Data Analysis with AI

Dr.-Ing. Grigory Devadze

What is Artificial Intelligence?

- **Definition and basic concepts:**

  Artificial intelligence (AI) refers to systems or machines that perform tasks that normally require human intelligence - e.g., learning, problem-solving, perception, and language understanding.

- **Difference between AI, Machine Learning, and Deep Learning:**
  - **AI (Artificial Intelligence):** Umbrella term for machines that act “intelligently”.
  - **Machine Learning (ML):** Subfield of AI where systems learn from data to solve tasks without explicit programming.
  - **Deep Learning:** Specialized subfield of ML that works with artificial neural networks.

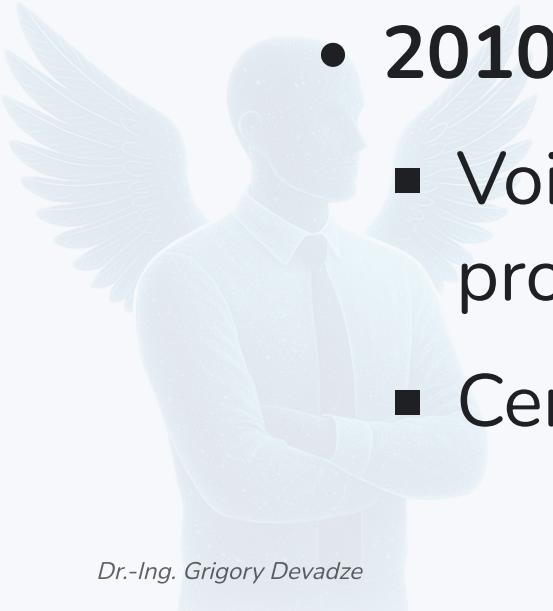
Brief Overview: History of AI

1. From the beginnings to expert systems

- **1956:** Term “Artificial Intelligence” first used.
  - Dartmouth Conference (John McCarthy et al.), goal: enable machines to act intelligently
- **1950s-1970s:** First AI programs
  - Chess programs, theorem proving, lots of optimism, but early setbacks (“AI winter”)
- **1980s:** First expert systems
  - Systems like MYCIN support medical diagnoses, AI use in industry; Limitation: knowledge must be entered manually, low flexibility

2. Modern breakthroughs: Machine Learning and LLMs

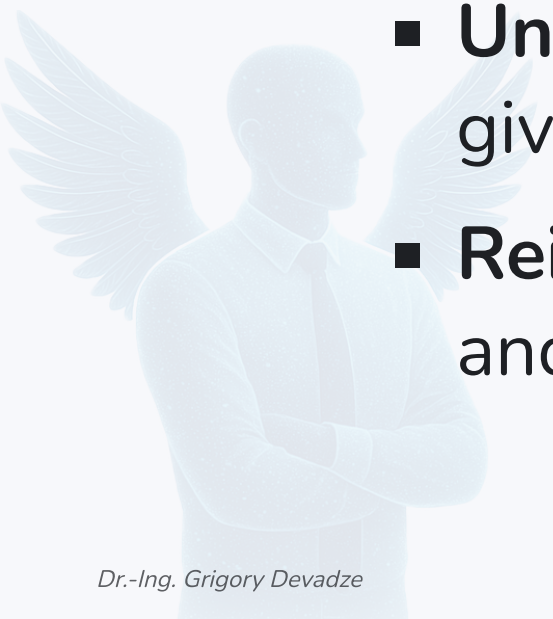
- **2000s:** Breakthroughs with machine learning thanks to powerful hardware
  - More data, better algorithms, “learning from experience”
- **2010s-today:** AI in everyday life
  - Voice assistants (Siri, Alexa), image and face analysis, product recommendations, autonomous driving
  - Central role of deep learning (deep neural networks)



- **LLMs (Large Language Models):**
  - Latest AI development based on deep learning (e.g., GPT, BERT)
  - Can understand and generate natural language
  - Applications: chatbots, text assistance, automatic translation
- AI and LLMs are changing everyday life, business, and research fundamentally and rapidly

How does an AI learn?

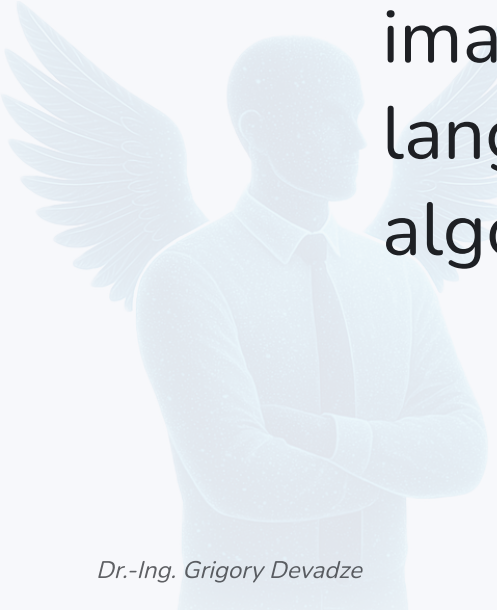
- **Training with data:** Algorithms detect patterns in large data sets.
  - **Supervised learning:** Training with data where the solution is known.
  - **Unsupervised learning:** Finding patterns without given solutions
  - **Reinforcement learning:** Learning through reward and punishment



Potential of AI in data analysis

- **Automated pattern recognition:** AI models independently discover hidden relationships and trends in large data sets that are often not visible to humans.
- **Faster and more precise analyses:** AI significantly accelerates data processing and often delivers more accurate results by minimizing error sources and combining data from different sources.

- **Decision support through predictions:** Predictive analytics with AI enables data-driven decision making (e.g., demand forecasting, failure probability, risk assessment).
- **Processing unstructured data:** AI can also handle text, images, or audio data - for example via natural language processing (NLP) or image recognition algorithms.



Overview of tools and technologies

- **Programming languages:**
  - **Python:** The most widely used language for AI and data-driven analyses, with many powerful libraries.
  - **R:** Often used specifically for statistical analyses and visualization.
- **Development environments:**
  - **Jupyter Notebooks:** For interactive analysis, visualization, and workflow documentation.

- **Frameworks and libraries:**
  - **TensorFlow and PyTorch:** For deep learning and complex neural networks.
  - **Scikit-learn:** For classic machine learning methods (clustering, classification, regression, etc.).
  - **Pandas and NumPy:** For data preparation and processing.
- **Other tools:**
  - **Power BI, Tableau** for visualization
  - **AutoML platforms** to simplify AI workflows
  - **MLOps**: deployment

What is Machine Learning?

- A subfield of artificial intelligence
- Systems learn from data autonomously
- Improvement without explicit programming

Types of ML

1. **Supervised Learning**
    - Labeled data
    - Example: spam detection
2. **Unsupervised Learning**
    - Find structure in unlabeled data
    - Example: customer segmentation
3. **Reinforcement Learning**
    - Learning through reward and punishment
    - Example: AlphaGo

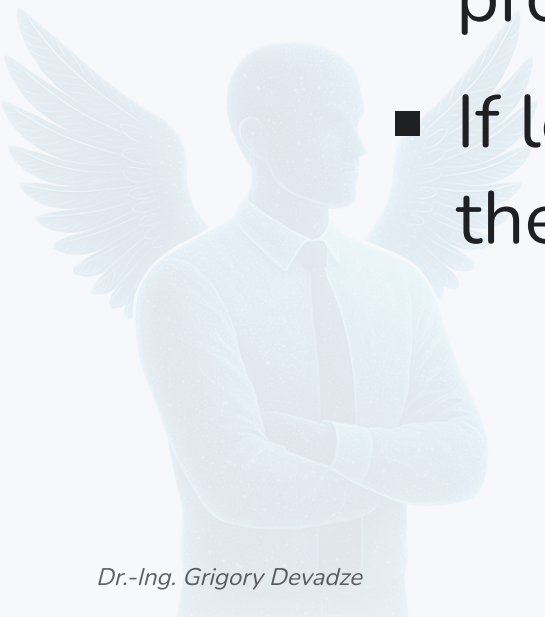
Example: Spam Detection with Machine Learning

How does an ML algorithm detect spam?

- A machine learning model can help classify unwanted emails (spam) by analyzing patterns in the messages.
- Step 1: Data collection and preparation
  - Data sources: emails labeled “spam” or “not spam”
  - Features:
  - Specific words (“Won”, “Free”, “Quick money”)
  - Number of links, use of uppercase letters
  - Sender address, length and structure of the email

- Step 2 - Training a model - Supervised learning:
  - The system is trained on many labeled emails.
  - Goal: learn rules that distinguish spam from legitimate emails.
- Typical approach:
  - Word analysis: which words are typical in spam messages?
  - Probability estimation: how often do certain words appear in spam vs. non-spam emails?
  - Feature weighting: which properties indicate spam most strongly?

- Step 3 - Application - classification of new emails
  - A new email arrives -> model analyzes content -> decision: spam or not spam?
  - If many “spam-indicative” words are present -> high probability of spam
  - If legitimate patterns are detected -> email stays in the inbox



- Step 4: Model optimization
- Avoiding errors:
- False positives -> legitimate emails incorrectly marked as spam
- False negatives -> spam is not detected and lands in the inbox
- Improvement through continuous learning and feedback: users mark emails as “Not spam” or “Report spam”.

Why is machine learning better than fixed rules?

Flexibility and adaptability

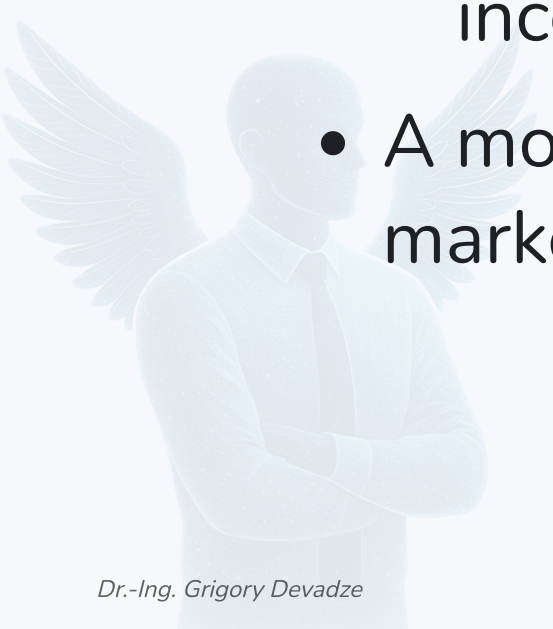
- Rule-based systems:
  - Must be constantly updated by humans when spam tactics change.
  - Example: “Free” used to be a typical spam word - today spammers disguise it as “Fr33” or “F.r.e.e.”.
- ML systems (if they are good):
  - Automatically learn from new spam patterns.
  - Detect altered spellings or new tricks by spammers.

Detection of hidden patterns

- Rule-based approaches:
  - Check only obvious criteria (e.g., “Does the email contain the word ‘casino’?”).
  - Cannot detect complex relationships.
- ML approach:
  - Finds hard-to-detect patterns via statistical analyses.
  - Can identify spam even when no typical spam word is present - e.g., via sentence structure, sender behavior, or similarities to other spam emails.

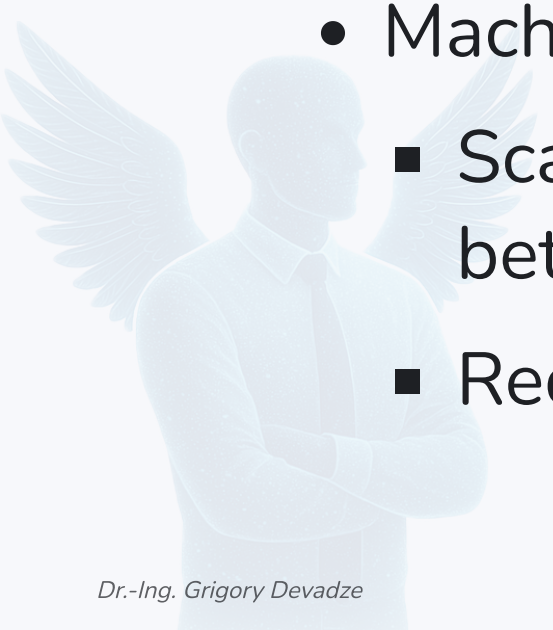
Higher accuracy and fewer errors

- Rules often make errors (“false positives”):
  - Example: A legitimate hotel booking confirmation contains the words “Free WiFi” - it might be incorrectly filtered as spam.
- A model learns to analyze the context: “Is it real hotel marketing or a fake giveaway?”



Scalability and automation

- Rule-based systems:
  - Require human experts who constantly write and test new rules.
  - Become inefficient with high data volumes.
- Machine learning:
  - Scales with the data - the larger the spam set, the better the model becomes.
  - Requires less maintenance after it has been trained.



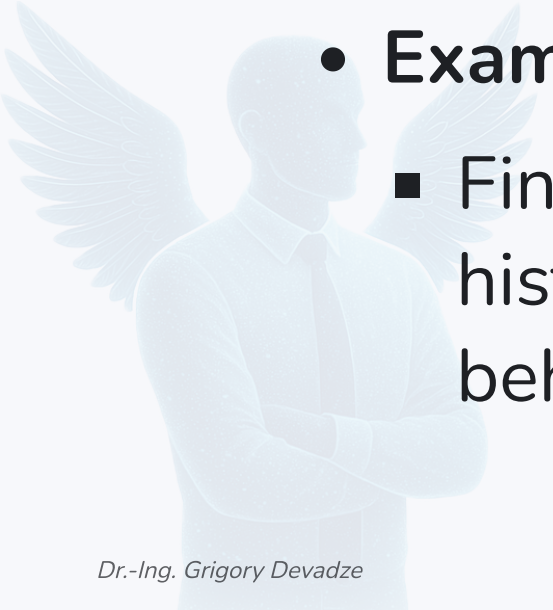
Protection against “smart” spammers

- Spammers constantly develop new tricks, for example:
  - Images instead of text (spam info embedded as an image)
  - Obfuscation (“C.a.s.i.n.o” instead of “Casino”)
  - Malicious links with short URLs (e.g., bit.ly/xyz)
- ML filters are adaptable:
  - Find spam even when it has changed a lot on the surface.
  - Can use artificial intelligence against artificial intelligence.

Challenges

The past cannot always predict the future

- Using historical data to predict the future assumes certain constant or steady-state conditions in a complex system.
- This is almost always wrong when the system involves people.
- **Example**:
  - Financial crises cannot be predicted solely from historical data, because market conditions and human behavior constantly change.



The problem of unknown features

- During data acquisition, the user first defines the variables for which data is collected.
- However, there is always the possibility that critical variables were not considered or even defined.
- **Example**:
  - In medicine, unknown genetic factors or environmental conditions can influence the outcome of a treatment even though they were not in the original data.

Self-destruction of algorithms

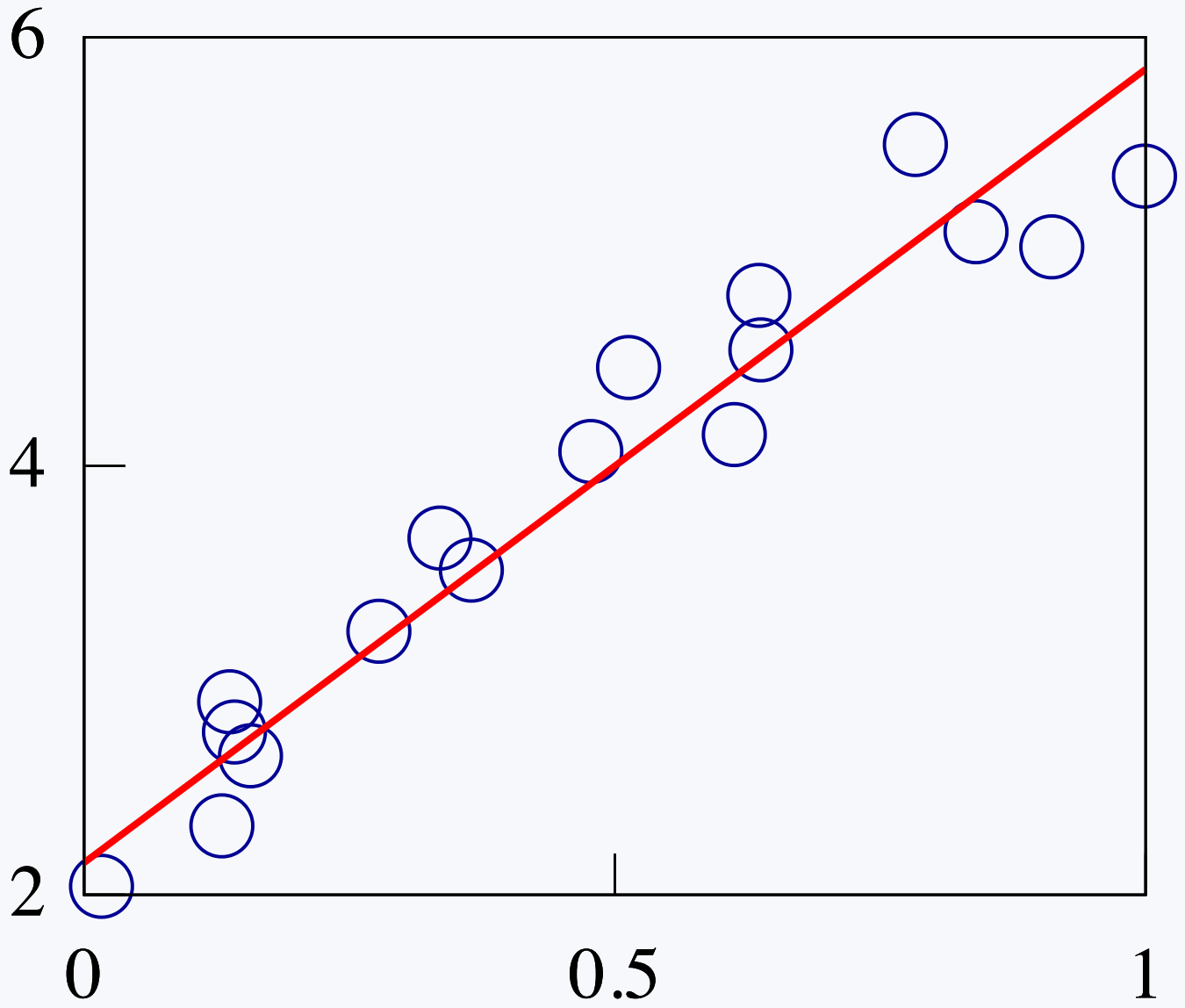
- When an algorithm becomes an accepted standard, it can be exploited by people who understand the algorithm and have an interest in manipulating the result.
- This leads to self-destruction of the algorithm, because it is no longer reliable.
- **Example**:
  - **CDO ratings before the 2008 financial crisis**:
    - Traders manipulated the input variables to obtain AAA ratings for their products, which contributed to the financial crisis.

Linear Regression

Linear regression is a **basic** (statistical) algorithm that models the relationship between a dependent variable (target variable) and one or more independent variables (features).

Important for predictive analytics, e.g.:

- Simplicity and interpretability
- Foundation for more complex models
- Fast model
- Broad applicability


6
4
2
0
0.5
1

What is the idea behind linear regression?

straight line through the data points that best describes the relationship

$$y = m \cdot x + b$$

Meaning of the parameters:

- x = input variable (e.g., advertising budget)
- y = predicted target variable (e.g., revenue)
- m = slope of the line (shows the influence of x on y)
- b = intercept (value of y when $x = 0$)

Multiple linear regression (multiple regression):

If multiple independent variables exist:

$$y = b + m_1 x_1 + m_2 x_2 + \cdots + m_n x_n$$

Example: Sales forecast based on advertising budget, number of stores, and market analyses.

Problem formulation: model equation

The goal is to find the **best** values for m and b!

The most common metric is the mean squared error (MSE).

Goal: make the error as small as possible:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y_i})^2$$

- y_i = actual value of the i-th data sample
- $\hat{y_i}$ = predicted value from the model
- n = number of data points

Because the MSE (mean squared error) loss is a convex function, we can set its derivative directly and solve for the optimal parameters m and b.

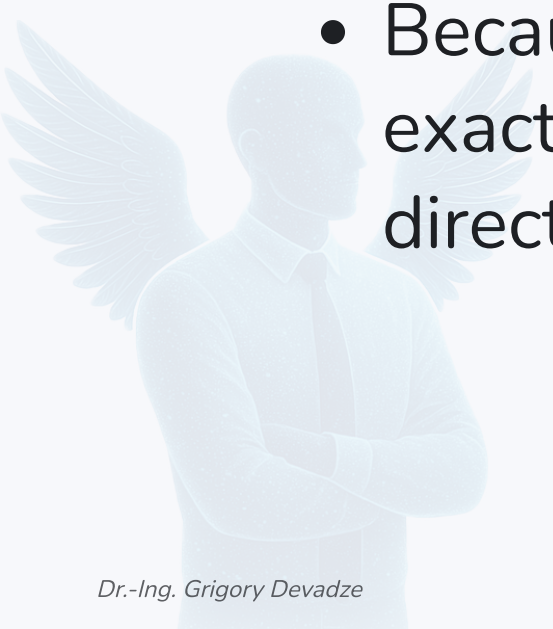
For multidimensional linear regression the optimal solution is:

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

- $\mathbf{X}$ = design matrix (contains all input variables)
- $\mathbf{y}$ = target vector (known values)
- $\mathbf{w}$ = parameter vector (contains weights m and b)

Why is this possible?

- The mean squared error cost function is a quadratic function of the parameters, which allows a simple derivative.
- Because it is a convex optimization problem, there is exactly one unique solution that can be computed directly.



Example: simple linear regression with analytical solution

For a simple regression with $w_1 = m$ and $w_0 = b$ we can directly use the normal equation:

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

This gives the optimal values for m (slope) and b (intercept)

Advantages of the analytical solution

- Exact solution - no approximation needed
- No hyperparameter (e.g., learning rate alpha) required
- Fast to compute for small to medium data sets
  Disadvantages of the analytical solution
- Computationally expensive for large data sets The matrix inversion $(\mathbf{X}^T\mathbf{X})^{-1}$ has a complexity of $O(n^3)$ - very slow for millions of data points.

Alternative: Gradient Descent

Cost function (error metric: Mean Squared Error, MSE)

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y_i})^2$$

Here:

- y_i = actual value
- $\hat{y_i}$ = predicted value using the model equation
  $\hat{y_i} = mx_i + b$
- n = number of data points

Partial derivatives of the MSE cost function

Gradient descent needs the partial derivatives of the error with respect to m (slope) and b (y-intercept)

Partial derivative with respect to m

$$\frac{\partial}{\partial m} MSE = -\frac{2}{n} \sum_{i=1}^{n} x_i (y_i - \hat{y_i})$$

Partial derivative with respect to b

$$\frac{\partial}{\partial b} MSE = -\frac{2}{n} \sum_{i=1}^{n} (y_i - \hat{y_i})$$

These derivatives determine how m and b must be adjusted toward a better model.

Update rule for gradient descent

$$m := m - \alpha \cdot \frac{\partial}{\partial m} MSE$$

$$b := b - \alpha \cdot \frac{\partial}{\partial b} MSE$$

α = learning rate (hyperparameter that determines how large the steps toward optimization are)

Gradient descent uses the derivative of the MSE function to adjust m and b step by step. This repeats until the changes are minimal (convergence).

Basic principle behind Gradient Descent (GD) and Stochastic Gradient Descent (SGD)

Method	Computes gradients on...	Advantages	Disadvantages
Batch Gradient Descent (BGD)	All data points	Very stable and accurate convergence	Computationally intensive for large data sets
Stochastic Gradient Descent (SGD)	**Only one random data point** per step	Very efficient for large data sets	Can fluctuate strongly (high variance)
Mini-Batch Gradient Descent (MBGD)	**Small group of examples (e.g., 32/64 points)**	Balance between GD and SGD, stable and efficient	Requires careful choice of batch size