Data Analysis and Knowledge Discovery

Lecture 4

Faculty of Mathematics and Computer Science
Babeș-Bolyai University

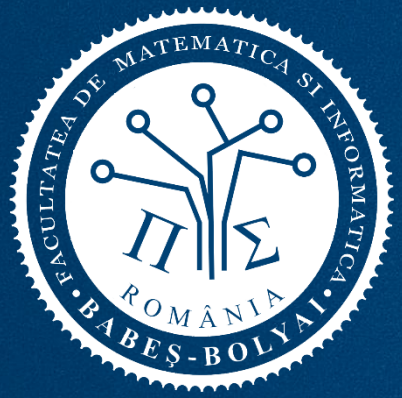
Sergiu Limboi, PhD Teaching Assistant

Motto: “Better features beat better algorithms.”

Feature Engineering in the Knowledge Discovery Pipeline

AGENDA

- Warm-Up
- The Role of Feature Engineering in Knowledge Discovery
- What is a Feature?
- Dimensionality Reduction
- Feature Selection
- Feature Extraction
- Handling Imbalanced Datasets
- Feature Importance
- Feature Construction
- Key Takeaways



Warm-Up

Faculty of Mathematics and Computer Science

Warm-Up

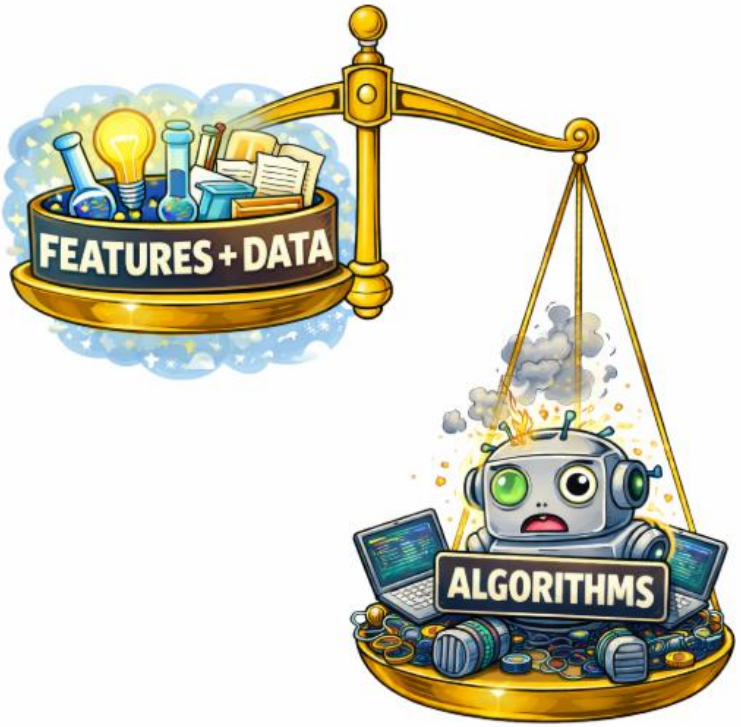
Go to www.menti.com and enter the code **8882 5171**

or use the QR code

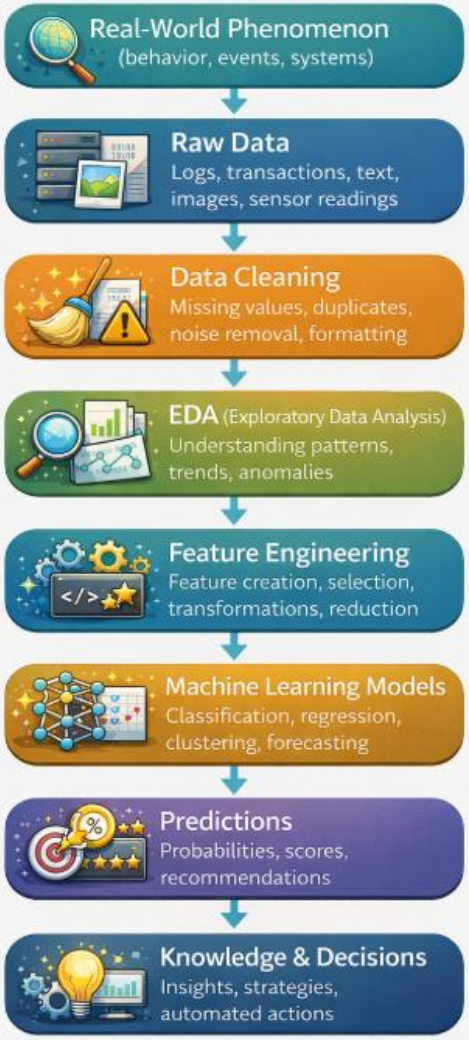
The Role of Feature Engineering in Knowledge Discovery

Industry reality

- In data science projects



The Role of Feature Engineering in Knowledge Discovery



The Role of Feature Engineering in Knowledge Discovery

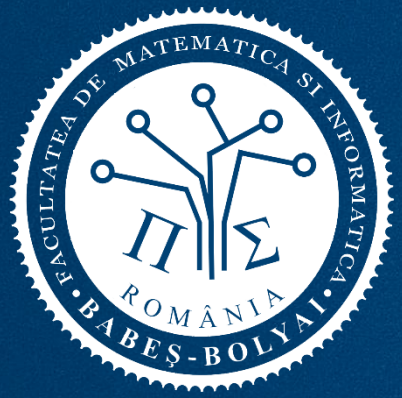
- Fraud detection dataset

Timestamp	Merchant	Amount
2025-03-01 02:14	Amazon	200$

- This is not enough for a model
- Engineered featured:

hour_ of_day	Avg_spending_last_24h	Distance_from_home

- Now patterns become visible
- Feature engineering is the process of constructing new input features, transforming existing features, and selecting relevant features in order to improve the performance of machine learning models.



What is a Feature?

What is a feature?

- A **feature** is a measurable attribute used as input for a machine learning model.

Age	Salary	Number_of_purchases

- Each column is a feature.
- Some features may be poor representation of reality.
- Example: predicting house price.
  - Bad feature→ house ID
  - Good features:
    - House age
    - Distance to city centre
    - Price per square meter

What is a Feature?

Age	Salary	Purchases
25	3000	2
40	7000	10

- Row = observation
- Column = feature
- A machine learning model sees data as a matrix of features.
- Machine learning models do not understand reality.
- They only see numbers.

What is a Feature?

- A model cannot understand names.

- But features like:
  - number_of_logins_last_30_days
  - average_session_time
  - days_since_last_login

MEANINGFUL !

The machine learning model learns patterns like:
Low activity → high probability of churn (leaving)

Dimensionality Reduction

Dimensionality Reduction

- Challenges in machine learning models
  - Large datasets often contain many features
  - Leads to higher computation time and overfitting
    - Overfitting occurs when a machine learning model learns the training data too well, including noise and/or irrelevant details, instead of learning the true underlying patterns.

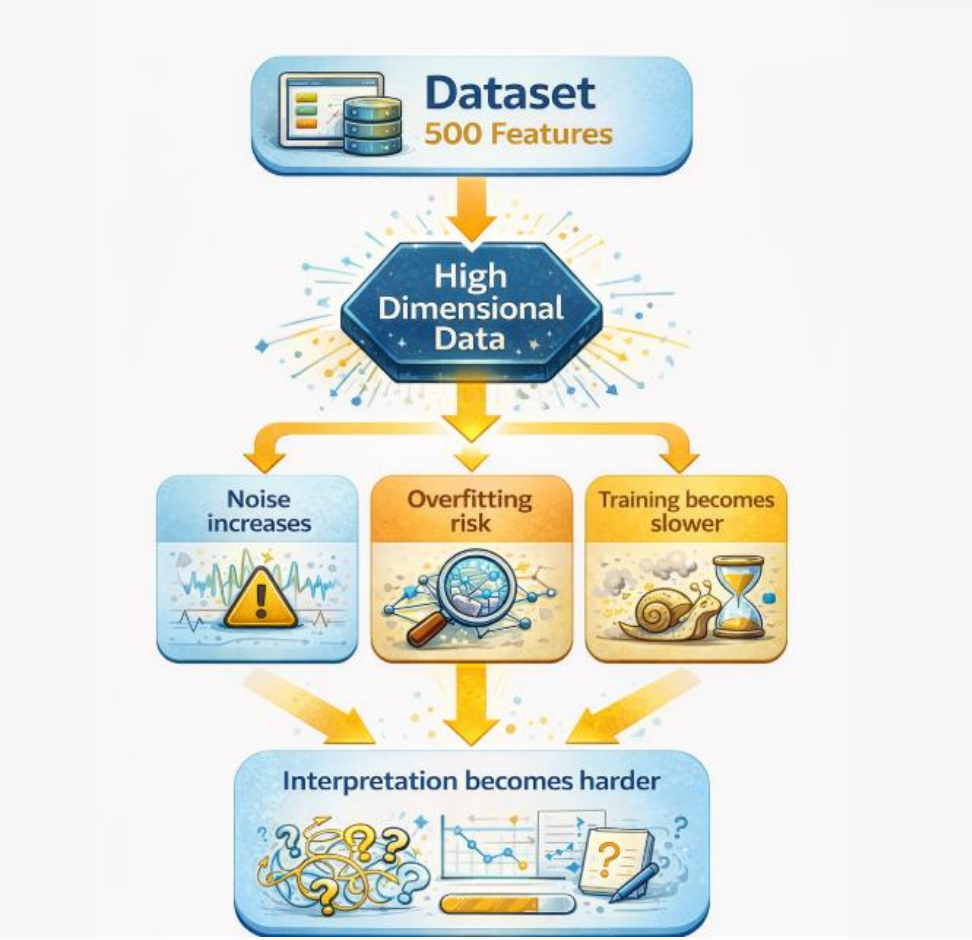
- Too many features, too many problems
  - Slower model training
  - Higher risk of overfitting

Problem: Too Many Features

- Medical dataset:
  - patient age
  - weight
  - blood
  - gene expression
  - cholesterol
  - ...
  - And 500 more variables...

Problem: Too Many Features

The curse of dimensionality



Dimensionality Reduction

- Solution?
  - Dimensionality reduction
    - Reduces the number of features in a dataset
    - Keeps the essential information
    - improves model performance and efficiency

- Instead of using 500 features, we transform the dataset into 10–20 new features that still capture most of the data variability.

Why reduce dimensionality?

- Faster computation
  - Fewer features lead to quicker model training and testing, especially for large datasets
- Better visualization
  - Easier to visualize data in 2D or 3D→ helps reveal hidden patterns
- Prevent overfitting
  - Simpler models are less likely to memorize noise, improving performance on new data

Challenges of Dimensionality Reduction

- Data loss & reduced accuracy
  - Important information may be lost, potentially affecting model performance
- Interpretability challenges
  - Transformed features (like Principal components) may lack clear meaning, making insights harder to understand
- Choosing the right component
  - Deciding the number of dimensions to keep:
    - Too few→ loss of information
    - Too many→ Overfitting

Types of Dimensionality Reduction

Method	Idea
Feature selection	Keeps the best original features
Feature extraction	Create new features from old ones

Feature Selection

Feature Selection

- Selects a subset of relevant features from the original set of features.
- Reduces dimensionality while keeping original features.
- Why?
  - Datasets today may contain hundreds or thousands of variables.
- Enhances interpretability and reduces overfitting.
- Requires domain knowledge and feature engineering.
- May lose useful information if important features are removed.

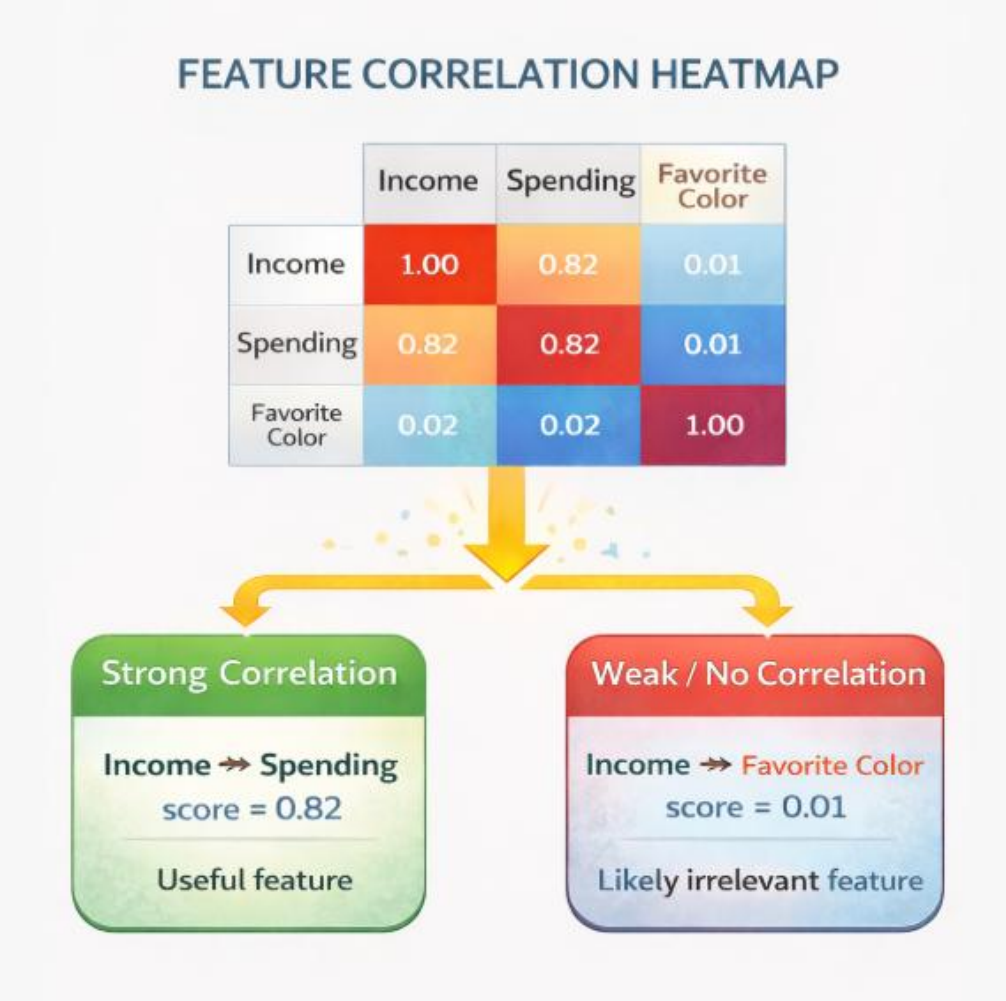
Feature selection techniques: Filter methods

- Filter methods select features based on their statistical relationship with the target variable.
- They evaluate each feature independently without involving machine learning algorithms.
- How it works:
  - Ranks features according to statistical scores like correlation, chi-square, or mutual information.
  - Selects features with the highest scores.

Feature selection techniques: Filter methods

- Pros:
  - Simple and fast
  - Independent of model training
  - Suitable for vary large datasets
- Cons:
  - Ignores feature dependencies
  - Might select irrelevant features if the correlation is weak

Feature selection techniques: Filter methods



Feature selection techniques: Filter methods

```python
import pandas as pd

# Example dataset
data = {
    "size": [50, 60, 80, 120, 150],
    "rooms": [2, 3, 3, 4, 5],
    "distance_to_city": [10, 8, 6, 3, 2],
    "favorite_color": [1, 2, 3, 1, 2],
    "price": [150, 200, 250, 400, 500]
}

df = pd.DataFrame(data)

# Compute correlation with target variable
correlation = df.corr()

print(correlation["price"])
```

```python
selected_features = correlation["price"].abs().sort_values(ascending=False)

print(selected_features)
```

Feature selection techniques: Wrapper methods

- Wrapper methods evaluate subsets of features by training a model on different feature combinations and selecting the best-performing subset.

- How it works:
  - Iteratively adds or remove features
  - Use the model's performance (like accuracy) as the selection criterion

- Pros:
  - Provides better accuracy
  - Considers feature dependencies

- Cons:
  - High computational cost
  - Prone to overfitting on small datasets

Feature selection techniques: Wrapper methods

- **Recursive Feature Elimination (RFE)** is a wrapper feature selection method that selects the most important features by training a model repeatedly and removing the least important features step by step.

- Train a model → rank features → remove the weakest feature → repeat until only the best features remain.

Recursive Feature Elimination (RFE)

Recursive Feature Elimination (RFE)

```python
from sklearn.feature_selection import RFE
from sklearn.linear_model import LogisticRegression

model = LogisticRegression()

rfe = RFE(model, n_features_to_select=3)

rfe.fit(X, y)

print("Ranking:", rfe.ranking_)
print("Selected features:", rfe.support_)
```

RFE ranks features using the importance values computed by the model, then repeatedly removes the least important feature.

If a feature has **importance = 0**, what should happen?

Remove it immediately !!!

Feature selection techniques: Embedded methods

- Embedded methods perform feature selection during model training.
- These methods automatically select features based on their importance within the model itself.
- Hot it works:
  - Feature selection happens while the model is trained
  - Techniques like Decision trees or Lasso regression automatically remove less important features.
- Pros:
  - Efficient and accurate
  - Automatically selects features
  - Considers features dependencies
- Cons:
  - Depends on the model
  - May not work well with all algorithms

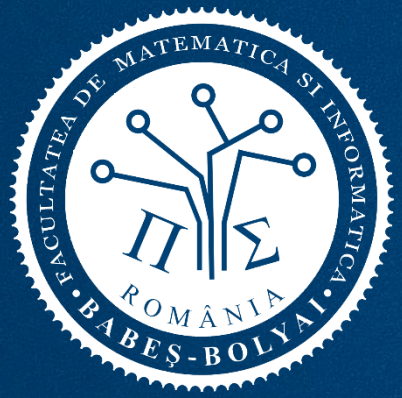
Feature selection techniques: Embedded methods

```
from sklearn.linear_model import Lasso

# Train Lasso model
lasso = Lasso(alpha=0.1)
lasso.fit(X, y)

# Print feature importance
print("Feature Coefficients:", lasso.coef_)
```

```
Feature Coefficients: [ 0.         -0.          0.40811896  0.        ]
```

Feature selection: comparison

Aspect	Filter Methods	Wrapper Methods	Embedded Methods
Idea	Use statistical tests to evaluate features	Evaluate subsets of features using a model	Feature selection occurs during model training
When selection happens	Before training the model	During repeated model training	Inside the training algorithm
Example techniques	Correlation, Chi-square, Mutual Information	Recursive Feature Elimination (RFE), Forward Selection	Lasso (L1), Decision Trees, Random Forest
Model required?	✗ No	✓ Yes	✓ Yes
Computational cost	Low	Very high	Medium
Speed	Very fast	Slow	Moderate
Considers feature interactions?	✗ No	✓ Yes	✓ Yes
Works well for large datasets?	✓ Yes	✗ Often too expensive	✓ Usually
Risk of overfitting	Low	Higher	Moderate
Typical use case	Quick feature filtering	Finding optimal subset of features	Automatic feature selection during training



Feature Extraction

Feature Extraction

- It creates new features by combining and transforming original features to reduce dimensionality.

- Why use feature extraction?
  - Improves model performance
  - Reduces noise and complexity
  - Helps in better visualization of data

- Reduces dimensionality by transforming data into a new space.

- Can be applied on raw data

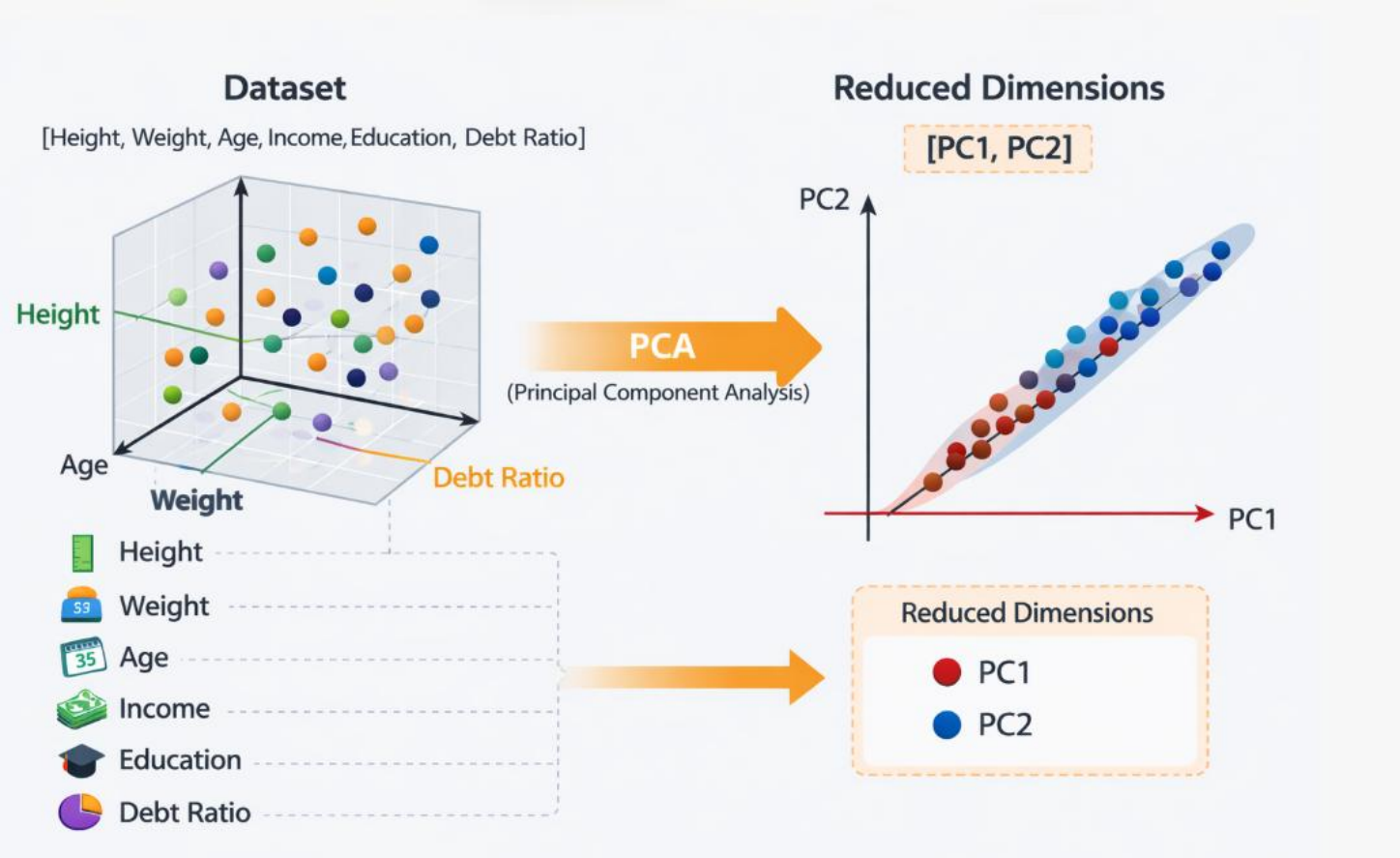
Principal Component Analysis (PCA)

- Principal Component Analysis (PCA) is a dimensionality reduction technique that transforms the original dataset into a smaller set of linearly uncorrelated features called principal components.

- The first principal component captures the maximum variance in the data, followed by the second, and so on.

- Instead of working with many correlated variables, PCA creates new variables.

Principal Component Analysis (PCA)



Principal Component Analysis (PCA)

- The diagonal shape appears because many features in the dataset are correlated.
- So the data varies mostly in one dominant direction.
- PCA detects that direction.
- That direction becomes:PC1
- PC1 → main data direction
  PC2 → secondary variation

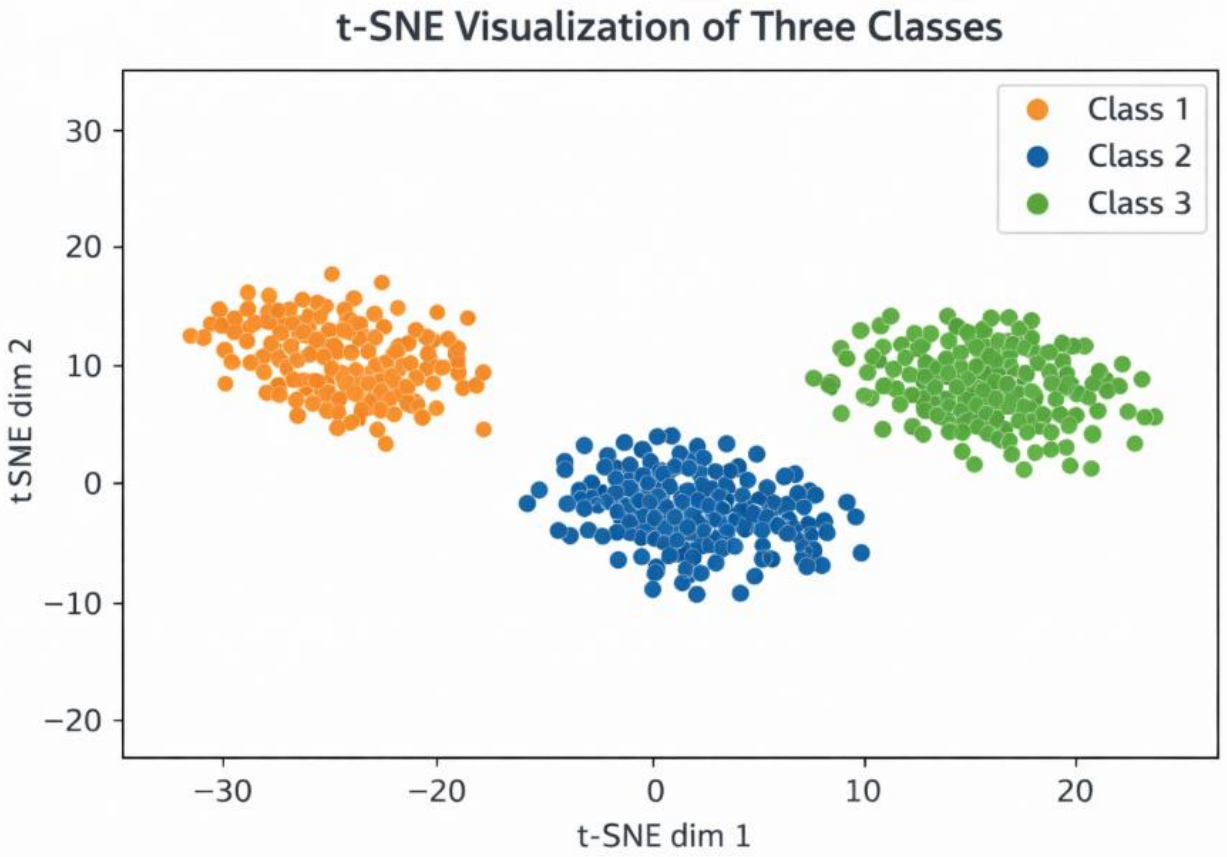
Principal Component Analysis (PCA)

- **Advantages**
  - simple and efficient
  - reduces dimensionality
  - removes correlated features
- **Limitations**
  - linear method
  - difficult to interpret new features

t-SNE (t-Distributed Stochastic Neighbor Embedding)

- t-SNE is a nonlinear dimensionality reduction method designed for visualizing high-dimensional data.
- It tries to preserve local relationships between points.
- Meaning: Points that are close in high-dimensional space remain close in the reduced space.
- t-SNE models similarity between points as **probabilities**.
- If two points are similar →high probability
- t-SNE tries to keep those probabilities similar in lower dimensions.

t-SNE (t-Distributed Stochastic Neighbor Embedding)



t-SNE (t-Distributed Stochastic Neighbor Embedding)

- **Advantages**
  - excellent for visualization
  - reveals clusters in complex datasets

- **Limitations**
  - slow for large datasets
  - not good for general dimensionality reduction in models
  - distances between clusters may be misleading

UMAP (Uniform Manifold Approximation and Projection

- UMAP is a modern nonlinear dimensionality reduction technique.

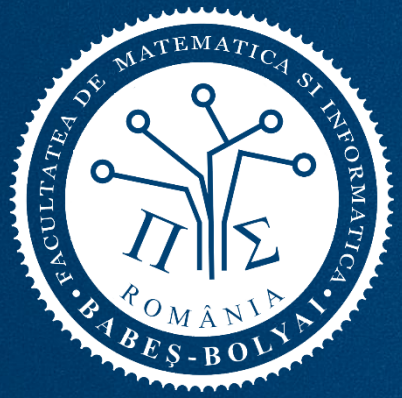
- It is similar to t-SNE but:
  - faster
  - better preserves global structure
  - scalable to larger datasets

- Even though there are 100 variables, the data might actually lie on a 10-dimensional structure.

- Why? Because many features are **correlated or constrained**.

Autoencoders

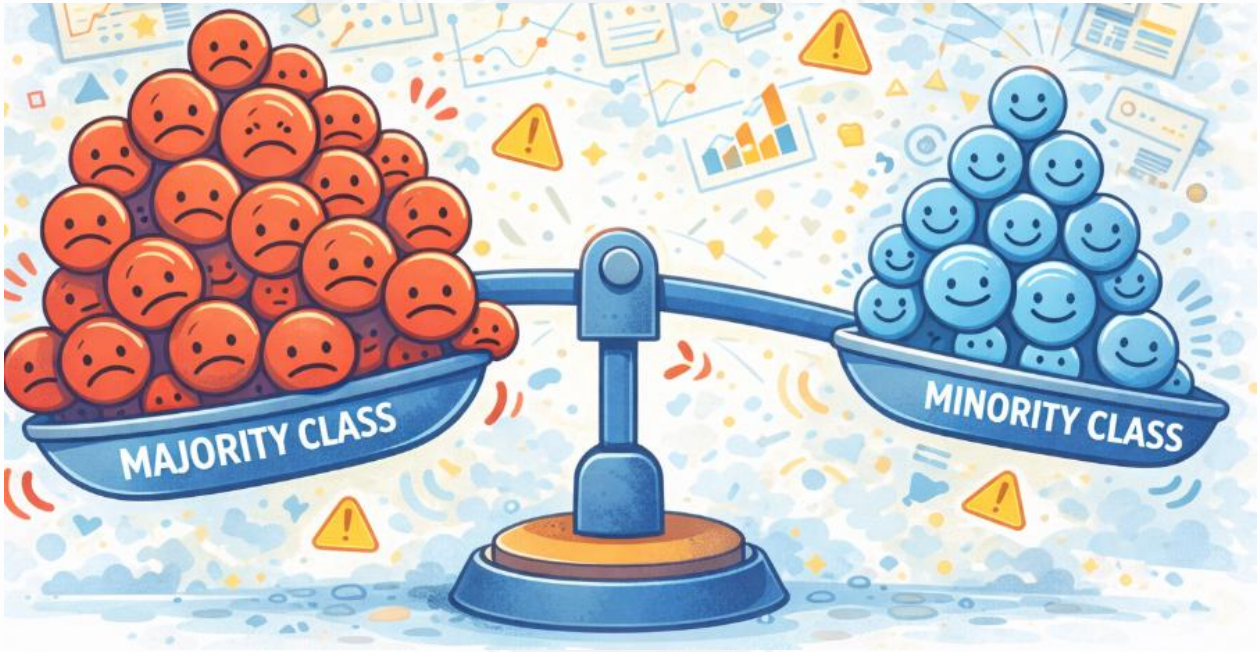
- Autoencoders are neural networks designed to compress data.
- Encoder: Compresses the input into a lower-dimensional representation
- Decoder: Reconstructs the original input from the lower-dimensional code



Handling Imbalanced Datasets

Handling Imbalanced Datasets

- Many real-world datasets are extremely imbalanced.



Oversampling (Increase minority class data)

- Oversampling increases the number of samples in the minority class.

- Original dataset:
  - Class A (majority) → 9800 samples
  - Class B (minority) → 200 samples

- After oversampling:
  - Class A → 9800
  - Class B → 9800

- Random Oversampling → It works by **duplicating existing minority samples**.
  - Problem: risk of overfitting

More Advanced Oversampling: SMOTE

- SMOTE (Synthetic Minority Over-sampling Technique) → Instead of duplicating data, SMOTE generates new synthetic samples.

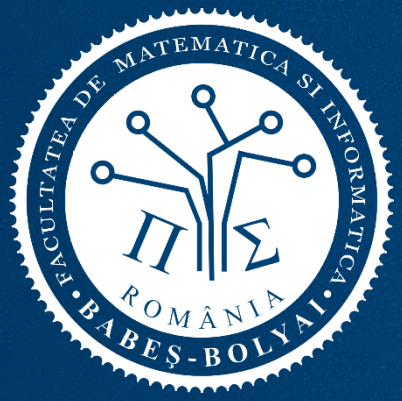
- How SMOTE works?
  - Selects a random minority class instance
  - Chooses one of its nearest neighbours
  - Generates a synthetic instance between the selected sample and its neighbour

- Oversampling is useful in problems like:
  - fraud detection
  - disease diagnosis
  - credit default prediction
  - anomaly detection

  - In these tasks the **important class is rare**.

Undersampling (Reduce Majority Class Data)

- Undersampling removes samples from the majority class.
- Original dataset:
  - No Fraud → 9800 observations
  - Fraud → 200 observations
- After undersampling:
  - No Fraud → 200
  - Fraud → 200
- Random undersampling:
  - Identify the majority class.
  - Randomly remove samples until classes become balanced.
- More advanced techniques:
  - Cluster centroids→ each majority class cluster is replaced by its centroid.
  - NearMess→ Majority points near minority clusters are retained.



Feature Importance

Feature Importance

- Feature importance refers to techniques used to assign a score to input features (independent variables) based on how useful or valuable they are in predicting the target variable (dependent variable).

- The concept is widely used in machine learning to:
  - Understand which features contribute the most to the model's predictions
  - Improve model interpretability
  - Perform feature selection to reduce dimensionality, improve model performance, and avoid overfitting.

Feature Importance

Age	Income	Credit score	Debt	Favorite colour

Feature	Importance
Credit score	0.40
Income	0.30
Debt	0.20
Age	0.10
Favorite colour	0.00

- Credit score→ very important
- Favorite colour → irrelevant

Model-based Feature Importance Technique

- The model assigns a weight (coefficient) to each feature
- Example: Price = 2.5 * Size + 1.8 * Rooms + 0.3 * Distance
- Size → strong influence
- Distance→ small influence
- Used in linear regression, logistic regression.

Tree-based Feature Importance Technique

- Features are important if they help split the data and reduce prediction error.
- Example: credit score > 700?
- Used in decision trees, random forest, gradient boosting.
- These algorithms provide feature importance by measuring how much each feature contributes to reducing impurity (e.g., entropy, gini impurity).
- Impurity is a concept used in decision trees to measure how mixed the classes are in a dataset or node.
  - A node is pure if it contains only one class.
  - A node is impure if it contains multiple classes.

Permutation-based Importance

- Permutation importance measures importance by breaking the relationship between a feature and the target.

- Steps:
  - Train the model.
  - Shuffle one feature.
  - Measure how much the model performance drops.
    - Large drop→ important feature
    - Small drop → irrelevant feature

Feature	Accuracy Drop
Credit score	-15%
Income	-8%
Age	-2%
Favourite colour	0%

Feature Construction

Feature Construction

- Feature construction is the process of creating new features from existing data in order to better capture patterns relevant to prediction tasks.

- Example: we have a dataset with height and weight
  - We build a new feature BMI (body mass index) = weight / height

- Feature aggregations:
  - Dataset with user id and purchase amount
  - Aggregated features: max_purchase, average_spending, total_spending, etc

Feature Construction

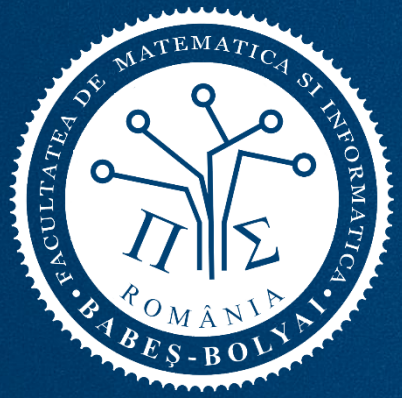
- Time-based Features
  - Timestamp
    - Hour of day
    - Month
    - Is weekend
    - Is holiday
    - Etc

- Interaction features:
  - Dataset with house size and location
    - Large house +city centre → very expensive house feature
    - Lage house + rural area→ cheap house feature

Feature Construction

- Domain-knowledge features
  - Often the most powerful features come from domain expertise.
  - Dataset with temperature and timestamp features
    - Season
    - Cooling degree days
    - Heating degree days

- Feature engineering is **creative problem solving**.



Key Takeaways

Faculty of Mathematics and Computer Science

Key Takeaways

- Feature engineering is one of the most important steps in data science
- Good features capture meaningful patterns in data
- Too many features can harm models
- Feature engineering connects data understanding with modelling
- Feature engineering includes several key processes
  - Feature creation (constructing new variables)
  - Feature selection (removing irrelevant variables)
  - Dimensionality reduction (compressing high-dimensional data)
  - Feature transformation (e.g., scaling)

Thank you for your attention — questions, thoughts, or challenges?

FACULTY OF MATHEMATICS AND COMPUTER SCIENCE
BABEȘ-BOLYAI UNIVERSITY

1 Mihail Kogălniceanu Street,
Cluj-Napoca, Cluj, România

www.cs.ubbcluj.ro