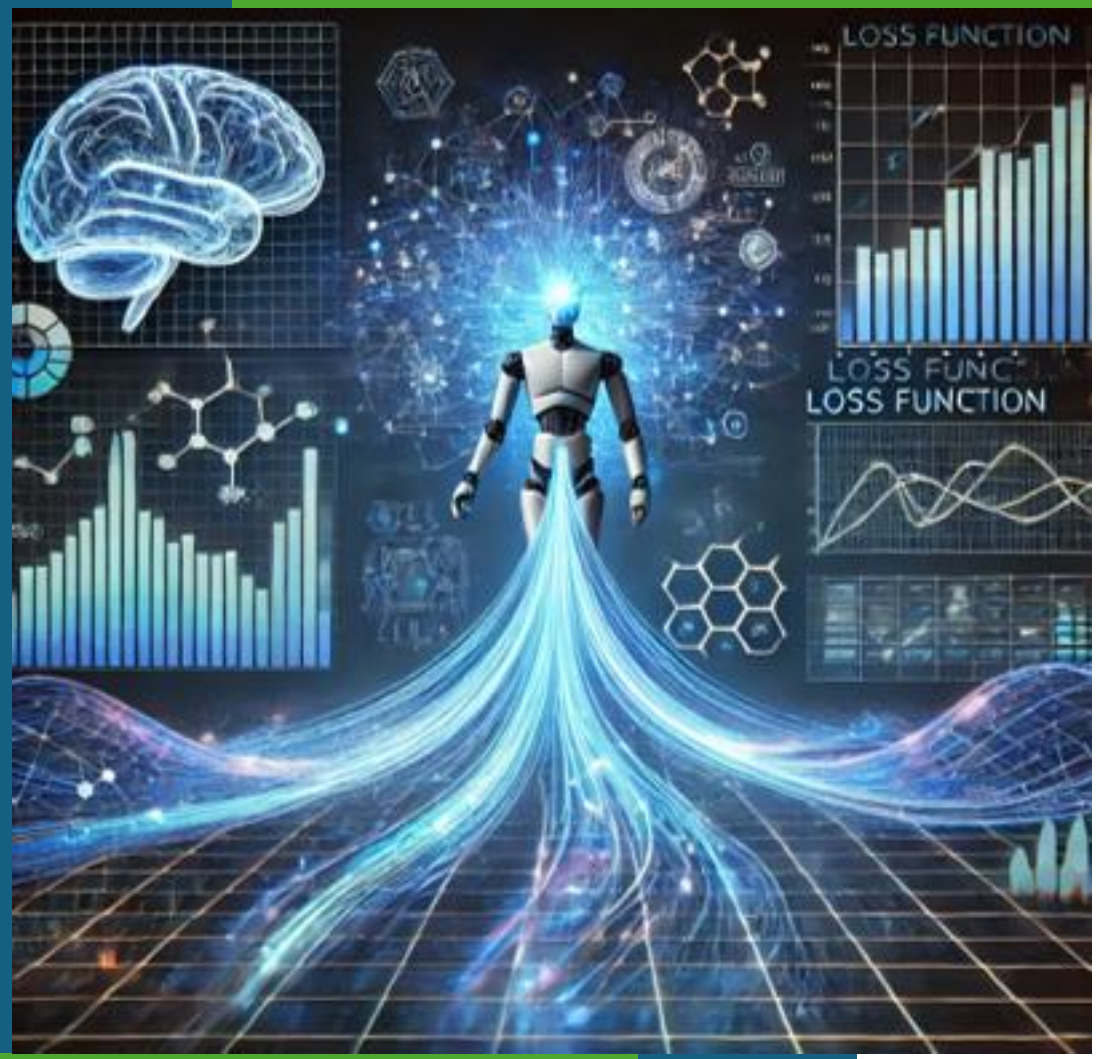
dr inż. Michał Malinowski

https://www.drmalinowski.edu.pl

Temat 8 .

Trening AI – Jak nauczyć maszynę myśleć?

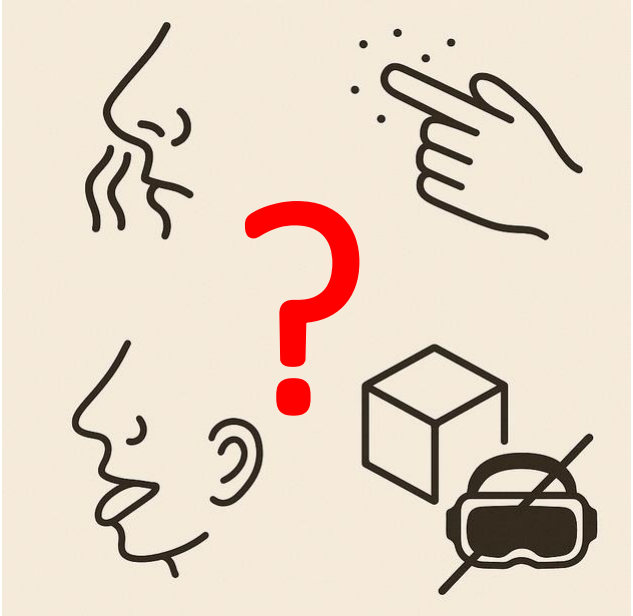
Wprowadzenie do Sztucznej Inteligencji



- AI vs. ML vs. ANN vs. Data Science
- Proces trenowania sztucznej sieci neuronowej
- Epoka i iteracja
- Optymalizacja
- Macierz pomyłek
- Fine-tuning AI
- Retrieval-Augmented Generation (RAG)
- Wytrenowane modele AI
- Jaką architekturę systemu AI wybrać?
- Platformy MLOps

Choć technologia potrafi dziś analizować dane, rozpoznawać obrazy i mówić do nas ludzkim głosem,
nadal nie potrafi przekazać zapachu, smaku, dotyku czy obrazu 3D bez dodatkowych gadżetów.

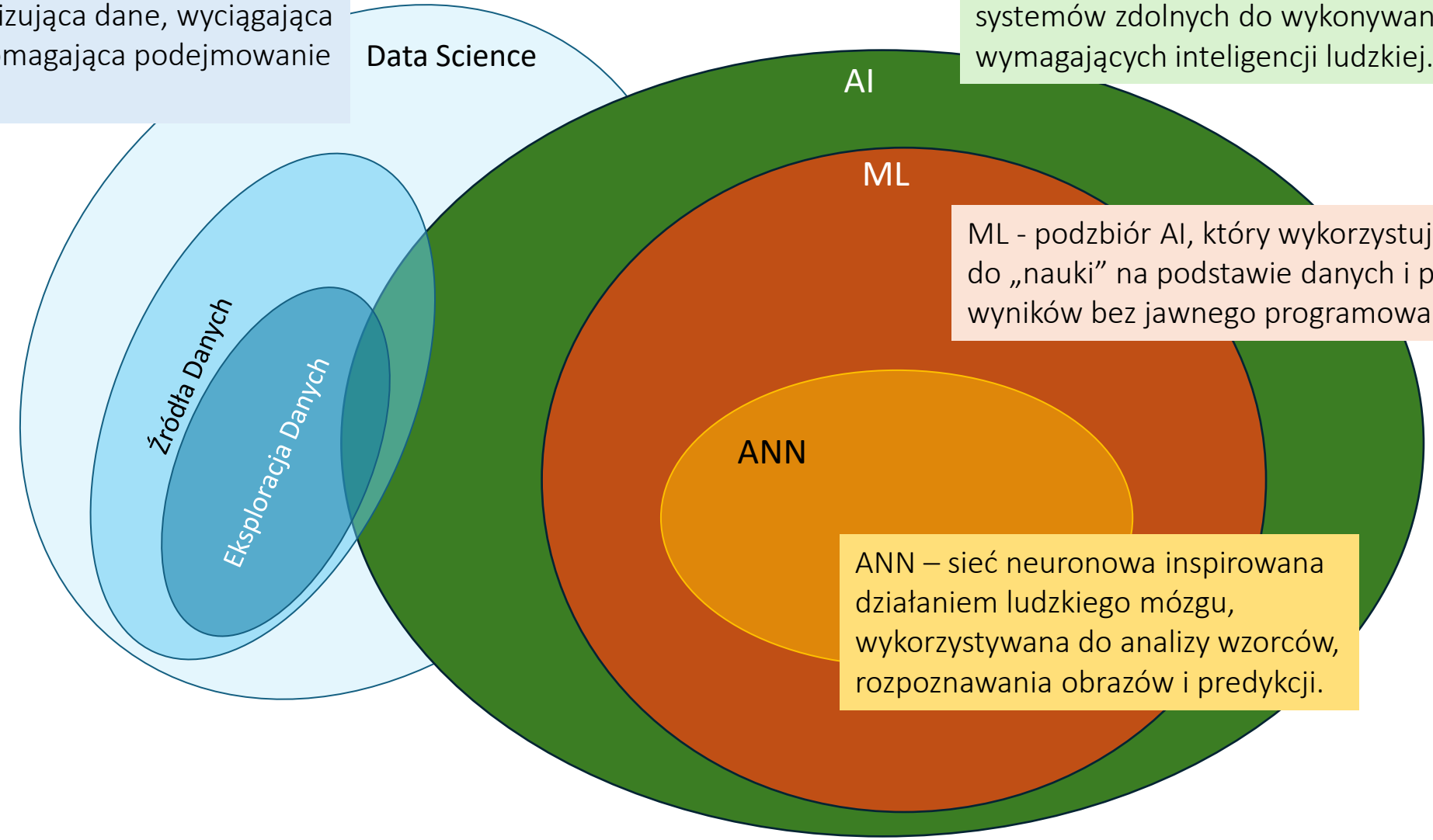
Czyli nie ma czego się jeszcze bać ☺



AI vs. ML vs. ANN vs. Data Science

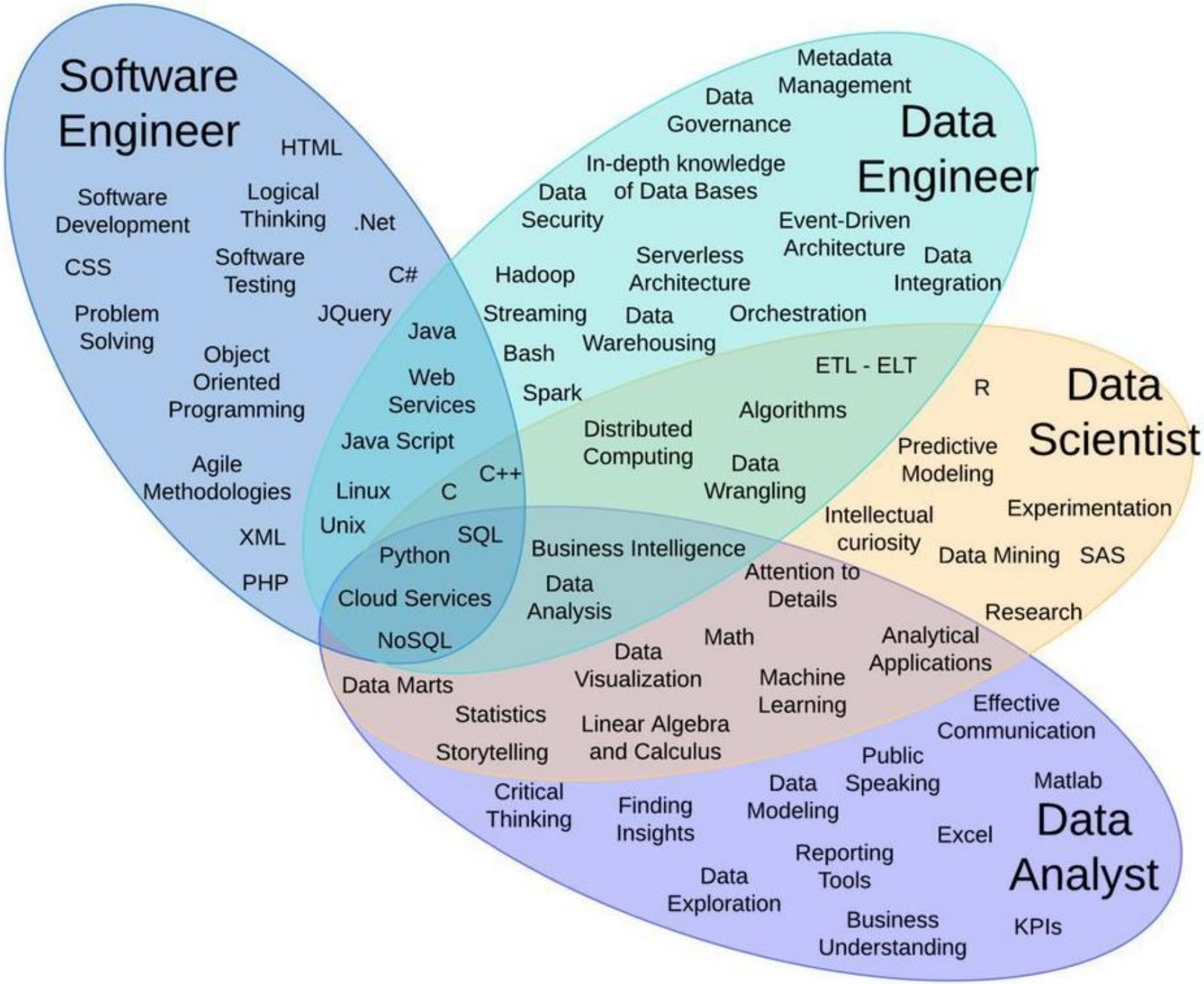
Data Science - interdyscyplinarna dziedzina analizująca dane, wyciągająca wnioski i wspomagająca podejmowanie decyzji.

AI - dziedzina nauki zajmująca się tworzeniem systemów zdolnych do wykonywania zadań wymagających inteligencji ludzkiej.



ML - podzbiór AI, który wykorzystuje algorytmy do „nauki” na podstawie danych i poprawy wyników bez jawnego programowania.

ANN – sieć neuronowa inspirowana działaniem ludzkiego mózgu, wykorzystywana do analizy wzorców, rozpoznawania obrazów i predykcji.


Software Engineer
HTML
Software Development
Logical Thinking
.Net
CSS
Software Testing
C#
Problem Solving
JQuery
Java
Object Oriented Programming
Web Services
Java Script
Agile Methodologies
Linux
C
C++
Unix
XML
SQL
Python
PHP
Cloud Services
NoSQL
Data Marts
Data Engineer
Metadata Management
Data Governance
In-depth knowledge of Data Bases
Data Security
Event-Driven Architecture
Serverless Architecture
Data Integration
Hadoop
Streaming
Data Warehousing
Orchestration
Bash
Spark
ETL - ELT
Distributed Computing
Algorithms
Data Wrangling
Data Scientist
R
Predictive Modeling
Experimentation
Intellectual curiosity
Data Mining
SAS
Business Intelligence
Attention to Details
Data Analysis
Research
Math
Analytical Applications
Data Visualization
Machine Learning
Effective Communication
Statistics
Linear Algebra and Calculus
Storytelling
Public Speaking
Matlab
Critical Thinking
Finding Insights
Data Modeling
Excel
Data Analyst
Reporting Tools
Data Exploration
Business Understanding
KPIs

Proces trenowania sztucznej sieci neuronowej

Trenowanie sztucznej sieci neuronowej

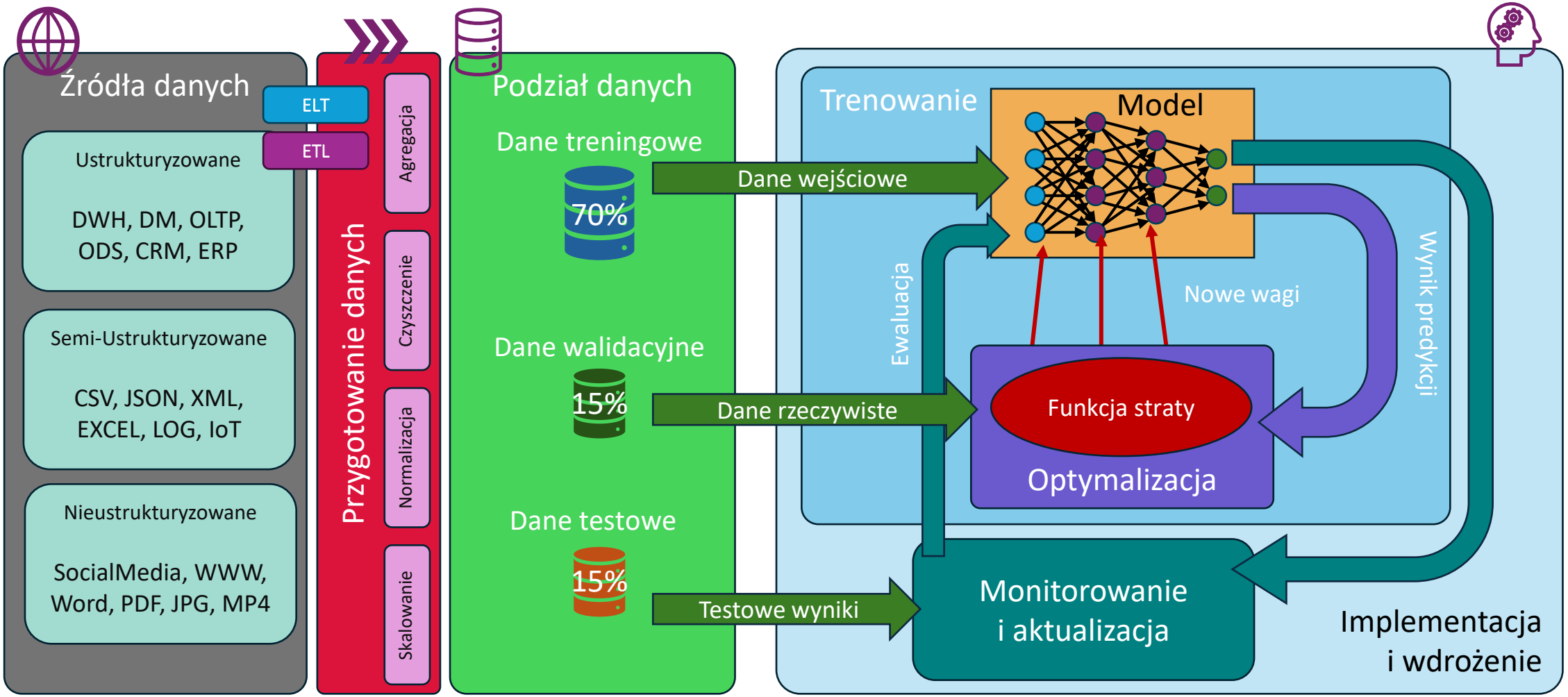
to **proces dostosowywania wag neuronów** w celu minimalizacji błędu prognozy. Dzięki iteracyjnemu uczeniu model staje się coraz dokładniejszy w przewidywaniu wyników.

Kluczowe koncepcje w trenowaniu ANN:

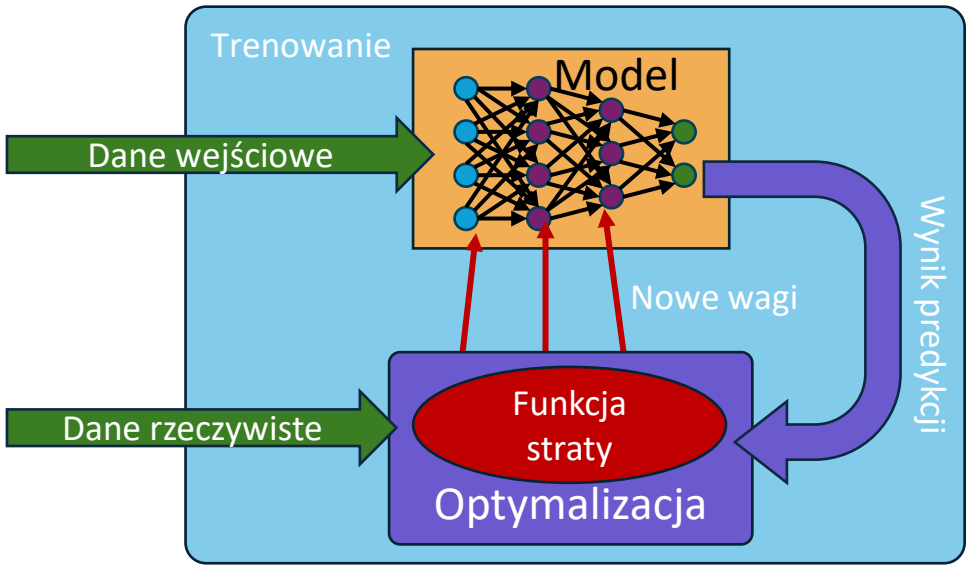
✓ **Overfitting i Underfitting** – model powinien generalizować na nowych danych, a nie tylko zapamiętywać dane treningowe.
✓ **Dropout i Normalizacja** – techniki poprawiające uogólnienie modelu i stabilność treningu.
✓ **Wybór architektury** – liczba warstw i neuronów wpływa na zdolność modelu do wykrywania złożonych wzorców.

Dropout
to **technika regularyzacji** stosowana w sztucznych sieciach neuronowych (ANN) w celu **redukcji przeuczenia (overfittingu)**. Polega na **losowym „wyłączaniu" (zerowaniu) części neuronów** podczas procesu trenowania modelu.

Architektura trenowania sztucznej sieci neuronowej



Proces trenowania sztucznej sieci neuronowej



Inicjalizacja wag

Losowe przypisanie wartości wag połączeń między neuronami.

Przekazanie danych treningowych przez sieć

Dane wejściowe przechodzą przez warstwy neuronów. Generowany jest wynik predykcji.

Obliczenie błędu (Funkcja kosztu)

Porównanie wyniku modelu z rzeczywistą wartością.

Propagacja wsteczna

Obliczenie gradientów błędu (funkcja straty) względem wag.

Aktualizacja wag (Optymalizacja)

Wagi neuronów są modyfikowane przy użyciu optymalizatora.

Powtórzenie procesu (Iteracje i epoki)

Model przechodzi przez dane treningowe wielokrotnie (epoki), aż osiągnie oczekiwaną dokładność.

Epoka i iteracja

Epoka (epoch)

to **jedno pełne przejście (iteracja) przez cały zbiór danych treningowych**, podczas którego model aktualizuje swoje wagi na podstawie wszystkich przykładów.

Każda epoka składa się z wielu **iteracji**, gdzie dane są często podzielone na **minibatch'e** w celu efektywnego przetwarzania.

Jak działa epoka?

1. **Model przetwarza cały zbiór danych** i oblicza błąd predykcji.
2. **Optymalizator aktualizuje wagi** modelu, aby poprawić wyniki.
3. **Proces powtarza się przez wiele epok**, aby stopniowo poprawić zdolność modelu do generalizacji.

Epoka i iteracja

Iteracja (Iteration)

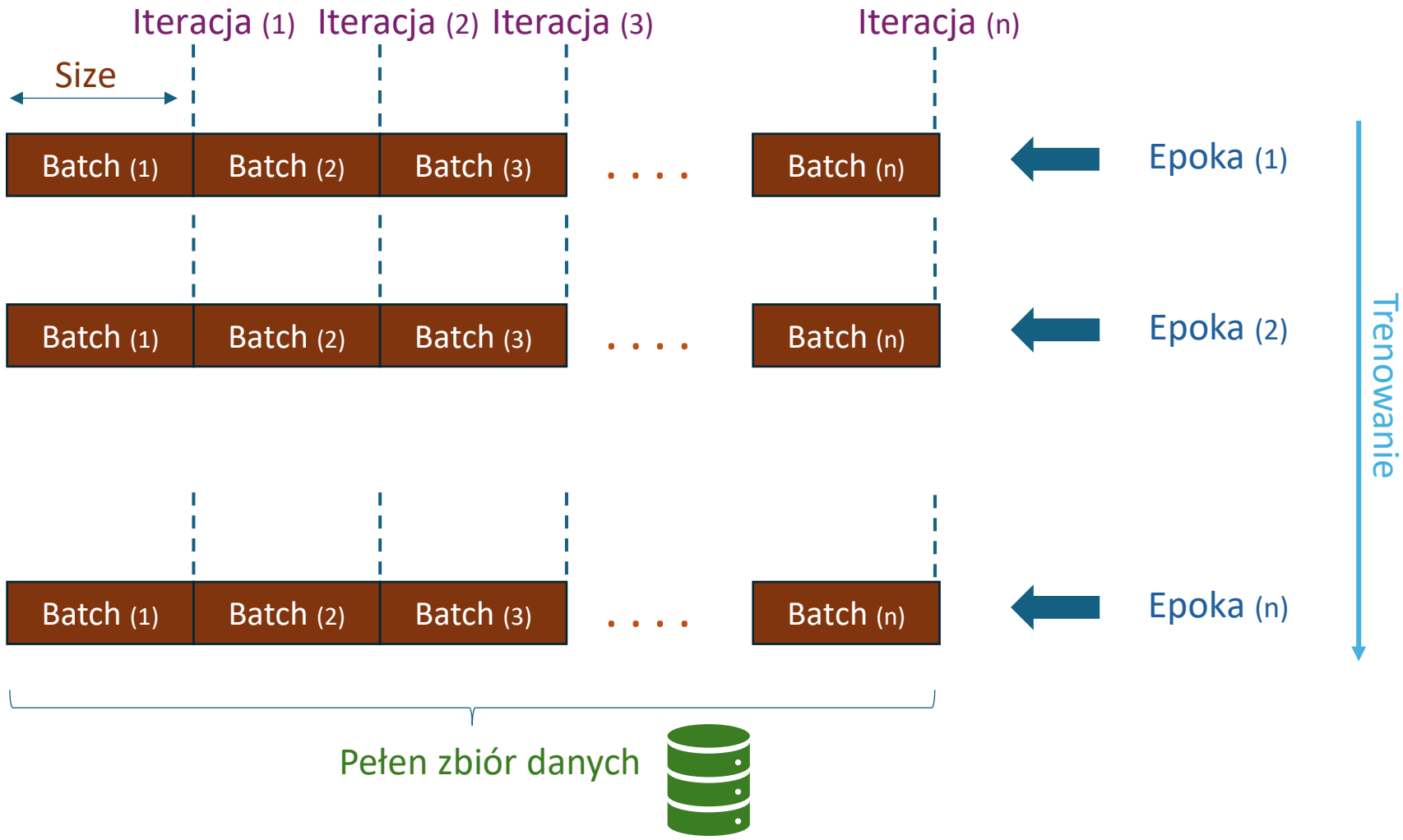
To pojedyncza aktualizacja wag na podstawie jednego batcha danych.

Batch size

to liczba przykładów przetwarzanych jednocześnie w jednej iteracji.

📌 **Przykład**: Jeśli mamy 10 000 próbek i batch size = 100,
to każda epoka składa się z **100 iteracji** (10 000 / 100 = 100).

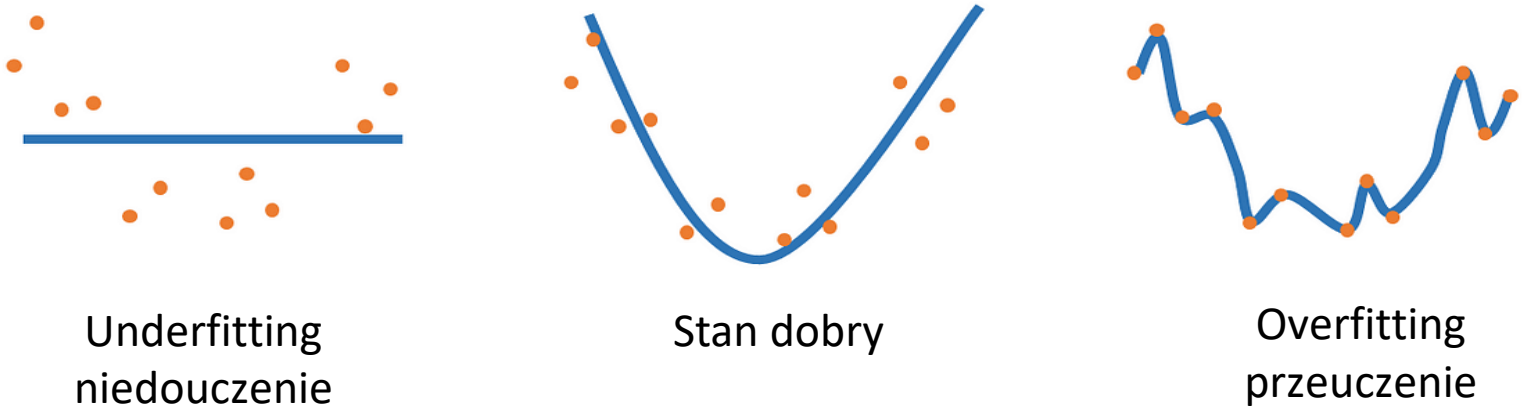
Proces trenowania sztucznej sieci neuronowej (ANN)



Epoka i iteracja

Jak dobrać liczbę epok i Batch size?

- **Za małe** → model nie nauczy się dobrze wzorców (**niedouczenie, underfitting**).
- **Za dużu** → model zacznie zapamiętywać dane zamiast uogólniać (**przeuczenie, overfitting**).
- **Metoda „early stopping”** – zatrzymuje trening, gdy model przestaje się poprawiać.



Underfitting
niedouczenie

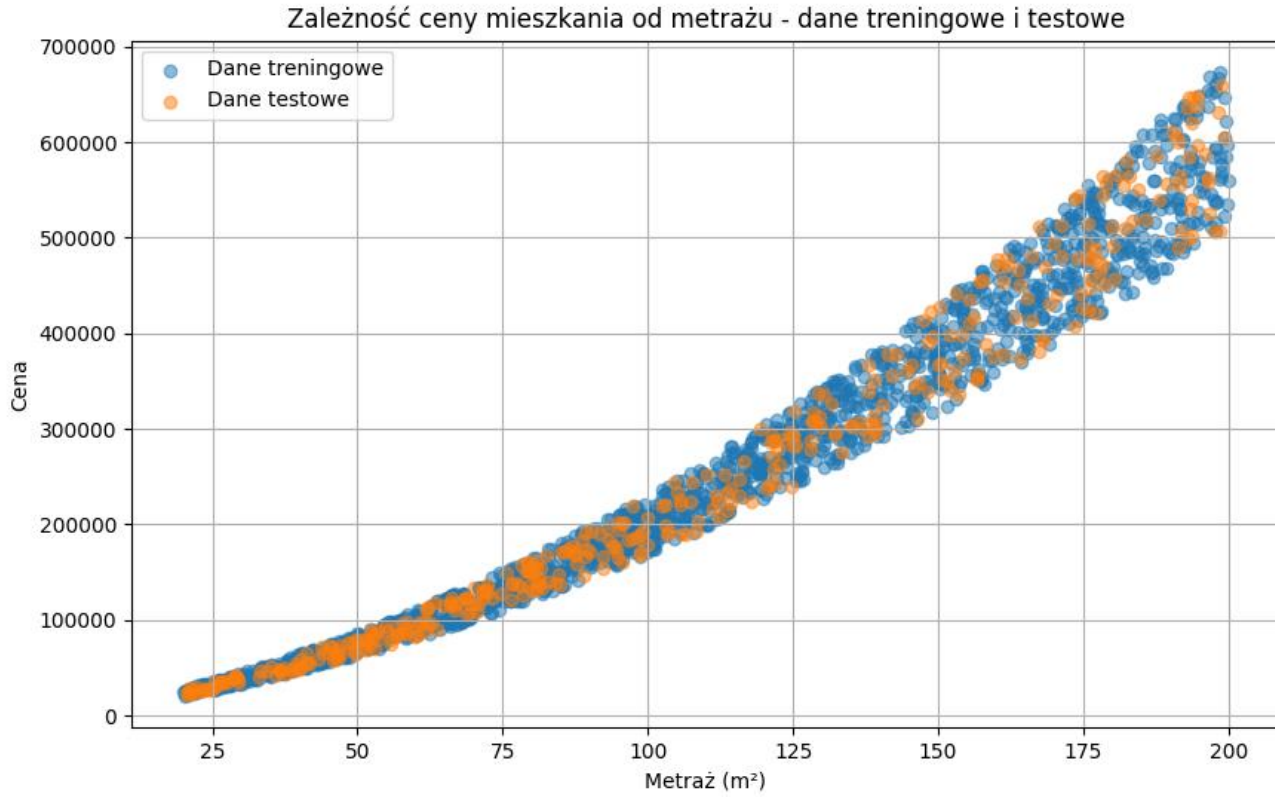
Stan dobry

Overfitting
przeuczenie

Regresja Liniowa vs Sztuczna Sieć Neuronowa

Zależność ceny mieszkania od metrażu

Dane = 2000 próbek
Dane treningowe 80% = 1600 próbek
Dane testowe 20% = 400 próbek



https://colab.research.google.com/drive/1J_-ERqZtdrt79HMFuOAZYWA3vhiD0z2K?usp=sharing

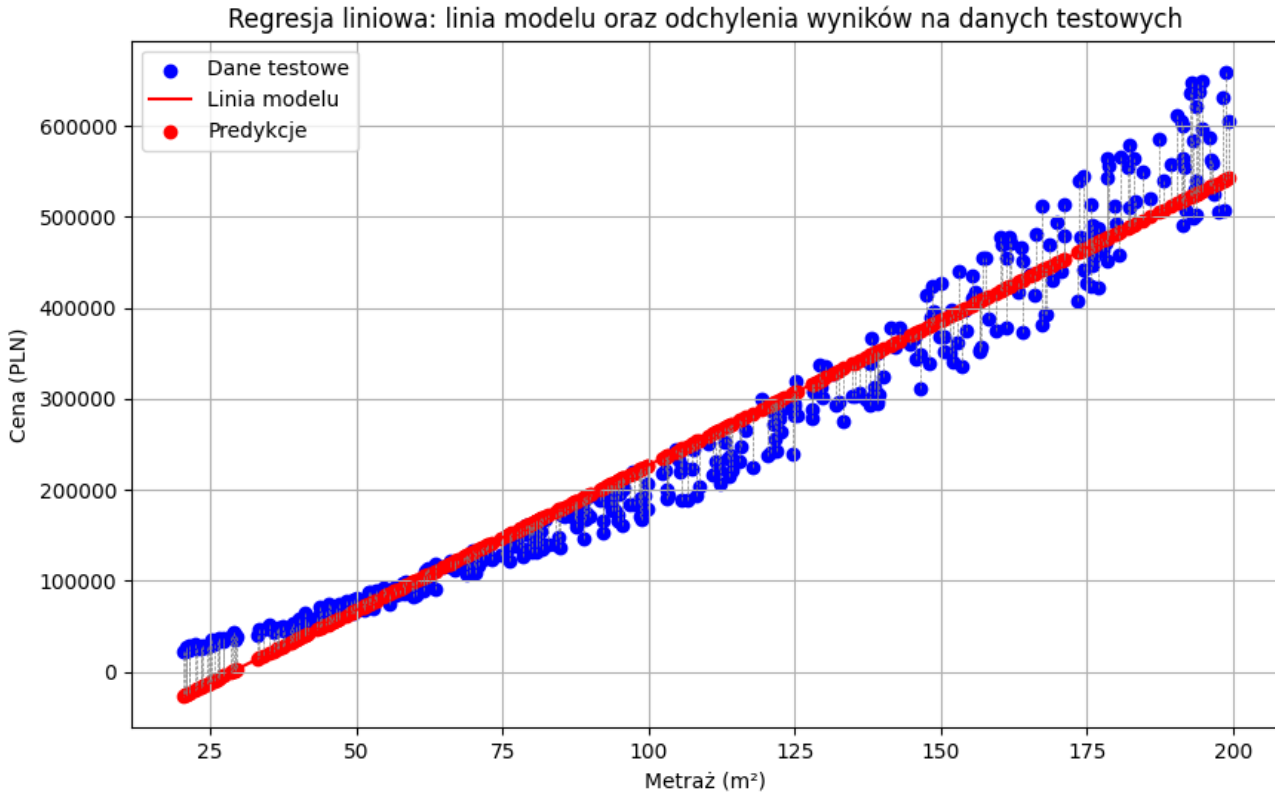
Regresja Liniowa vs Sztuczna Sieć Neuronowa

Regresja Liniowa

Czas treningu: 0.02 s

Wzór modelu regresji liniowej:
f(x) = Cena = 3183.47 * Metraz -91495.18

Ocena modelu na danych testowych
R^2 : 0.9553220157043947



Regresja Liniowa vs Sztuczna Sieć Neuronowa

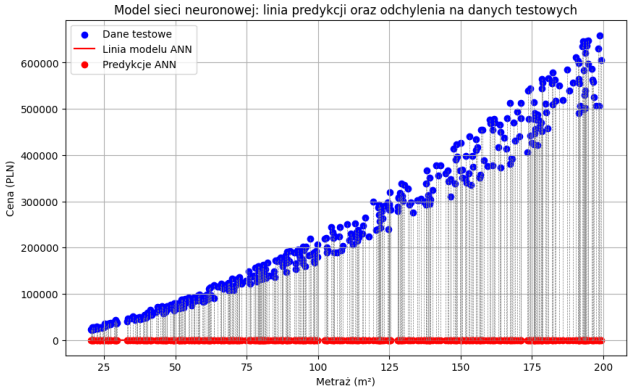
Sztuczna Sieć Neuronowa (ANN)

Liczba parametrów = (64 wejścia × 32 neurony) + 32 biasy = **2 080**.

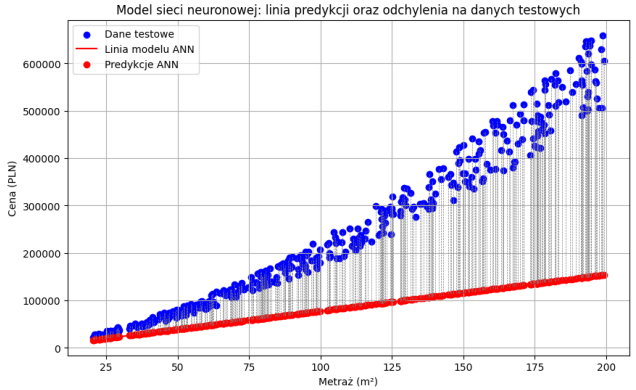
Warstwa	Liczba neuronów	Liczba parametrów
Wejściowa	64	128
Wewnętrzna	32	2080
Wyjsciowa	1	33

Trening

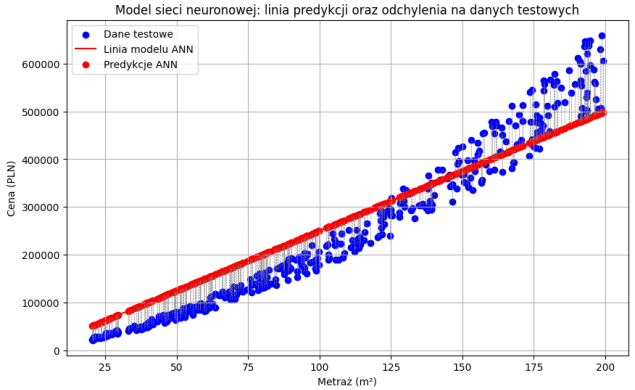
Liczba epok	Czas treningu	R^2
1	1.24 s	-2.07931785940786
20	6.31 s	-0.5242032584355099
100	27.74 s	0.8955676529369911



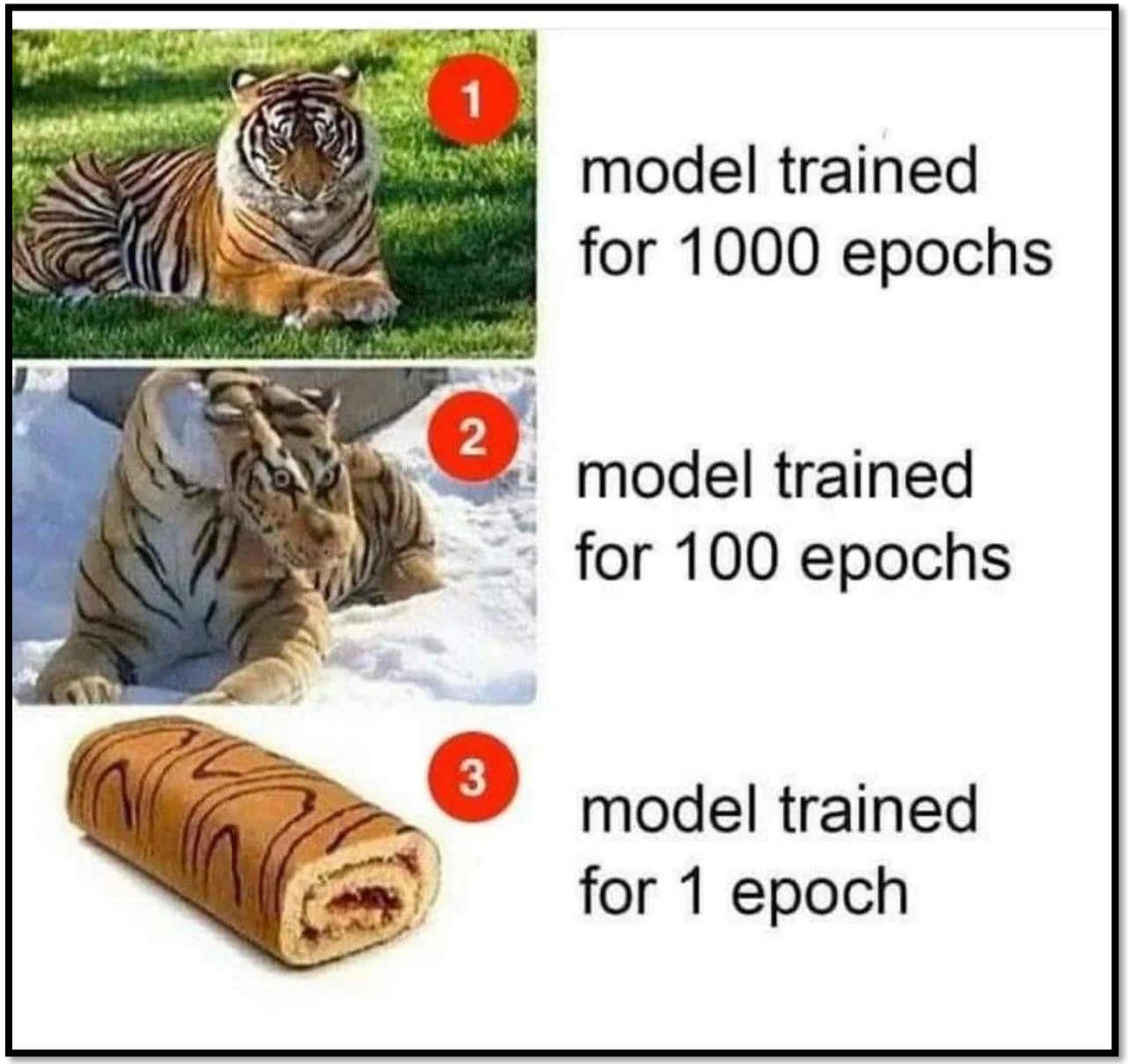
Liczba epok: 1



Liczba epok: 20



Liczba epok: 100


1
model trained
for 1000 epochs
2
model trained
for 100 epochs
3
model trained
for 1 epoch

Optymalizacja

Optymalizacja

w sieciach neuronowych to **proces dostosowywania wag (parametrów modelu)**, aby **zminimalizować błąd** predykcji i poprawić ogólną wydajność modelu.

Jest to kluczowy krok w uczeniu głębokim, który wpływa na **szybkość, stabilność i dokładność** modelu.

Kluczowe elementy optymalizacji

1 Funkcja kosztu (Loss Function) 🎯

Mierzy, jak bardzo przewidywania modelu różnią się od rzeczywistych wartości.

Przykłady:

- *MSE (Mean Squared Error)* – regresja.
- *Cross-Entropy Loss* – klasyfikacja.

Optymalizacja

Kluczowe elementy optymalizacji

2 Algorytmy optymalizacji (Optimizers) ⚡

Algorytmy aktualizujące wagi neuronów na podstawie gradientów.

Najpopularniejsze metody:

✓ **SGD (Stochastic Gradient Descent)** – podstawowy, ale może być wolny.

✓ **Adam (Adaptive Moment Estimation)** – szybki, adaptacyjny, szeroko stosowany.

✓ **RMSprop** – często używany w sieciach rekurencyjnych (RNN).

3 Szybkość uczenia (Learning Rate, η) 🔄

Decyduje, jak duże kroki wykonuje model podczas aktualizacji wag.

- **Za wysoka wartość** → model może nie zbiegać do minimum.
- **Za niska wartość** → uczenie będzie bardzo wolne.
- **Rozwiązanie:** *Decay* (zmniejszanie learning rate w czasie).

Optymalizacja

Kluczowe elementy optymalizacji

4 Batch size i epoki 📊

- **Batch size** → liczba przykładów przetwarzanych jednocześnie przed aktualizacją wag.
- **Większy batch size** → stabilniejsze uczenie, ale większe wymagania sprzętowe.
- **Liczba epok** → liczba pełnych przejść przez cały zbiór danych.

5 Regularizacja 🔍

Zapobiega *przeuczeniu (overfitting)* poprzez ograniczenie zdolności modelu do zapamiętywania danych.

Metody:

✓ **Dropout** – losowe „wyłączanie" neuronów w warstwach w trakcie treningu.

✓ **L1/L2 (Regularyzacja Lasso/Ridge)** – ogranicza wielkość wag neuronów.

Optymalizacja

Kluczowe elementy optymalizacji

6 Przygotowanie danych (Data Preprocessing) 📏
Przyspiesza uczenie i stabilizuje model.
Metody:
✓ Standardizacja (z-score, μ=0, σ=1) – używana w regresji.
✓ Min-max scaling (0-1) – popularne w sieciach neuronowych.

7 Early Stopping 🛑
- Przerywa trenowanie, gdy model przestaje poprawiać swoje wyniki na zbiorze walidacyjnym.
- Zapobiega przeuczeniu i oszczędza czas obliczeń.

Macierz pomyłek

Macierz pomyłek (*Confusion Matrix*)

to **narzędzie oceny jakości klasyfikatora**, stosowane w **uczeniu maszynowym i AI**.

Pozwala zobaczyć, **jak dobrze model przewiduje klasy**, pokazując liczbę poprawnych i błędnych klasyfikacji dla każdej kategorii.

Zastosowanie w AI

✅ **Wykrywanie oszustw bankowych** – **minimalizacja FP (fałszywych alarmów)** jest kluczowa.
✅ **Diagnostyka medyczna** – **FN (błędna diagnoza negatywna)** jest niebezpieczna, bo oznacza przeoczenie choroby.
✅ **Filtry antyspamowe** – **FP (błędnie oznaczone e-maile jako spam)** mogą powodować utratę ważnych wiadomości.

Macierz pomyłek

Kluczowe metryki

Przewidziana klasa → Prawdziwa klasa ↓	Pozytywna (1)	Negatywna (0)
Pozytywna (1) 🎯	🔷 TP (True Positive) przewidziane pozytywne przypadki	❌ FN (False Negative) model błędnie uznał, że przypadek jest negatywny
Negatywna (0) 🎯	❌ FP (False Positive) model błędnie uznał, że przypadek jest pozytywny	🔷 TN (True Negative) Poprawnie przewidziane negatywne przypadki

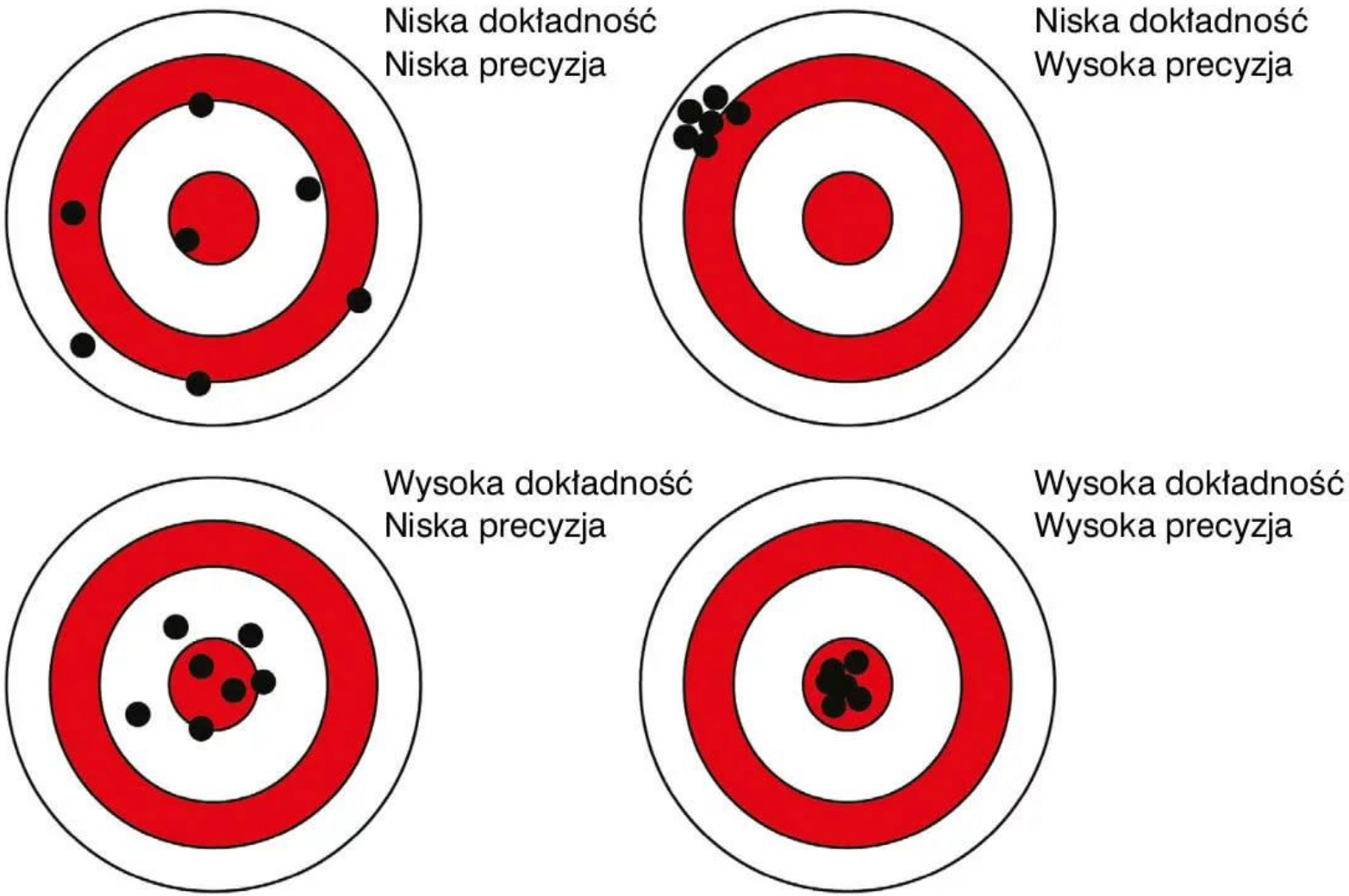
Accuracy (dokładność) – miara pokazująca, jak dobrze model klasyfikuje wszystkie przykłady.
Accuracy= (TP + TN) / (TP + TN + FP + FN)

Precision (precyzja) – miara określająca, jak wiele z przewidywanych pozytywnych przypadków faktycznie jest pozytywnych.
Precision= TP / (TP + FP)

Recall (czułość) – miara pokazująca, jak dobrze model wychwytuje rzeczywiste pozytywne przypadki.
Recall= TP / (TP + FN)

F1-Score – harmoniczna średnia precyzji i czułości, łączącą obie powyższe miary w jedną. Jest szczególnie przydatna, gdy mamy do czynienia z niezbalansowanymi danymi.
*F1-Score= 2 * (Precision * Recall) / (Precision + Recall)*

Dokładność i precyzja



Zadanie



Treść

W Google Colab https://colab.research.google.com/ przetestuj macierz pomyłek.

1. Wygeneruj 100 przykładowych danych w postaci klasyfikacji e-maili jako spam lub nie spam losowo jako poprawna lub niepoprawna klasyfikacja. Pokaz wygenerowane dane w tabeli.
2. Na podstawie tych danych zbuduj macierz pomyłek
3. Oszacuje teraz
   - Accuracy (dokładność)
   - Precision (precyzja)
   - Recall (czułość)
   - F1-Score
4. Dokonaj interpretacji wyników po polsku

https://colab.research.google.com/drive/1a3xXR1PTFb74HBfqNa4JBo_YN5UhUpn-?usp=sharing

Fine-tuning AI

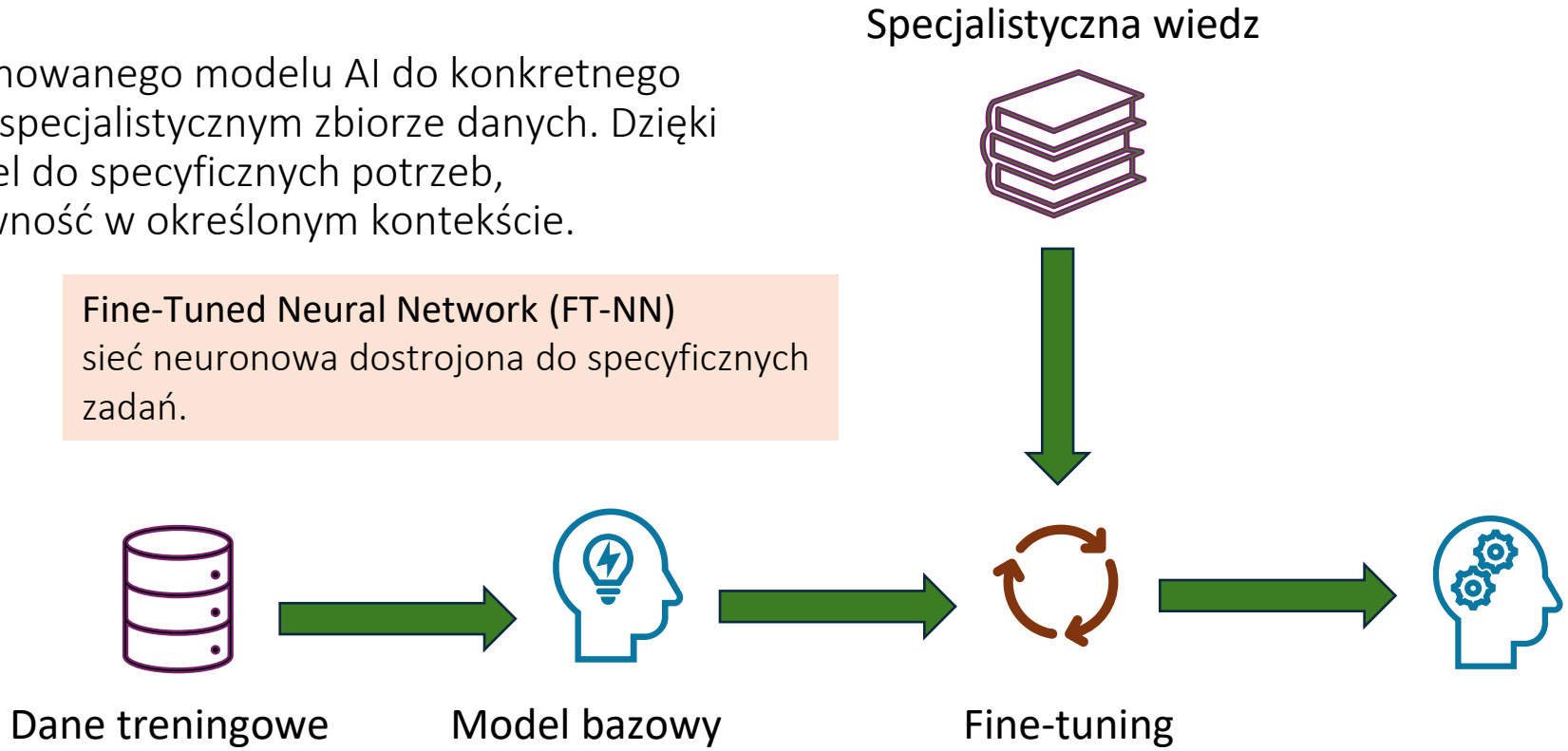
Fine-tuning

to proces dostrajania wcześniej wytrenowanego modelu AI do konkretnego zadania poprzez ponowne uczenie na specjalistycznym zbiorze danych. Dzięki temu można dostosować ogólny model do specyficznych potrzeb, poprawiając jego dokładność i efektywność w określonym kontekście.

Fine-Tuned Neural Network (FT-NN)
sieć neuronowa dostrojona do specyficznych zadań.

◆ Co daje fine-tuning

- Lepszą personalizację
- Poprawę jakości wyników
- Zmniejszenie błędów
- Efektywność obliczeniową



Retrieval-Augmented Generation (RAG)

Retrieval-Augmented Generation (RAG)

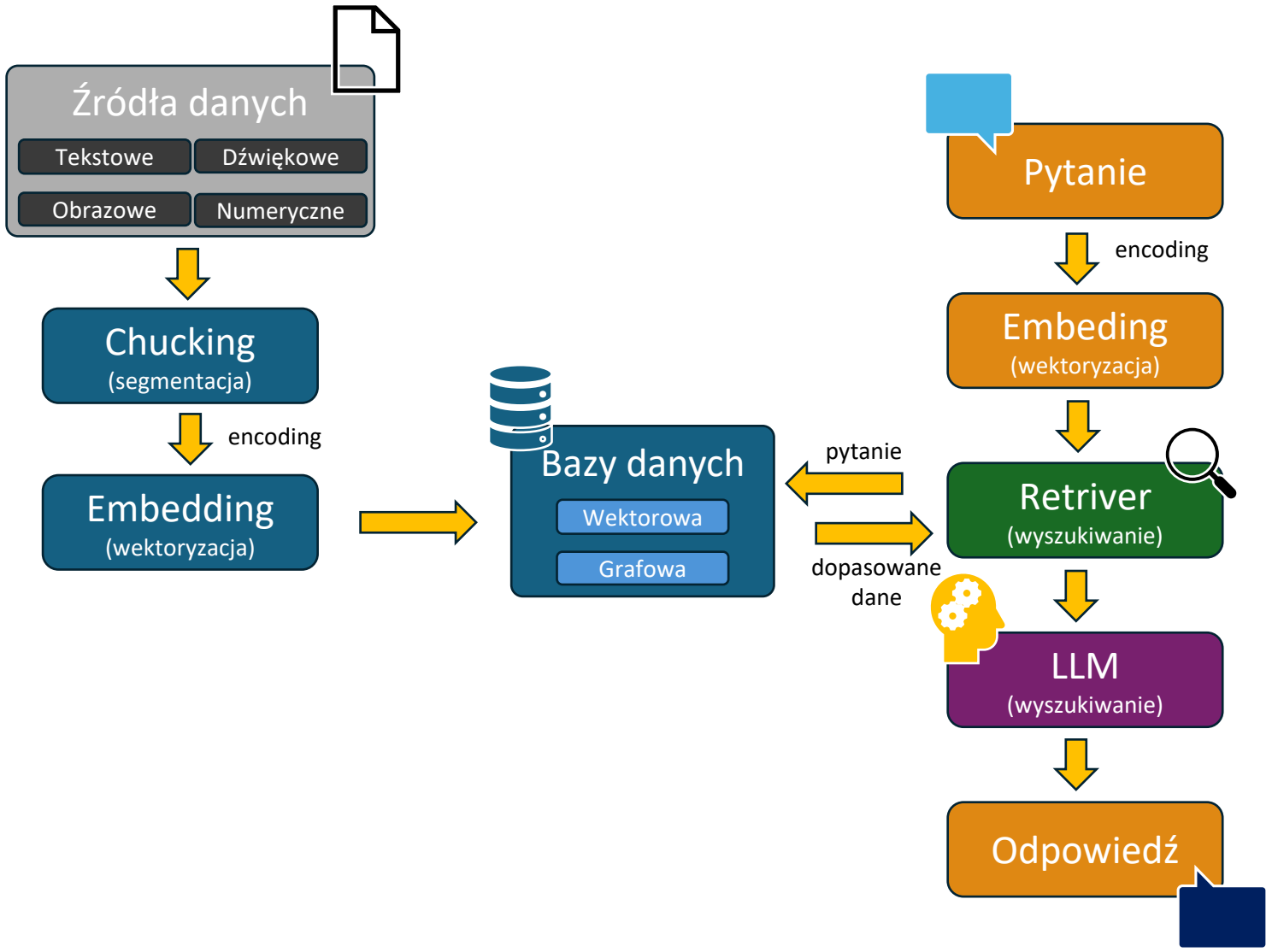
to technika łącząca **wyszukiwanie informacji** i **generowanie odpowiedzi** w modelach AI.

Pozwala na **dokładniejsze i bardziej kontekstowe odpowiedzi**, ponieważ model może pobierać aktualne informacje z baz danych przed wygenerowaniem odpowiedzi.

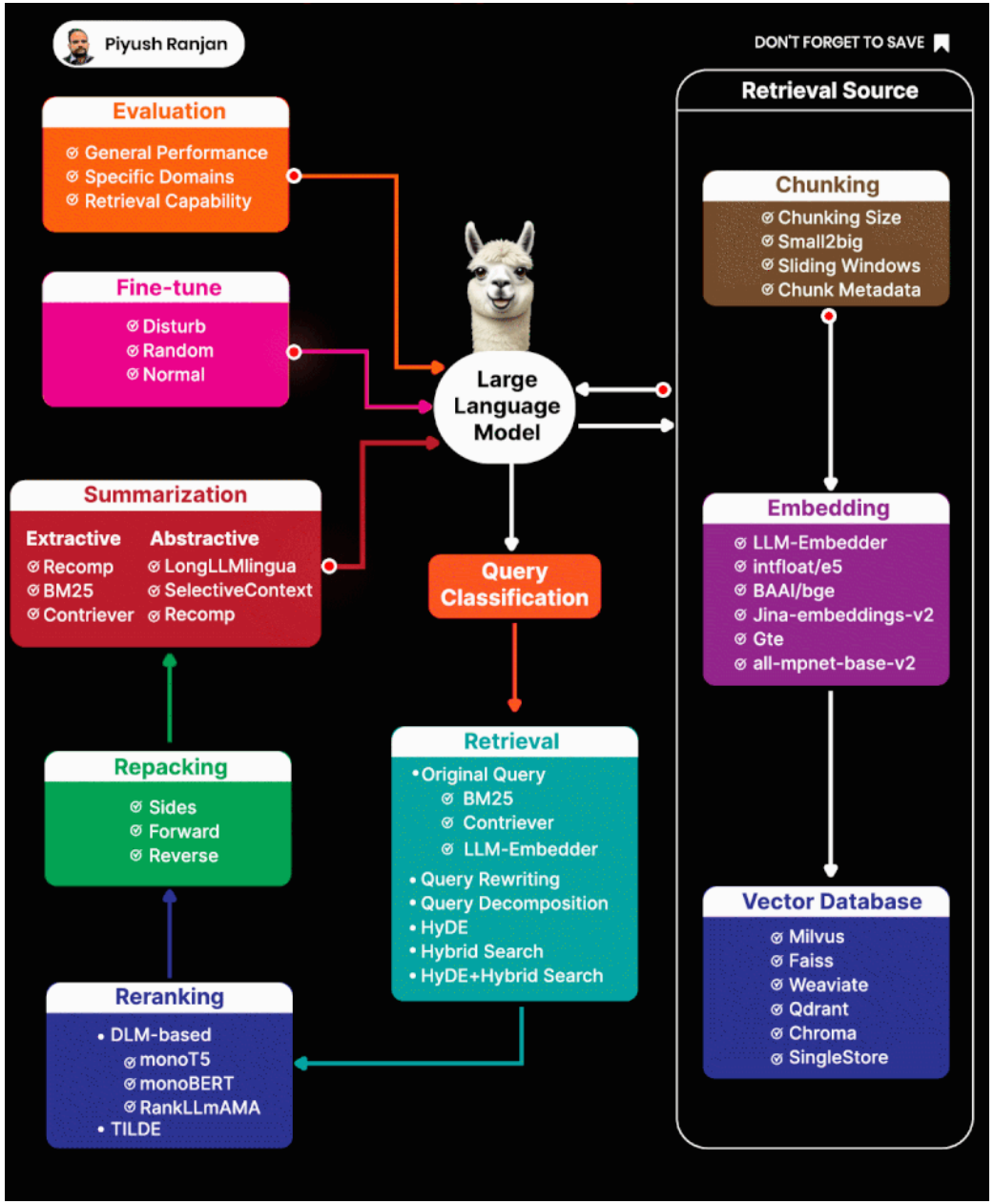
🔷 Dlaczego RAG jest lepszy od standardowych modeli AI?

✓ **Większa precyzja** – model AI korzysta z aktualnych informacji, zamiast polegać tylko na wcześniejszym treningu.
✓ **Lepsza interpretacja kontekstu** – wyszukiwanie w bazach wiedzy pozwala unikać błędnych odpowiedzi (tzw. halucynacji AI).
✓ **Zastosowanie w wielu branżach** – m.in. chatboty, analiza dokumentów, systemy rekomendacji.

Retrieval-Augmented Generation (RAG) w AI



RAG najlepsze praktyki



Wytrenowane modele AI

Repozytoria gotowych modeli AI

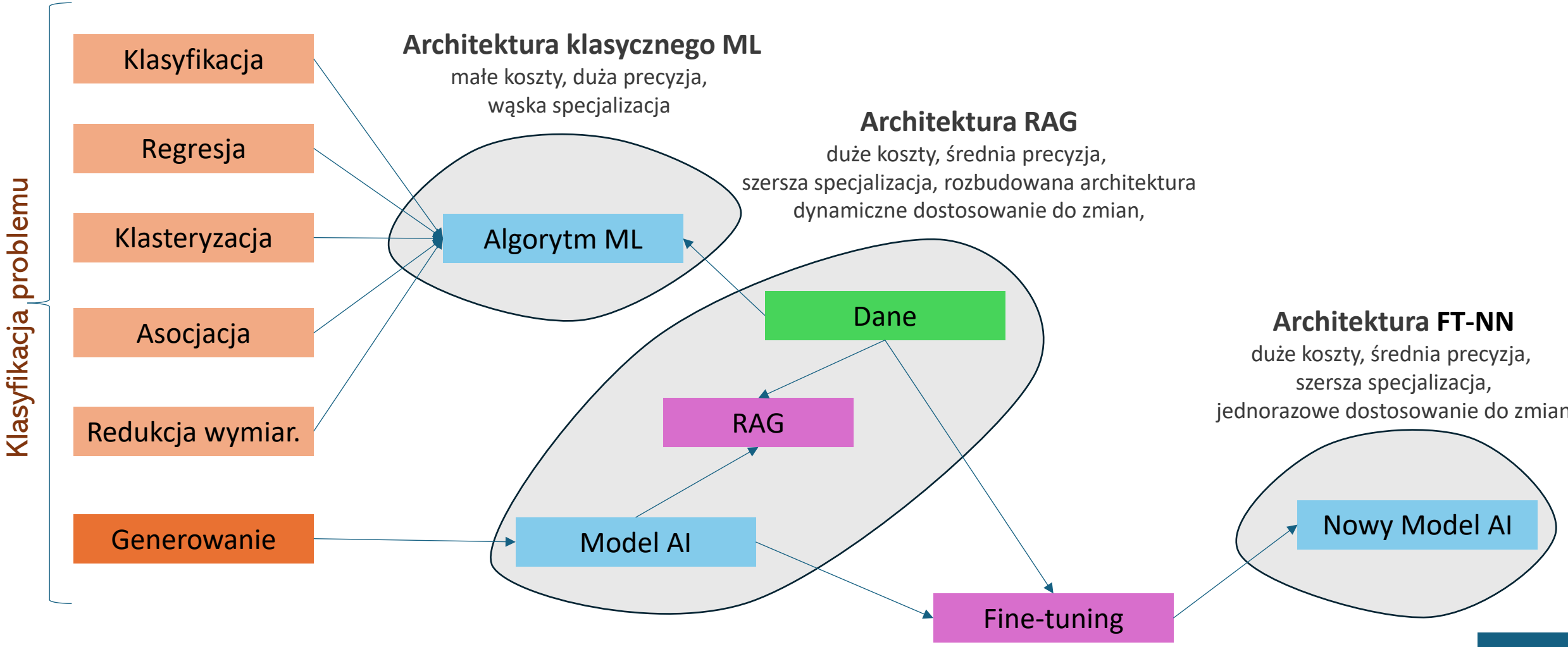
Biblioteka / Platforma	Opis	Link
Hugging Face Transformers	Największa biblioteka modeli NLP, CV i Audio. Obsługuje modele takie jak GPT, BERT, T5, CLIP, Stable Diffusion	https://huggingface.co/models
TensorFlow Hub	Gotowe modele AI do klasyfikacji obrazów, NLP, przetwarzania mowy i rekomendacji	https://tfhub.dev/
Torch Hub	Modele deep learning dla PyTorch – NLP, CV, generatywne AI	https://pytorch.org/hub/
Keras Applications	Wbudowane modele CV w Kerasie (np. ResNet, MobileNet, EfficientNet)	https://keras.io/api/applications/
NVIDIA TAO Toolkit	Modele AI zoptymalizowane pod GPU (Computer Vision, NLP, Audio)	https://developer.nvidia.com/tao-toolkit
Llama (Meta AI)	Modele open-source od Meta, alternatywa dla GPT	https://ai.meta.com/resources/models-and-libraries/

Wytrenowane modele AI

TOP Modele LLM dla języka polskiego

Model	Dostępność	Zalety	Najlepsze zastosowania	Link do pobrania
GPT-4 (OpenAI)	Chmurowy (API)	Najlepsza jakość generacji Dobre rozumienie polskiego Kontekst do 128k tokenów	Asystenci AI, chatboty, analiza tekstu	https://platform.openai.com/
Mistral 7B / Mixtral	Open-source	Dobra jakość Możliwość uruchomienia lokalnie Brak ograniczeń chmury	Prywatne systemy AI, AI-on-premise	https://huggingface.co/BahaaEldin0/Mistral-7b-Mixtral-7b
Llama 2 (Meta, 7B-70B)	Open-source	Solidna jakość Możliwość fine-tuningu	Przetwarzanie tekstu, AI w firmach	https://huggingface.co/meta-llama/Llama-2-7b-hf
Falcon 40B	Open-source	Dobre rozumienie języków europejskich Wolny od licencji	NLP, generowanie treści	https://huggingface.co/tiiuae/falcon-40b
BLOOM (176B, 7B)	Open-source	Jeden z największych modeli open-source Obsługuje wiele języków	Wielojęzyczne AI, badania akademickie	https://huggingface.co/bigscience/bloom
Bielik (11B)	Open-source	Największy polski model językowy Dostosowany do języka polskiego Możliwość lokalnego wdrożenia	Przetwarzanie języka polskiego, analiza dokumentów, NLP dla firm	https://huggingface.co/speakleash/Bielik-7B-v0.1

Jaką architekturę systemu AI wybrać?



MLOps

Zarządzanie cyklem życia modeli uczenia maszynowego - MLOps (Machine Learning Operations)

to **zbiór praktyk i narzędzi**, które łączą **uczenie maszynowe (ML) i operacje IT (Operations)** w celu **zarządzania i wdrażania modeli ML** w sposób skalowalny, niezawodny i zautomatyzowany.

MLOps obejmuje cały **cykl życia modelu uczenia maszynowego**, od **tworzenia i trenowania**, przez **wdrażanie, monitorowanie, utrzymanie**, aż po **aktualizacje i zarządzanie danymi**.

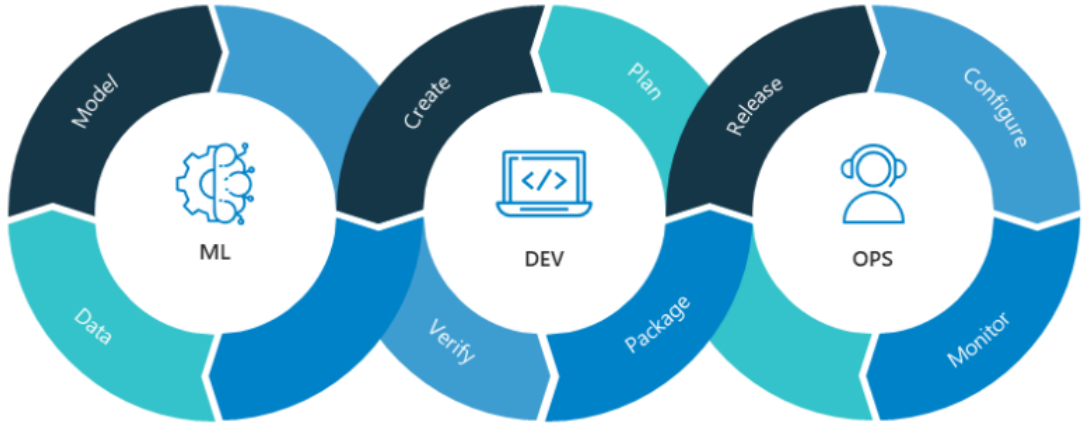
🔷 **Korzyści wynikające z MLOps**

✅ **Automatyzacja procesów** – redukcja błędów i przyspieszenie wdrożeń modeli.

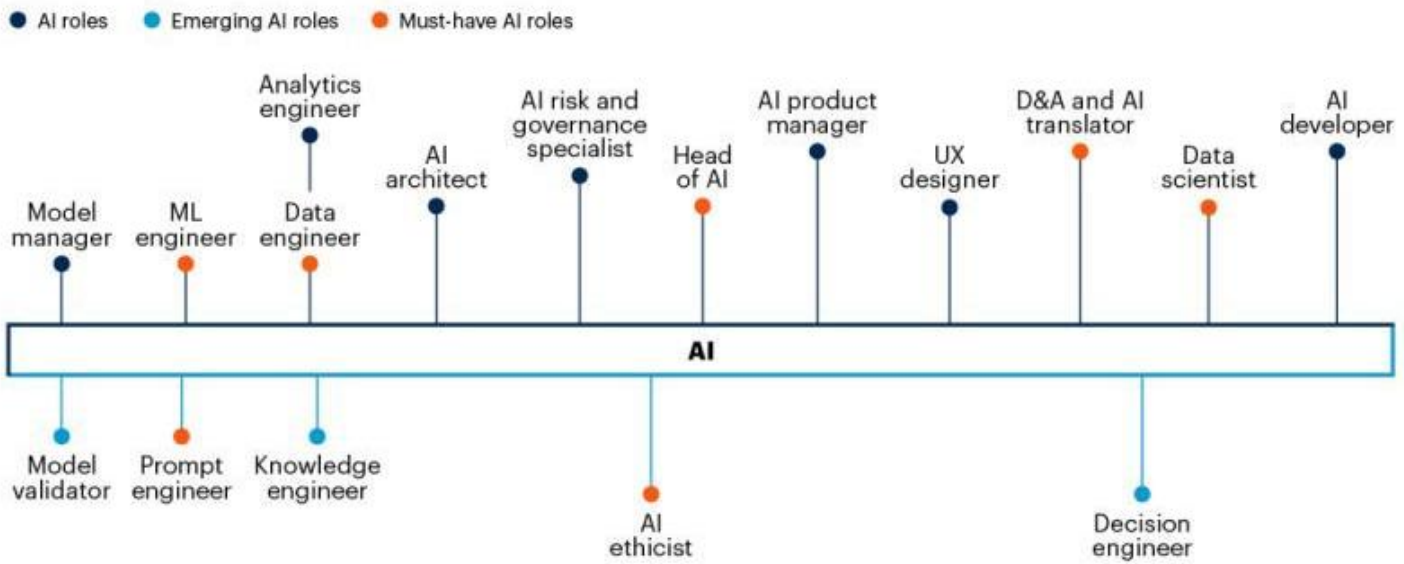
✅ **Reprodukowalność** – możliwość odtwarzania wyników modeli w różnych środowiskach.

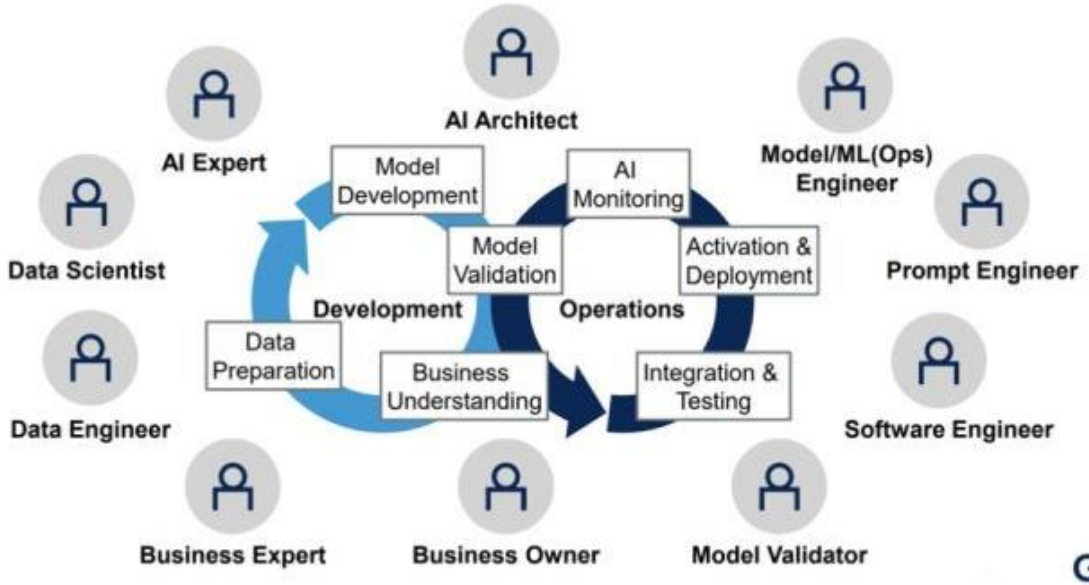
✅ **Monitorowanie i zarządzanie modelami** – zapewnienie stabilności i efektywności w dłuższym okresie.

✅ **Skalowalność** – łatwe wdrażanie modeli na dużą skalę w chmurze lub on-premise.



Istotne role w procesie implementacji systemu AI





MLOps

Platformy zarzadzania modelami AI

Platforma	Opis	Instalacja On-Premise	Open Source	Link
MLflow	Open-source'owa platforma do zarządzania cyklem życia modeli ML.	TAK	TAK	https://mlflow.org/
Kubeflow	Framework do zarządzania pipeline’ami ML na Kubernetesie.	TAK	TAK	https://www.kubeflow.org/
IBM Watson AI	Narzędzie IBM dla chmury i on-premise, wspierające AI Governance.	TAK	NIE	https://www.ibm.com/watsonx
Weights & Biases (W&B)	Narzędzie do monitorowania eksperymentów ML i wersjonowania modeli.	CZĘŚCIOWO	NIE	https://wandb.ai/
Google Vertex AI	Kompleksowa platforma AI w Google Cloud, integrująca MLOps, AutoML i Model Registry.	NIE	NIE	https://cloud.google.com/vertex-ai
Amazon SageMaker	Usługa AWS do trenowania, monitorowania i wdrażania modeli ML.	NIE	NIE	https://aws.amazon.com/sagemaker/
Azure ML	Platforma Microsoftu do budowy, wdrażania i monitorowania modeli ML.	NIE	NIE	https://azure.microsoft.com/en-us/products/machine-learning

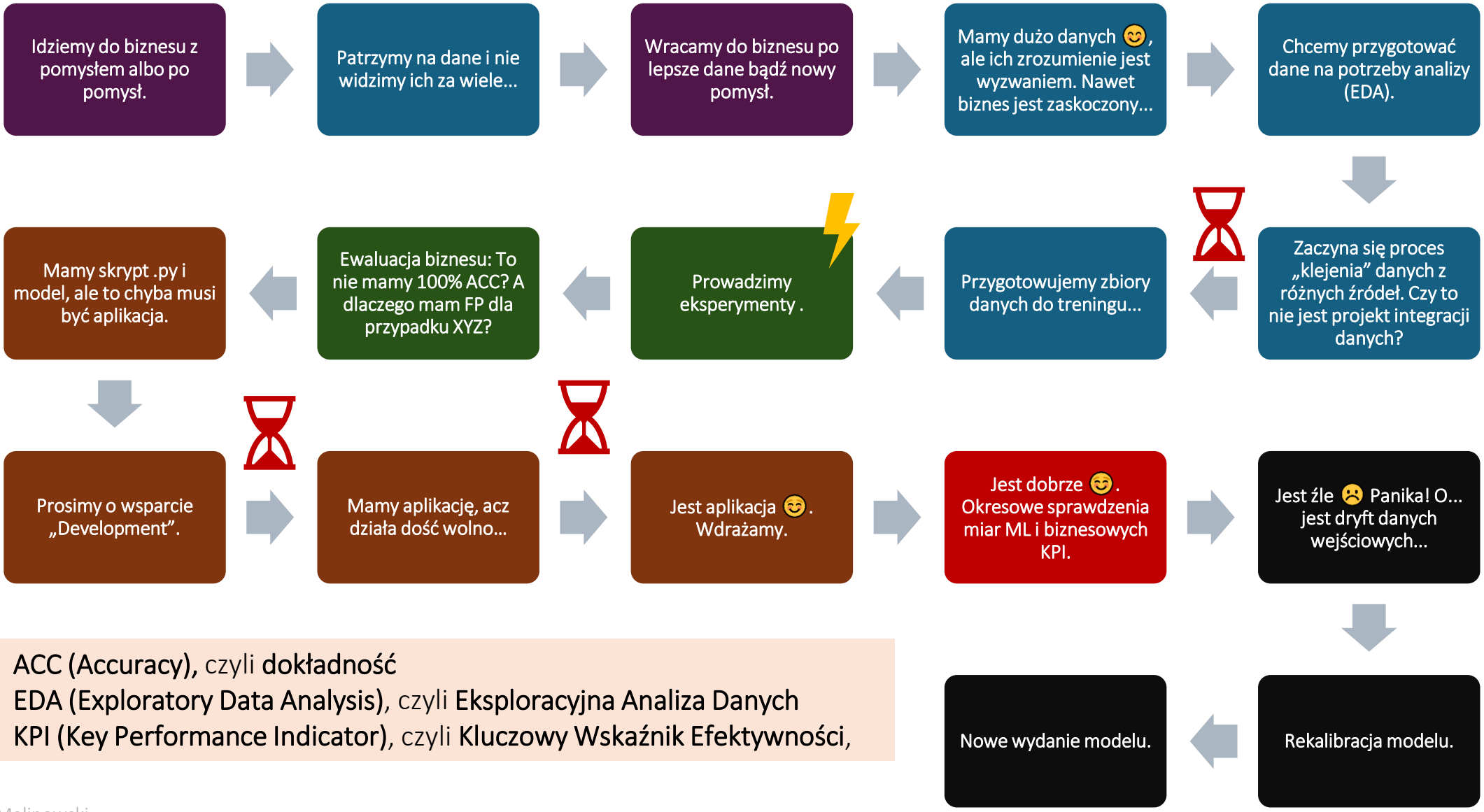
MLOps

Metody dostępu do systemów AI w aplikacjach

Metoda	Zalety	Najlepsze dla	Przykłady
API REST/SOAP (Chmurowe AI)	Skalowalność, brak konieczności zarządzania infrastrukturą	Aplikacje webowe, SaaS	OpenAI GPT, Google Vertex AI, Azure Cognitive Services
SDK i Biblioteki AI	Wydajność, pełna kontrola nad modelem	Aplikacje lokalne, R&D	TensorFlow, PyTorch, OpenAI Gym
Mikroserwis AI (Lokalny serwer AI)	Elastyczność, prywatność, wdrażanie on-premise	Firmy potrzebujące AI bez chmury	Flask, FastAPI, TensorFlow Serving, Docker
Wbudowanie AI (Edge, IoT, Mobile AI)	Real-time, brak potrzeby internetu	AI w aplikacjach mobilnych, IoT, embedded	TensorFlow Lite, ONNX Runtime, Apple Core ML
Serverless AI (Funkcje chmurowe)	Automatyczne skalowanie, płacisz za użycie	Chmurowe aplikacje AI	AWS Lambda + SageMaker, Google Cloud Functions

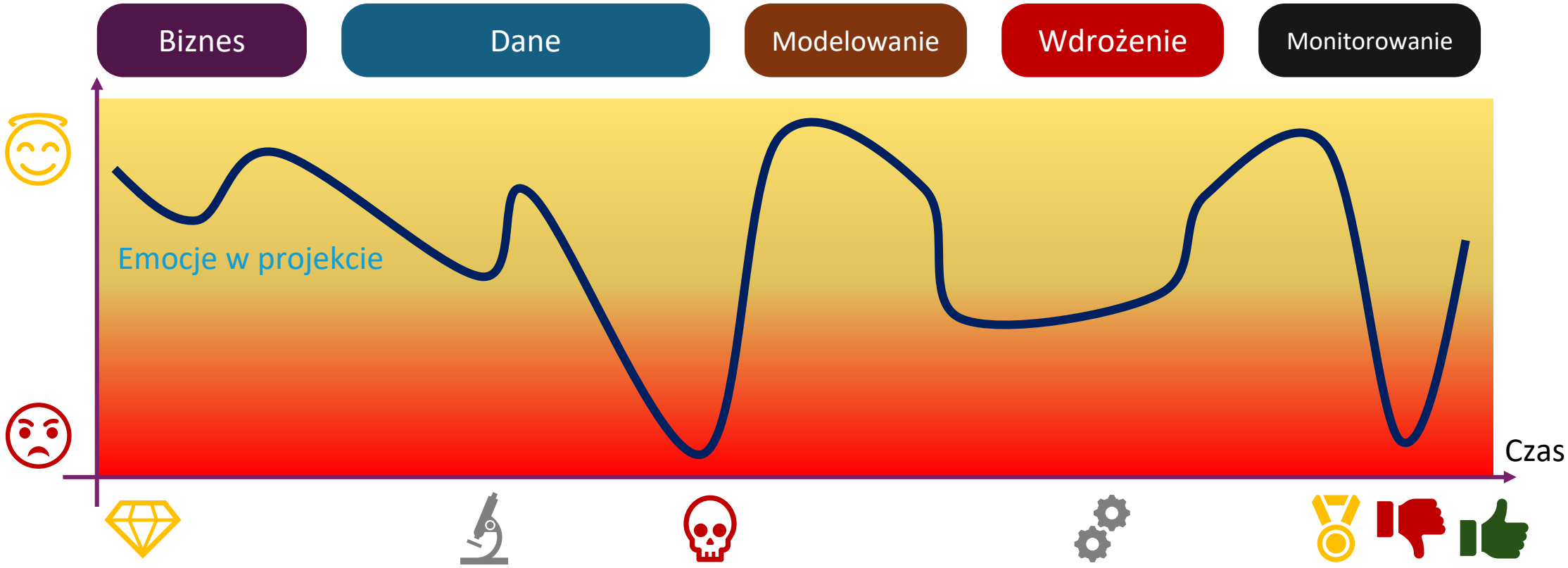
📌 Jeśli chcesz AI w chmurze → API REST lub Serverless AI
📌 Jeśli chcesz AI lokalnie → SDK, Mikroserwis AI, Wbudowane AI

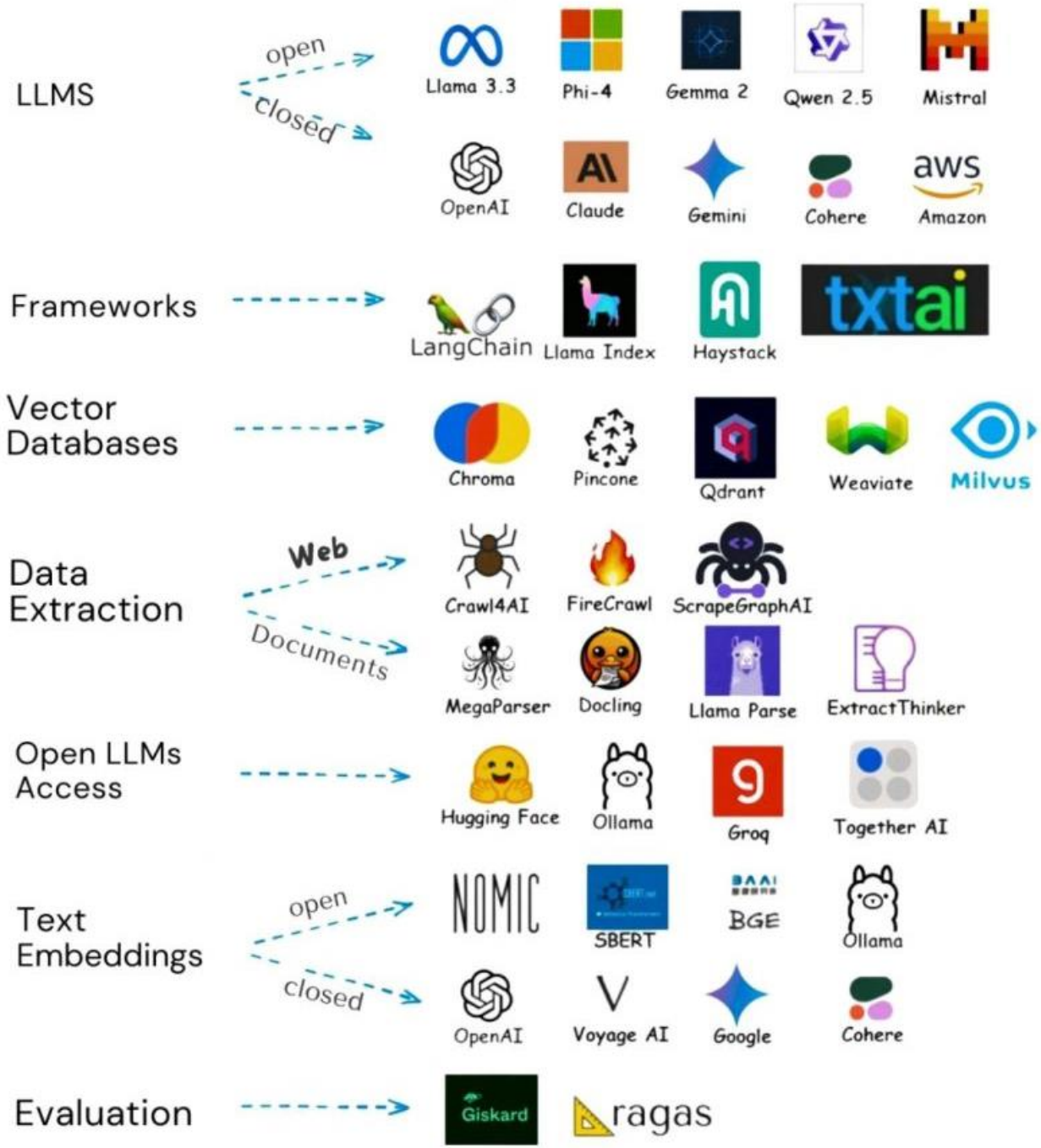
Nasz projekt AI

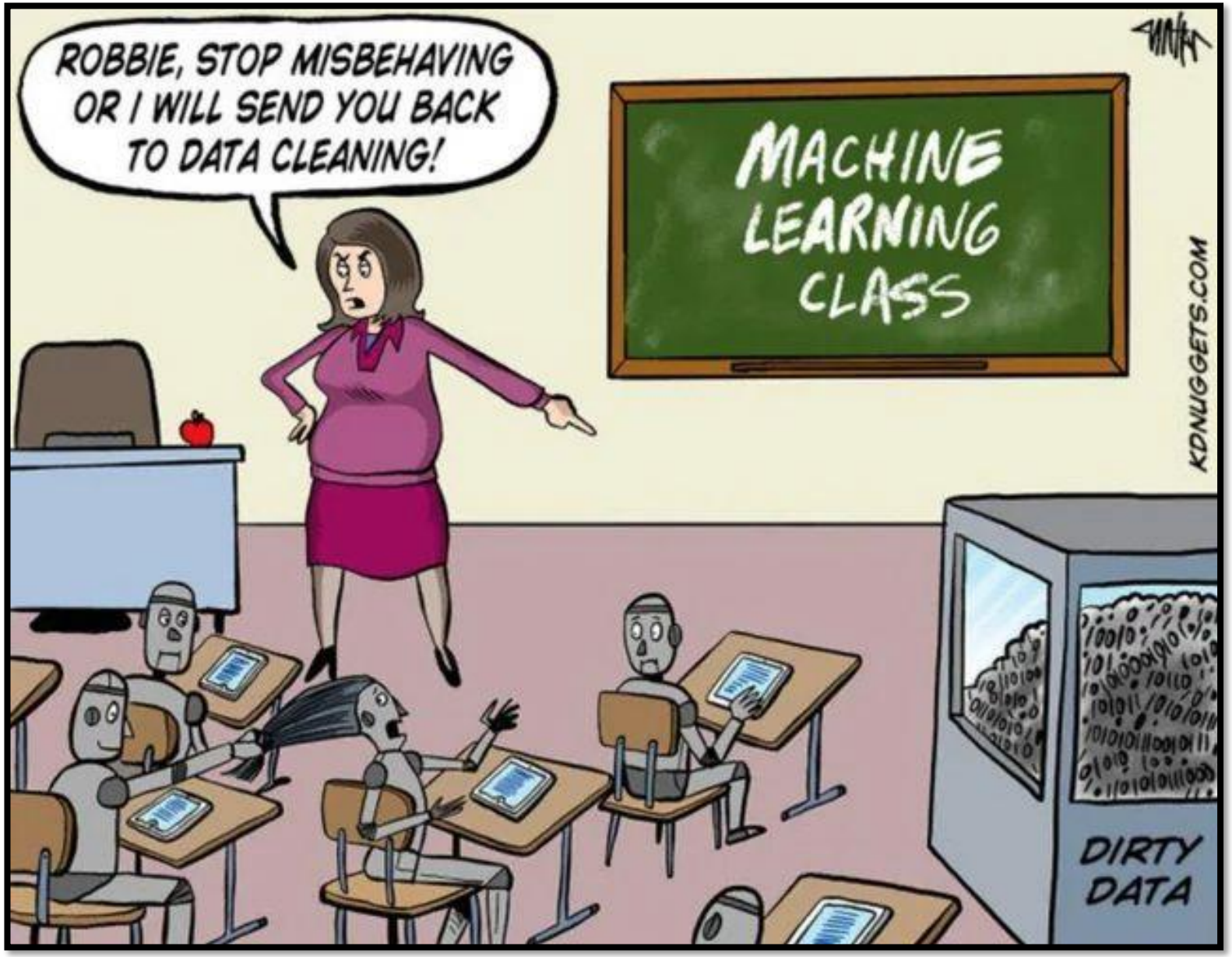


ACC (Accuracy), czyli dokładność
EDA (Exploratory Data Analysis), czyli Eksploracyjna Analiza Danych
KPI (Key Performance Indicator), czyli Kluczowy Wskaźnik Efektywności,

Nasz projekt AI




LLMS
open
closed
Llama 3.3
Phi-4
Gemma 2
Qwen 2.5
Mistral
OpenAI
Claude
Gemini
Cohere
Amazon
Frameworks
LangChain
Llama Index
Haystack
txtai
Vector Databases
Chroma
Pincone
Qdrant
Weaviate
Milvus
Data Extraction
Web
Documents
Crawl4AI
FireCrawl
ScrapeGraphAI
MegaParser
Docling
Llama Parse
ExtractThinker
Open LLMs Access
Hugging Face
Ollama
Groq
Together AI
Text Embeddings
open
closed
NOMIC
SBERT
BGE
Ollama
OpenAI
Voyage AI
Google
Cohere
Evaluation
Giskard
ragas


ROBBIE, STOP MISBEHAVING OR I WILL SEND YOU BACK TO DATA CLEANING!
MACHINE LEARNING CLASS
KDNUGGETS.COM
DIRTY DATA