution Bar Plot (Dimension 1)

- Each bar = contribution of a variable
- Dashed red line = **expected average contribution**

Expected value: 1/p
Where: p = number of variables
Here: 1/8 = 12.5%
Interpretation: Variables above the dashed line contribute **more than average** to the component.



Contribution Bar Plot (Dimension 2)
Interpretation: Variables with the highest contributions define the **meaning of that dimension**.
Example: Dimension 2 may relate more to: Salary, Career possibilities, Customers, Work-life balance



PCA Step 4: Interpret the dimensions

Interpretation is based on **variable loadings**. Distance from origin indicates **strength of relationship.**

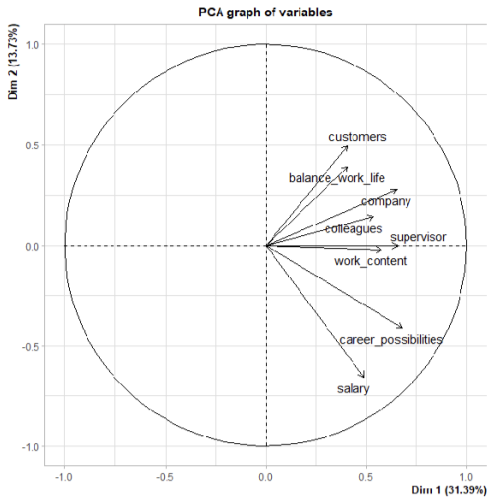
1) Idenitfy variables with high loadings
   - **Variables at the end of an axis** → strongly related to that dimension
   - **Variables near the origin** → not strongly related to any dimension
   - **Variables between axes** → related to multiple dimensions

2) Look for common theme

Moderate positive correlation between:
→ customers, work-life balance and dimension 2.

High poisitive correlations between:
→ supervisor, career possibilities, company and dimension 1.

High to moderate negative correlation between:
→ salary, career possibilities and dimension 2



PCA Step 5: Use in subsequent analysis

After PCA, we compute **component scores**.

Formula: $Z_s = u_1X_1 + u_2X_2 + \ldots + u_pX_p$

Each observation gets values for Dimension 1 and Dimension 2

Respondent	**Dim 1**	**Dim 2**
1	3.99	2.19
2	-0.32	-1.18
3	2.52	-1.79

Standardized realizations of X for respondent 1

Respondent	Work content	Colleagues	Supervisor	Salary	Career possibilities	Customers	Company	Work-life balance
1	1.57	1.44	1.52	0.47	0.40	2.61	1.99	1.93

Variable weights for dimension 1

Dimension	Work content	Colleagues	Supervisor	Salary	Career possibilities	Customers	Company	Work-life balance
1	0.36	0.34	0.41	0.31	0.43	0.26	0.41	0.26

$Z_1 = 0.36 \times 1.57 + 0.34 \times 1.44 + 0.41 \times 1.52 + 0.31 \times 0.47 + 0.43 \times 0.40 + 0.26 \times 2.61 + 0.41 \times 1.99 + 0.26 \times 1.93 = 3.99$

PCA in Supervised Learning

- PCA can be used for **feature engineering**.
  Idea: Replace several correlated variables with one principal component.
- Example: Bankruptcy prediction
  Variables: Profit margin, EBIT / Assets, Net income / Assets
  These are highly correlated.
- Instead create: **Profitability component**
  Prediction model: Bankruptcy = f(leverage, liquidity, profitability component)

Benefits

- Reduces multicollinearity
- Simplifies models

Cluster Analysis

Main idea: Group observations into clusters so that:

- objects in the **same cluster are similar**
- objects in **different clusters are different**

Based on similarity between observations

Uses

- Market segmentation
- Fraud detection
- Data reduction

For example: Playing Cards What are the natural groupings?
Possible clusters: By suit (hearts, clubs, etc.), By number
→ Clusters depend on the features used.

Data Exploration

- Cluster analysis is **exploratory**.
- Quote from Hastie et al. (2009): There is no direct measure of success in unsupervised learning.
- Therefore: Results must be interpreted carefully, Researchers rely on heuristics and judgment

Types of Clustering Methods

1. Hierarchical Clustering
Creates a **tree structure of clusters**.
Two approaches:

- **Agglomerative**
  - Start with individual observations
  - Merge clusters
- **Divisive**
  - Start with one cluster
  - Split into smaller clusters, each observation is one cluster

2. Non-hierarchical Clustering
Example: **k-Means clustering**

- Number of clusters **k chosen beforehand**
- Observations assigned to nearest cluster center

k-Means Clustering

k-Means is a **partitioning method**. It divides objects into **k clusters**.
Clusters are: distinct and non-overlapping
Why popular? → Simple, Efficient, Works well with large datasets

Step 1: Generate **k random cluster centers**. These are initial means.

Step 2: Assign each observation to the **nearest cluster center**.
Distance is usually measured using **Euclidean distance**.
Thus each point belongs to the **closest centroid**.

Step 3: Update cluster centers.
Each cluster center becomes: the mean of the observations assigned to that cluster.
Thus cluster centers move toward the **center of their assigned points**.

Step 4 Repeat steps:

1. Assign observations to nearest center
2. Update cluster means

Continue until **convergence**.
Convergence occurs when: cluster assignments **no longer change**

k-Means: Strengths

1. Identifies groups of similar observations

- Useful for discovering patterns in data.

2. Good for segmentation

- Frequently used in marketing, finance, and customer analysis.

3. Simple and computationally efficient

- Easy to implement.
- Works well on **large datasets**.

4. Reveals heterogeneity

- Different clusters may represent **different behavioral patterns**.

k-Means: Limitations

1. Number of clusters must be chosen beforehand

- Selecting the correct **k** can be difficult.

2. Sensitive to initialization

- Different starting cluster centers can produce **different results**.

3. Sensitive to outliers

- Extreme values can **distort cluster centers**.

Conducting k-Means Clustering

The clustering process follows **five main steps**:

1. Formulate the problem
2. Select a distance measure
3. Decide on the number of clusters
4. Interpret the clusters
5. Use clusters in subsequent analysis

Example Dataset: Loan Applications

Dataset: **500 rejected loan applications** from a US online bank.
After removing non-numeric variables (date, loan title, ZIP code, state), four numeric variables remain:

Variable	Scale
Amount requested	$1,000 – $35,000
Credit risk score	0 – 850
Debt-to-income ratio	0 – 3.16
Employment length	0 – 10 years

Credit score categories:

Score	Category
0–299	Very Poor
300–579	Poor
580–669	Fair
670–739	Good
740–799	Very Good
800–850	Excellent

Step 1: Formulate the Problem

Problem definition: Goal is segmentation
Research question: Are there distinct clusters of rejected loan applicants based on their characteristics?
Important: Each variable uses **different measurement scales**.

Loan application	Amount requested	Credit risk score	Debt-to-income ratio	Employment length
1	25,000	690	0.49	0
2	3,000	0	0	0
3	5,000	654	0.09	0

Step 2: Select a Distance Measure

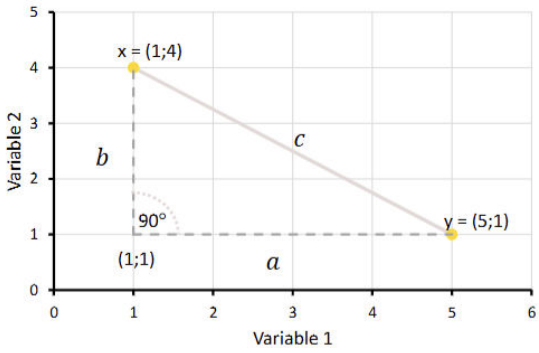
Clustering is based on similarity between observations.
Key idea: Objects with smaller distances between them are more similar.

Euclidean Distance For two observations (x) and (y):
This measures the straight-line distance between two points in multidimensional space.

$$d_{euc}(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

Important issue: The scale of variables matters.
If one variable has much larger values: It dominates the distance measure.
Solution: Standardize variables before clustering.

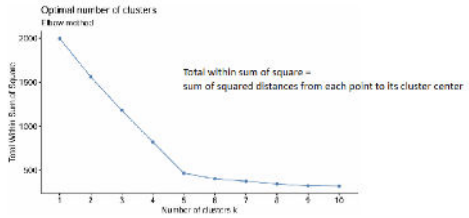
Euclidean Distance Illustration using the Pythagorean theorem.
Two points: x = (1,4) y = (5,1)
Distance is calculated as the straight-line distance between them.



Step 3: Decide the Number of Clusters
In k-means, k must be chosen before running the algorithm. However, selecting k is difficult. Methods to determine k:
1. Elbow Method
Look for the point where the improvement slows.
2. Average Silhouette Method
Choose the k that maximizes the silhouette score.
Both methods rely on **visual interpretation**.

Elbow Method

- The elbow method uses a plot of: **Within-cluster variance vs number of clusters**
- Within-cluster variance measures how tightly observations are grouped.
- Goal: Minimize **within-cluster variation**.
- Formula idea: Total within sum of squares = sum of squared distances between points and their cluster center.
- **Interpretation:** Choose k where the curve forms an **elbow**.
- After this point: Adding clusters provides little improvement.

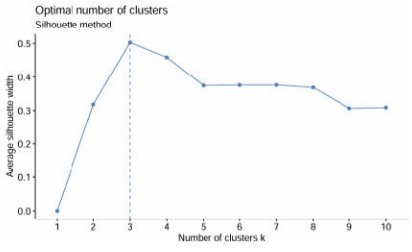


Silhouette Method

Another approach: **Average silhouette score**.
Silhouette value: Measures how well an observation fits its cluster.
High silhouette score means:

- observation is **similar to its own cluster**
- observation is **different from other clusters**

Range: -1 to +1
Interpretation: Higher average silhouette → **better clustering**.
Choose k with **maximum silhouette score**.



NbClust Package
In practice, software can help determine k. Example: **NbClust package (R)**
It evaluates clustering using **30 different indices**.
The package: tests multiple clustering schemes, compares different numbers of clusters and suggests the **best clustering solution**

Step 4: Interpret Clusters

Once clustering is completed, analyze cluster statistics.

Cluster	Size
1	9
2	275
3	51
4	133
5	32

Interpretation: Cluster 2 is the largest group of rejected applicants. → Cluster size indicates importance of clusters.

Cluster Visualization

Clusters can be visualized in 2D space.

→ fviz_cluster(): This function first applies PCA to reduce dimensions. Then clusters are plotted in two dimensions. This allows us to visualize clusters clearly.

Interpreting Cluster Centroids

Cluster centroids represent average values of variables within each cluster.

Example centroid table:

Cluster	Amount Requested	Credit Score	Debt-to-Income	Employment Length
1	9,888	480	3.03	1.22
2	6,811	615	0.18	0.38
3	14,191	665	0.26	8.37
4	26,376	665	0.28	0.32
5	6,528	0	0.01	0.78

Interpretation: Clusters describe different borrower types.

Example: Cluster 3: higher employment length, moderate credit score

Step 5: Use in Subsequent Analysis

- Clusters can be used in further analysis.
- Clustering can be used for feature engineering.
- Clusters can also become new variables.

Example: Borrower types. Clusters may represent:

→ Young borrowers: low income, short credit history, high income growth

→ Established borrowers; stable jobs, long credit history

Prediction model example: Default risk = f(income, debt ratio, cluster)

Benefits

- Captures **heterogeneity**
- Different groups may follow **different behavioral patterns**

Example: Income predicts default for young borrowers,
Debt ratio predicts default for established borrowers.

Lecture 3: Supervised Learning I

Important distinction:

- **Unsupervised learning** uses **unlabeled data** to **discover patterns**
- **Supervised learning** uses **labeled data** to **predict outcomes**

Example 1: Find a comprehensive index for job satisfaction
→ This is unsupervised learning, because the goal is discovering patterns/structure.

Example 2: Which borrowers have their loan applications rejected?
→ This is supervised learning, because we want to predict an outcome.

Supervised Machine Learning

- Uses **labeled data (training data)** to train an algorithm.
- The algorithm learns patterns from the labeled data.
- It can then **predict outcomes for new, unseen data**.

Terms below are used interchangeably:

- feature = predictor = independent variable = X
- outcome = dependent variable = label = Y

Learning a Prediction Function

In supervised learning the algorithm estimates a function: $y = f(x)$
Meaning: model **learns the relationship between predictors (X) and outcome (Y).**

Examples of algorithms used to estimate (f): Regression, Bayesian classification, Decision trees, Random forests, Support Vector Machines, etc.

Example application: Predict fraud
Inputs (X): CEO bonus, Loss, Leverage, Governance characteristics
Output (Y): Fraud (yes/no)
Model learns: $Fraud = f(Firm\ characteristics)$

Practical Applications

Machine learning has many real-world uses, especially in accounting.
Example Auditors Applications: Client selection, Audit planning, Analytical procedures
Large accounting firms (e.g. Big 4) invest heavily in ML tools. Use cases:

- Analysis of journal entries
- Detect suspicious patterns such as: questionable keywords, unauthorized entries and large numbers of entries just below approval limits

Example: Regulators (SEC)
Machine learning can help detect misconduct. Example workflow:

1. **Unsupervised learning**: Detect patterns in financial filings or text.
2. **Supervised learning**: Predict which filings may involve misconduct.

Machine Learning in Research

Machine learning is widely used in **accounting research**.
Historically: Mainly used for **fraud detection**.
New research topics include:

1. Loss estimation: ML predictions outperform managerial estimates
2. Future earnings prediction
3. Peer firm selection
4. Litigation risk prediction

Research Spillovers to Practice

Example research question: Can machine learning help choose better directors?
Findings Erel, Stern, Tan & Weisbach (2021):

- Directors predicted by ML to perform poorly: actually perform poorly in out-of-sample tests.
- Characteristics of weak directors: more likely male, hold many directorships, have larger networks

Implication: Machine learning can improve corporate governance decisions.

The supervised learning workflow.

1. Explore the data
2. Preprocess the data (feature engineering)
3. Train algorithms
4. Predict outcomes for unseen data

This lecture focuses on building prediction models using training data.
Model evaluation will be covered in Topic 4.

Step 1: Explore the Data (EDA)

Goals of EDA: Understand the structure of the dataset. Typical tasks:

1. Check summary statistics of variables
2. Identify missing values
3. Detect outliers
4. Identify important/unimportant variables

Methods used: Correlation matrix, Data visualizations, Principal Component Analysis

Step 2: Data Preprocessing

This includes:

- cleaning data
- feature engineering
- preparing data for machine learning models.

Data Splitting

- Data must be split before modeling.
- Purpose: Evaluate model performance later.
- Typical split Dataset:
  → **Training set:** used to train the model
  → **Test set:** used to evaluate model performance

Common split ratios:

- 80% training / 20% testing
- In R default: 75% / 25%

Missing Values → Imputation

- Replace missing values with estimated values.

Examples: Mean, Median, Model-based predictions, k-nearest neighbor imputation
This ensures the dataset can be used for modeling.
How much missing data is too much? → no general rule, but 20% missing is common

Outliers

1. Detect outliers

Use simple functions such as: summary statistics, histograms, boxplots

2. Transformation and normalization

Examples of transformations: Logarithms, Normalization, Scaling, Centering, Purpose:

- Reduce extreme values
- Make data more suitable for machine learning algorithms.

Feature Creation and Selection

= Create new variables from existing variables.

- Examples: Calculate ratios, Create dummy variables, Create interaction terms
- Example: Instead of using revenue and assets separately, you might create: profitability ratio or leverage ratio

Important idea: Human intuition and creativity in understanding the data can make a big difference. Key takeaway: A simple model with good features can outperform a complex model with poor features.

Why not include all variables? Reasons:

- Some features may add noise.
- Too many features may cause overfitting.
- Some variables may be irrelevant.

Methods to select features:

1. **Simple filters**: Remove variables with little variation or correlation.
2. **Recursive Feature Elimination (RFE)**: Iteratively remove least important variables.
3. **Stepwise selection**: Add or remove predictors step-by-step based on model performance.

Goal: Keep only the most useful predictors.

Step 3: Train Algorithms

At this stage: The machine learning algorithm learns patterns from the training data.

Examples of supervised learning algorithms: Linear regression, Logistic regression, Naïve Bayes, k-Nearest Neighbor (k-NN), Decision Tree, Random Forest, Gradient Boosting, Many others

Key question: Which algorithm should you choose? → It depends on:

- The research question
- The audience
- The type of data
- Computational resources

Model Evaluation

After training the model: Evaluate performance using testing data (Topic 4).
Examples of evaluation metrics:
Continuous outcomes: Mean Absolute Percentage Error (MAPE)
Binary classification outcomes: Accuracy = Correct predictions / Total predictions
Additional evaluation tools:

- Confusion matrix
- ROC curves

Important concept: Machine learning modeling is an iterative process.
Meaning: You rarely build the best model in one attempt.

Some supervised learning algorithms

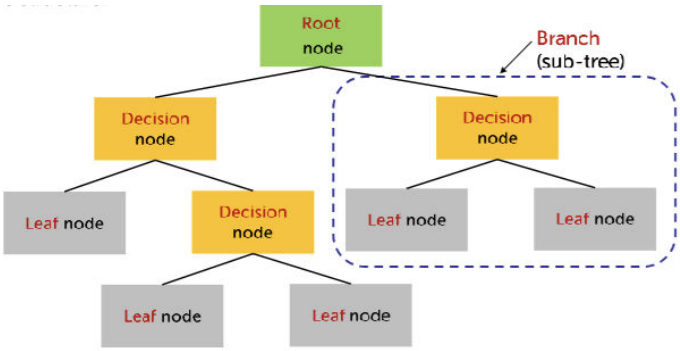
- Basic concept: Provide computer with **training data** containing input variables (X) and output variable (Y). The computer learns **patterns** from the data.
- Common algorithms include: Decision Trees, Random Forest, Gradient Boosting, Nearest Neighbor, Naïve Bayes, Logistic / Linear Regression, Support Vector Machines (SVM), Neural Networks, Lasso, Ridge, etc.

Decision Trees

Example problem: You want to detect whether a photo shows a winter family vacation. This is a binary classification problem: Yes vs No. Possible features to check:

1. Are there humans in the picture?
2. Is there snow?
3. Are there multiple people?

Decision trees classify data using a sequence of simple questions.



Components of a decision tree

Root node: The starting point of the tree.
Decision nodes: Nodes where a feature-based split occurs.
Branch: Connection between nodes.
Leaf nodes (terminal nodes): Final outcomes.

Goal: Create leaf nodes that are pure (contain only one class).

Decision Nodes vs Leaf Nodes

Decision node: A point where the algorithm chooses a **feature to split on**.

- Example: "Is there snow?" This divides the data into subgroups.

Leaf node: Final prediction node.

- The algorithm predicts: either a **pure class** or the **majority class** in that node.
- Objective: Create accurate predictions with as few decisions as possible.

Choosing the Best Split

Different algorithms exist: ID3, C4.5, CART, MARS
Common metric used: **Information Gain**
Formula idea: Information Gain = Entropy(parent) - Average Entropy(children)
Entropy measures **impurity**. Lower entropy means data is more homogeneous.
The algorithm selects the split that **maximizes information gain**.

When to Stop Splitting

Decision trees can grow **very large**. Large trees can cause **overfitting**.
Solution: **Pruning**
Two main approaches:

1. Pre-pruning
Stop tree growth early using rules.
Examples: Minimum number of observations per node or Maximum tree depth
Problem: Stopping too early may lead to **horizon effect** (premature stopping).

2. Post-pruning
Build full tree first, then simplify it. Branches may be replaced by **leaf nodes**.
Goal: Reduce complexity and Improve generalization.

Decision Tree Pros and Cons

Pros:

- Easy to interpret
- Simple to visualize
- Handles both numerical and categorical data
- Requires little data preparation

Cons:

- Can overfit
- Sensitive to data changes.

Greedy Nature of Decision Trees

Decision trees are **greedy algorithms**.

- Meaning: They choose the **best split at the current step**.
- Problem: They cannot reconsider earlier decisions later.
- Example: A feature that might be better used later may be incorrectly used early.
- Result: Decision trees may **not find the globally optimal solution**.

Random Forest

Concept: Random forest is built using **many decision trees**.
Instead of relying on a single tree: Random forest **combines many trees**.
This improves prediction accuracy and stability.
Random forest is an **ensemble model**. Meaning: Multiple models are combined.

For classification: Use **majority voting**
For regression: Use **average prediction**
Random forests reduce **overfitting** compared to single trees.

Random Forest Feature 1: Random Sampling

Also called: **Bootstrap sampling**
Each tree is trained on a different subset of the training data.
Some observations may appear multiple times.
Purpose: Make trees different from each other.

Random Forest Feature 2: Random Feature Selection

Second source of randomness:
At each split, the algorithm selects a **random subset of features**.

Example classification problem: Predict whether a fruit is apple or orange
Possible features: Color, Shape, Whether it is sliced
Different trees may use **different features**. This increases model diversity.

Random Forest Feature 3: Voting

Final prediction process: Each tree makes a prediction.
Example: Tree predictions, Final prediction: 4 apple, 2 orange
This is called **majority voting**.

Random Forest Workflow

1. Training dataset
2. Generate multiple bootstrap samples
3. Train one decision tree per sample
4. Each tree predicts the outcome
5. Predictions are aggregated using **majority voting**

Final output: **Combined prediction**

Random Forest: Bagging

Important term: **Bagging** (Bootstrap + Aggregating)
Meaning: Random forests use **multiple random datasets** to build many trees.
How randomness is introduced:

1. **Random sampling of training data**
   - Each tree is trained on a random subset of observations.
   - Sampling is done with replacement.
   - Some observations appear multiple times.
2. **Random subset of features**
   - At each split, the tree considers only some features instead of all.

Purpose: Create **diverse trees,** Reduce correlation between trees, Improve prediction accuracy.

Random Forest Pros and Cons

Pros:

- Higher predictive accuracy
- Less overfitting
- Works well with complex data
- Handles large datasets

Cons:

- Harder to interpret
- More computationally expensive
- Not as transparent as a single tree

Interpretability vs performance trade-off:

- Decision trees → easy to interpret
- Random forests → more accurate but complex

Gradient Boosting

Also called: **Gradient Boosted Trees**. Another **ensemble model**.
Concept: Combine many **weak learners** (usually small decision trees).

Random Forest	**Gradient Boosting**
Trees built independently	Trees built sequentially
Predictions aggregated at end	Each tree improves previous trees
Focus on randomness	Focus on reducing prediction errors

Important ideas:

- **Gradient** → minimizing prediction error
- **Boosting** → improving weak models step-by-step.

Regression Example

Predict continuous outcome variable Y from feature X. Data shows a nonlinear relationship between X and Y. Goal: Predict Y using multiple simple trees.
Gradient boosting works for: regression, classification.

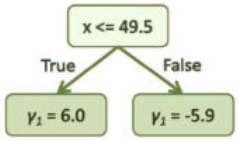
First Step of Gradient Boosting

Start with a **naive** prediction. Initial prediction: $F_0 = \bar{y}$
Meaning: Predict the average value of Y for every observation.
This produces initial prediction errors → These errors are called: **Residuals**

Correcting Errors

Next step: Train a small regression tree using the residuals as the target variable.
Residuals represent: $r_1 = actual - predicted$
The tree tries to predict the errors made by the first prediction.
Goal: Reduce prediction errors.

Updating the Prediction



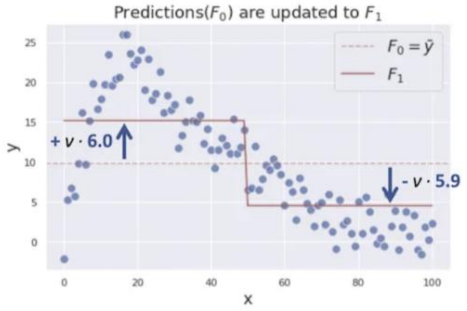
The new tree predicts correction values: Example: $r_1 = 6.0, -5.9$
These predicted residuals are added to the previous prediction.
Updated prediction: $F_1 = F_0 + v \cdot r_1$
Where:

- (v) = learning rate between 0 and 1

Purpose: Prevent large updates and control model complexity.

Updated Model

After correction:
The prediction function becomes: $F_1 = \begin{cases} F_0 + v \cdot 6.0 & if\ x \leq 49.5 \\ F_0 - v \cdot 5.9 & otherwise \end{cases}$
This updated model is closer to the true values.
The model gradually improves with each iteration.



Iterative Improvement

Gradient boosting repeats the process:

1. Calculate residuals
2. Fit a tree to residuals
3. Update prediction
4. Repeat

Continue until the prediction improvement converges to zero

Gradient Boosting Pros and Cons

Pros:

- Very high predictive accuracy
- Works well for complex datasets
- Captures nonlinear relationships
- Often best-performing model in competitions

Cons:

- Computationally expensive
- Loss of interpretability
- Hard to interpret
- Can overfit if poorly tuned

K-Nearest Neighbor (kNN)

Another supervised algorithm.
Concept: Predict outcomes based on **similar observations**.
Steps:

1. Store all training observations.
2. For a new observation: calculate distance to all training points
3. Find the k closest neighbors
4. Use their outcomes to make prediction.

Example: Predict survival of a Titanic passenger.
New passenger: 20 years old, male, 3rd class
Find the 5 most similar passengers.
If: 3 survived and 2 died
Prediction → Survived

kNN for Classification vs Regression

Classification problem: Prediction based on **majority class** of neighbors.
Regression problem: Prediction based on **mean** or **median** of neighbors.

Importance of k

Parameter: **k = number of neighbors**
Choice of k strongly affects results.
Prediction depends on which class dominates among the nearest neighbors.

- Small k: more flexible, risk of overfitting
- Large k: smoother predictions, risk of underfitting.

Steps of kNN Algorithm

1. Observe the dataset
2. Calculate distances between new observation and training data
3. Rank neighbors by distance
4. Select the **k closest observations**
5. Predict outcome based on their votes.

Distance Measures

Distance calculation depends on feature type.

- **Continuous variables** → Most common: **Euclidean distance**: $\sqrt{\sum(x_i - x_i')^2}$
  Measures geometric distance between points.
- **Categorical variables** → Use: **Hamming distance**: $\sum|x_i - x_i'|$
  Distance is: 0 if features are the same, 1 if different.

Feature Normalization

Problem: Features may have different range and units.
Example: Age: 0 ~ 80 years, passenger class: 1 ~ 3
Large-scale variables dominate distance calculations.
Solution: **Normalize features**
Example: **Min-Max Normalization** → $x_{normalized} = \frac{x - x_{\{min\}}}{x_{\{max\}} - x_{\{min\}}}$
Result: Values scaled between **0 and 1**.

kNN Pros and Cons

Pros:

- Simple algorithm
- Easy to implement
- No assumptions about data distribution
- Works well for small datasets

Cons:

- Slow and expensive for large datasets
- Sensitive to irrelevant features
- Requires careful choice of k
- Sensitive to scaling of variables.

Naïve Bayes

Another classification algorithm. Based on Bayes' theorem.
Purpose: Estimate the probability of outcome Y given predictors X.
Posterior probability: P(Y|X)
Required probabilities:

- (P(Y)) → prior probability
- (P(X)) → probability of features
- (P(X|Y)) → conditional probability

Important assumption: features are **independent**.
This assumption is often unrealistic → hence the term **"naïve."**

Bayes Theorem Formula

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}$$

Example dataset variables:

Male (X)	Survived (Y)
1	1
0	0
1	0
0	1
0	1

Probabilities can be estimated from training data frequencies.

- $P(X = 1 \mid Y = 1) = 1/3$
- $P(Y = 1) = 3/5$
- $P(X = 1) = 2/5$

Final calculation: $P(Y = 1 \mid X = 1) = 0.5$

Naïve Bayes With Multiple Features

For multiple predictors → Naïve assumption: all predictors are independent.
Example: predict survival for a passenger: Age = 20, Male, 3rd class
Compute: $P(\text{Survived} = 1 \mid X_1, X_2, X_3)$
Multiply probabilities:

→ $P(Age \mid Survived) \cdot P(Male \mid Survived) \cdot P(Class \mid Survived) \cdot P(Survived)$

Do this for both: Survived = 1 and Survived = 0
Choose the class with the higher posterior probability.
If P(Survived = 1 | X) > P(Survived = 0 | X) → Predict **Survived**.

Naïve Bayes Pros and Cons

Pros:

- Easy to build
- Fast and efficient
- Works well with **large datasets**
- Can update easily with new data
- Good for **high-dimensional data** (e.g., text)

Cons:

- Assumes **independence of features** (often unrealistic) so does not hold in most cases

Lecture Summary

No single best algorithm
Different algorithms work better depending on: problem type, data characteristics, prediction goals.

Always compare models
Procedure: Train several models, Evaluate performance on out-of-sample data, Select the best model.

Feature engineering is critical
Good predictors are essential. Key rule: **Garbage in → Garbage out**
If input data is poor, even the best algorithm will fail.

Lecture 4 Part 1: Model Evaluation I

Recap of what you've learned so far:

- Data preparation
- Detecting relationships
- Making predictions

Learning goals today:

- Understand what makes a **"good" model**
- Understand **overfitting**
- Learn how to **evaluate predictive performance** for:
    - Regression models
    - Classification models

What is a "good" model? (Explaining vs Predicting)

Two perspectives:

1. Explanatory models (research focus)
- Aim: understand causal relationships
- Focus on internal validity (causal identification) and statistical significance
- Also care about external validity (generalization)

2. Predictive models (this course)
- Aim: accurate predictions on new data
- Less focus on interpreting coefficients
- Still useful if causal insights are found

Scientific value of prediction
1. Helps discover new patterns not in theory
2. Helps improve measurement of concepts
3. Tests how well theories perform in reality
4. Provides benchmark accuracy

Key idea:
- If theory ≈ prediction benchmark → theory is strong
- If far away → room for improvement

Implications: Explaining vs Predicting
- Variable selection: Prediction → variables must be available in future
- Multicollinearity: Problem for explanation. Less problematic for prediction
- Missing values: Prediction → must not be included (e.g., imputation)
- Model choice: Explanation → interpretable. Prediction → accuracy matters more
- Real-world: often combine multiple models

Overfitting (concept)

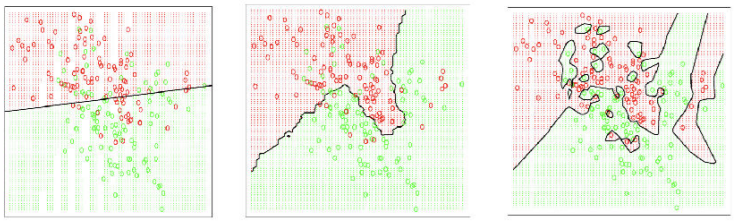
We want to find real patterns → good predictions. But if model is too flexible, we will always find such patterns. However, these "patterns" might just be chance occurrences in the data.

Overfitting = Finding chance occurrences in data that look like interesting patterns, but which do not generalize

Model Complexity: more complex models are also more likely to accommodate random data structures.

Underfitting vs Good vs Overfitting

- Overfitting = "memorizing" training data
- Works well on training data
- Performs poorly on new data



Under-fitting Good Over-fitting

Model Evaluation Approaches

1. Domain knowledge validation

- Expert checks ("sanity check")
- Ensure: Model makes sense, Data context is valid

2. Holdout validation (technical)

- Split data into (1) Training set → build model and (2) Test set → evaluate performance
- Compare: In-sample (1) vs out-of-sample (2) accuracy
- Advanced: Resampling (e.g., cross-validation) → more robust estimates

Train/Test Split

- Typical split: 80% training / 20% testing (common rule)
- Important considerations: Need enough observations vs number of features
- Splitting methods: Random split (default), Stratified splitting (important for rare events, you take random samples), Time-based split (for time series)

Learning Curves: Accuracy increases with more data, but eventually plateaus

Model Types

Two types of prediction problems:

1. Regression problems: Predict **continuous values**. Example: house price

2. Classification: Predict **binary outcomes**. Example: Fraud: yes/no. Very common in business

Evaluating Regression Models

Key difference from classification:

- Classification → accuracy can be captured in terms of errors; correct vs incorrect
- Regression → always some error

Instead of counting errors, focus on **magnitude** of difference between predicted and realized values ($\hat{y}_n - y_n$).

Common metrics: Root mean squared error (RMSE), Mean absolute error (MAE), Mean absolute percentage error (MAPE). You want them as low as possible.

RMSE (Root Mean Squared Error)

- Sum square errors → average → square root
- Same unit as data itself
- Sensitive to outliers

$$RMSE = \sqrt{\frac{\sum_{n=1}^{N}(\hat{y}_n - y_n)^2}{N}}$$

MAE (Mean Absolute Error)

- Average absolute error
- Easier to interpret
- Less sensitive to outliers

$$MAE = \frac{1}{N}\sum_{n=1}^{N}|\hat{y}_n - y_n|$$

Trade-off: RMSE penalizes large errors more

MAPE (Mean Absolute Percentage Error)

- Error relative to actual value
- Advantage: Easy to interpret (percentage)
- Problems:
    - Undefined if actual value = 0
    - Very large errors when actual values are small
    - Asymmetry: Max underprediction = 100%. Overprediction = unlimited

$$MAPE = \frac{1}{N}\sum_{n=1}^{N}\left|\frac{(\hat{y}_n - y_n)}{y_n}\right|$$

Important lesson: Different models and metrics → different conclusions

R^2

Formula: $R^2 = 1 - (RSS/TSS)$ →

$$R^2 = 1 - \frac{\sum_{n=1}^{N}(y_n - \hat{y}_n)^2}{\sum_{n=1}^{N}(y_n - \bar{y})^2}$$

- Interpretation: % of variance explained by the model
- In principle, just a normalized version of the mean squared error
- Gives same ranking as RMSE when comparing models

Evaluating classification models

Many business decisions are based on classification problems that entail classification errors, e.g.,

- Start a fraud investigation? (Might not be a fraud case.)
- Invite a job applicant? (Turns out not to be qualified.)

Key question: How do we evaluate classification models?

Accuracy

In a classification model, you can have correct predictions and errors:

- True positives (TP), True negatives (TN)
- False positives (FP), False negatives (FN)

$$\boldsymbol{Accuracy} = \frac{correct\ predictions}{total\ predictions} = 1 - error\ rate$$

Problem: Too simplistic. Misleading for imbalanced data
Example: If only 0.1% buy → predict "no" always → 99.9% accuracy. But useless model

Confusion Matrix

Definition: Table comparing predicted vs actual classes Structure:

	Actual Positive	**Actual Negative**
Predicted Yes	True Positive	False Positive
Predicted No	False Negative	True Negative

→ Shows types of errors (type 1/type 2) explicitly

Classification Metrics derived from confusion matrix:

True Positive Rate: Frequency of true positive → *TP/(TP + FN)*
False Negative Rate: Frequency of false negative → *FN/(TP + FN)*
• Analogous: True Negative Rate ("Specificity")/False Positive Rate

Precision: Accuracy over cases predicted to be positive ($TP/(TP + FP)$), also called "Positive Predictive Value"
F1-score: $2 * \frac{Precision * Recall}{Precision + Recall}$

Why so many metrics? → Different contexts → different costs of errors

Expected Value

Idea: Evaluate model based on **economic value**
General formula: EV = $p(o_1) \times v(o_1) + p(o_2) \times v(o_2) + p(o_3) \times v(o_3) + \ldots$
Idea: combine the probability of outcome with the value (profit/cost)

Expected Value (Visual)

1. Data → model
2. Model → confusion matrix
3. Combine with: probabilities, costs/benefits
4. → expected value

Expected Value (Formula)

$$p(p) \times [p(Y|p) \times b(Y,p) + p(N|p) \times b(N,p)] + p(n) \times [p(N|n) \times b(N,n) + p(Y|n) \times b(Y,n)]$$

Breakdown:

- The "class priors" $p(p)$ and $p(n)$ are either (approximately) known or can be taken from the confusion matrix (when the testing data is representative)
- $p(Y|p)$ etc. can be taken from the confusion matrix
- $b(Y,p)$ etc. need to be taken from other information sources outside the model

Expected Value Example (Marketing)

Scenario: Decide whether to send a marketing offer. Given:

- Cost per offer = €1
- Profit if customer responds = €99
- Dataset: 61 positives (buyers), 49 negatives

Confusion matrix: TP = 56, FP = 7, TN = 42, FN = 5

→ Goal: compute **expected profit**

Actual / Predicted	Positive (p)	Negative (n)	Total
Positive (Y)	56	7	63
Negative (N)	5	42	47
Total	61	49	110

Steps:

- Compute probabilities:
  - $p(p)$ = 61/110 = 0.55, $p(n)$ = 49/110 = 0.45
  - $p(Y|p)$ = 56/61 = 0.92, $p(N|p)$ = 5/61 = 0.08
  - $p(N|n)$ = 42/49 = 0.86, $p(Y|n)$ = 7/49 = 0.14
- Plug into expected value formula:
  - 0.55 × 0.92 × 99€ + 0.08 × 0€+ 0.45 × [0.86 × 0€ + 0.14 × -1€] ≈ **50. 04€**
- Final result:
  - Expected profit ≈ €50.04

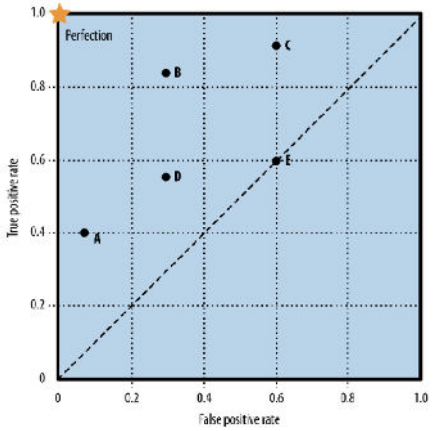
Visualizing evaluation methods

Receiver Operating Characteristic (ROC) Graph

= A graphical approach to show and compare the performance of different confusion matrices

- X-axis: False Positive Rate $(FP/(FP + TN)$
- Y-axis: True Positive Rate $(TP/(TP + FN)$

Shows trade-off: More true positives ↔ more false positives



ROC Graph Interpretation

- (0,0): Predict everything negative; No FP, no TP
- (1,1): Predict everything positive
- (0,1): Perfect classification
- Diagonal line (0,0 → 1,1): Represents random guessing
  - Guess 50% positive → (0.5, 0.5)
  - Guess 90% positive → (0.9, 0.9)
  - Thus, a random classifier moves on the diagonalbased on the frequency with which it guesses thepositive class.Seminar Data Analytics

Comparing Classifiers

Key rule: A model is better if it is more north-west (away from the diagonal). This gives a higher TPR and a lower FPR.

Conservative vs Permissive Models

Classifiers appearing on the lefthand side of a ROC graph, near the x axis, may be thought of as "**conservative**": they raise alarms only with strong evidence, so they have low false positive, but they have low true positive rates as well.

Classifiers on the upper righthand side of a ROC graph may be thought of as "**permissive**": they make positive classifications with weak evidence, so they have high true positive, but they often have high false positive rates.

Imbalanced Data Insight

In many real cases: many more negatives than positives
Therefore: Left side of ROC curve is often most important

Ranking Classifiers

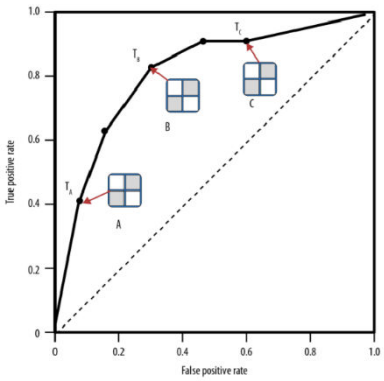
Key idea: Classification is based on **probabilities/scores** and a threshold
But: You can change threshold → changes confusion matrix and model performance

Threshold Example: 100 positives + 100 negatives, Sorted by predicted probability
As threshold decreases: More observations classified as positive

ROC Curve Construction Steps:

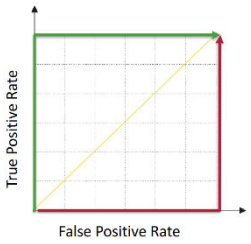
1. Sort observations by prediction score
2. Start with highest threshold → (0,0)
3. Gradually lower threshold
4. Each step:
    - One more predicted positive
    - New confusion matrix
    - New point in ROC space

Final point: (1,1) → everything classified positive



The ROC "curve" is a step function, but appears smooth with enough test instances

Perfect vs Worst Model



→ Ideal curve hugs **top-left corner**

AUC (Area Under Curve)

Definition: Area under ROC curve

- Its value ranges from zero (worst imaginable model) to one (perfect model).

Interpretation: Probability that a random positive is ranked above a random negative

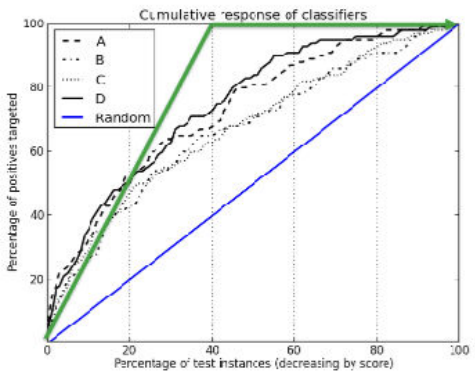
Also equivalent to: Mann-Whitney test and Gini coefficient (transformed)

Cumulative Response Curve

Alternative to ROC (more intuitive).

→ Plots true positive rate against the percentage of the testing data that's being predicted as positive when setting the threshold as the top X% of the score.

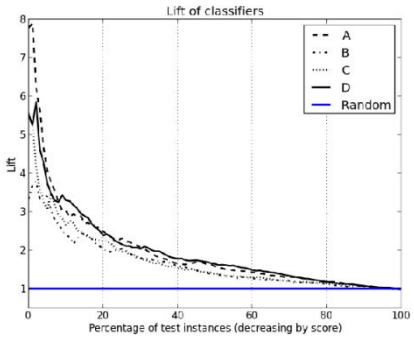
- X-axis: % of population targeted
- Y-axis: % of true positives captured



Lift

Lift measures improvement over random:
Lift = cumulative response curve at a given point / diagonal line value at that point
Example: Lift = 2x → model is twice as good as random

Lift Curve



- X-axis: % of population targeted
- Y-axis: lift

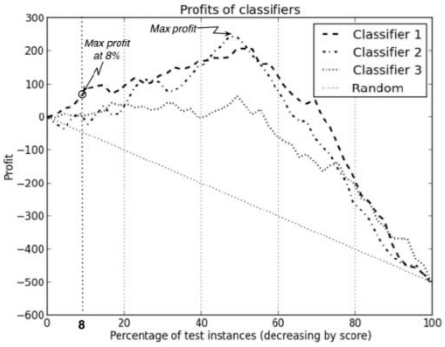
Interpretation:

- Higher curve = better model
- Lift decreases as more population is included

Profit Curves

Use expected value framework
Purpose: Compare models based on profit**,** Choose optimal threshold
Steps: Rank observations, Evaluate profit at each threshold

Profit Curve Example



- Each model has different profit curve
- Random model = straight downward line

Best model depends on: Budget constraints and Threshold

Profit Curve Interpretation

- All curves begin and end at the same point.
- Left side: No one targeted → no cost, no profit
- Right side: Everyone targeted → max cost + max profit
- Middle: Optimal point

Example: Best model classifier depends on: % of customers you can target

Limitations of Profit Curves

Challenges:

1. Need class probabilities (priors)
2. Need cost/benefit values

Problems: Often unknown, Hard to estimate, Hard to communicate