

# Uczenie maszynowe - notatki

Wstęp .....	5
Uczenie półnadzorowane .....	11
Redukcja Wymiarów .....	11
Ważne Pojęcia .....	12
Aspekty Teoretyczne .....	12
Historia Uczenia Maszynowego .....	13
Regresja .....	16
Rozwiązania Analityczne .....	17
Regresja Wielomianowa .....	19
Ocena Złożoności Modelu .....	19
Złożoność Uczenia i Testowania .....	20
Najważniejsze Pojęcia .....	20
Gradient Descent (Metoda spadku gradientu) .....	20
Pułapki funkcji kosztu .....	21
Pożądana funkcja kosztu .....	21
Skalowanie cech .....	22
Batch Gradient Descent .....	22
Stochastic Gradient Descent (SGD) .....	22
Mini-Batch Gradient Descent .....	22

Regularyzacja .....	23
Ogólne informacje .....	23
Funkcja straty/kosztu podczas fazy treningowej/testowej .....	23
Ridge Regression .....	24
Lasso Regression.....	24
Elastic Net .....	25
Early Stopping .....	25
Podsumowanie regularizacji regresji liniowej .....	26
Multi-class/Label/Output.....	26
Klasyfikacja wieloklasowa (Multi-Class Classification) .....	26
Klasyfikacja wieloetykietowa (Multi-Label Classification).....	27
Klasyfikacja wielowyjściowa (Multi-Output Classification) .....	28
Funkcje straty (Loss functions).....	28
Motywacje.....	28
Funkcje straty regresji .....	30
Przykłady Funkcji Straty.....	31
Opis Pojęć.....	31
Metryki (Metrics) .....	32
Ogólne Kryteria Oceny Modelu.....	32
Dlaczego Używamy Różnych Metryk? .....	32
Porady .....	33
Typy Metryk .....	33
Popularne Metryki.....	33
Metryki Regresji .....	33
Metryki Klasyfikacji .....	34
Przykłady Obliczeń.....	36
Opis Pojęć.....	36
Uczenie Bayesowskie .....	37
Wprowadzenie do podejmowania decyzji .....	37
Wnioskowanie.....	37
Podejmowanie racjonalnych decyzji przy niepewności .....	38
Funkcja użyteczności .....	38

Teoria Bayesa .....	38
Kluczowe pojęcia .....	39
Klasyfikatory probabilistyczne .....	39
Klasyfikatory Bayesowskie.....	39
Bayesian Optimal Classifier (BOC) .....	40
Optymalizacja Bayesowska .....	40
Bayesian Belief Networks (BBN) .....	40
Minimum Description Length (MDL) .....	41
Naive Bayes Classifier (NBC) .....	41
Wnioski.....	41
Podsumowanie.....	41
Regresja Logistyczna (Logistic Regression) .....	42
Wprowadzenie do regresji logistycznej .....	42
Model regresji logistycznej.....	42
Trening modelu.....	43
Funkcja kosztu .....	43
Rozwiązanie .....	44
Regularizacja.....	45
Wprowadzenie do regresji softmax.....	45
Model regresji softmax .....	45
Predykcja .....	46
Funkcja kosztu .....	46
Rozwiązanie .....	47
Wyjaśnienia kluczowych pojęć.....	47
PCA (Principal Component Analysis) .....	48
Wprowadzenie do PCA .....	48
Redukcja wymiarów .....	48
Analiza czynnikowa.....	48
Główne składniki: wymagania .....	48
Interpretacja geometryczna .....	49
Założenia PCA .....	49
Kryteria wyboru liczby składników .....	49

Algorytm PCA .....	49
Przykłady zastosowania PCA .....	49
Podsumowanie.....	49
Kluczowe pojęcia .....	49
SVM (Support Vector Machine) .....	51
Kluczowe Pojęcia.....	56
Drzewa decyzyjne.....	58
Struktura drzewa decyzyjnego.....	58
Atrybuty .....	58
Możliwe porządki rozwijania węzłów .....	58
Kryteria podziału w węzłach .....	59
Główne problemy .....	59
Algorytmy i metryki.....	59
Zalety i wady .....	62
Źródła szumu .....	62
Typy atrybutów .....	63
Kategoryczne/nominalne atrybuty.....	64
Atrybuty numeryczne/ciągłe .....	64
Procedura dla atrybutów numerycznych/ciągłych .....	64
Klasy testów .....	64
Typy testów .....	64
Brakujące wartości atrybutów.....	65
Metryki.....	65
Przeuczenie (Overfitting) .....	65
Przycinanie (Pruning) .....	66
Drzewa decyzyjne obojętne (Oblivious Decision Trees) .....	68
Fuzzy Decision Trees.....	69
Drzewa decyzyjne oparte na algorytmach ewolucyjnych .....	69
Kompromis obciążenie - wariancja .....	70
Podstawowe formalizmy .....	70
Rozkład błędu.....	71
Redukcja wariancji.....	73

Regresja a klasyfikacja .....	73
Metryki oceny modeli .....	73
Logika zdań (CNF/DNF).....	73
Wprowadzenie do logiki.....	74
Zastosowania logiki .....	74
Rodzaje logiki .....	75
Problemy w zastosowaniu logiki.....	75
Definicje podstawowe.....	75
Logika zdań (propositional logic) .....	76
Składnia logiki zdań .....	76
Spójniki logiczne.....	77
Semantyka logiki zdań.....	77
Prawa logiczne i transformacje .....	78
Dowodzenie w logice zdań.....	78
Analiza Dyskryminacyjna (Discriminant Analysis) .....	79
Najważniejsze punkty:.....	81

## Wstęp

### Proces uczenia maszynowego (ML Process)

- Dane (data)- uczenie maszynowe polega na nauce z przykładów. Każdy przykład składa się z cech i wartości tych cech. Macierz X zawiera wektory cech dla wszystkich przykładów treningowych.

$$\mathbf{x}^{(i)} = [x_1^{(i)}, x_2^{(i)}, \dots, x_j^{(i)}, \dots, x_m^{(i)}]$$

- $n$  number of training examples,  $i=1, \dots, n$
- $m$  number of features,  $j=1, \dots, m$
- $x_j^{(i)}$  value of the  $j$ -th feature of the  $i$ -th sample
- $\mathbf{x}^{(i)}$  vector of features of the  $i$ -th sample
- $\mathbf{X}$  matrix ( $n \times m$ ) of vectors of features
- Etykieta (label) - celem jest przewidywanie etykiet, które mogą być binarne (0/1) lub ciągłe.
- $y^{(i)} = t_i$  label of the  $i$ -th sample
- $\mathbf{Y} = [y^{(1)}, y^{(2)}, \dots, y^{(i)}, \dots, y^{(n)}]^T$
- Features (cechy) - inne nazwy: atrybut, zmienna.

#### Hipoteza uczenia (Learnig Hypothesis)

- **Definicja:** Hipoteza jest funkcją  $h: X \rightarrow Y$ , gdzie  $Y$  to etykiety (labels).  
W klasyfikacji  $X = \mathbb{R}^m$ ,  $Y = \{1, 2, 3 \dots k\}$ , gdzie  $k$  = klasy etykiet.  
W regresji  $X = \mathbb{R}^m$ ,  $Y = \mathbb{R}$
- Modele parametryczne: Zakładają określoną formę funkcji  $f(x) = y$  np. regresja liniowa.

- Modele nieparametryczne: Nie mają z góry określonej liczby parametrów, liczba parametrów rośnie wraz ze wzrostem danych testowych np. drzewa decyzyjne.

#### Notacja statystyczna (Statistical Learning Notation)

- Zbiór treningowy (training set): zbiór składający się z par  $(x, y)$ , gdzie  $x$  to wektor cech, a  $y$  to etykiety. Celem jest oszacowanie funkcji  $f$ , która pozwoli przewidywać  $Y$  na podstawie  $X$ .

#### Modele dyskryminacyjne i generatywne

- Modele dyskryminacyjne – bezpośrednio modelują  $P(Y|X)$  i są łatwiejsze do obliczenia oraz dokładniejsze.
- Modele generatywne – modelują wspólne rozkłady  $P(X, Y)$ , dostarczają więcej informacji i pozwalają na próbkowanie z rozkładu.

#### Kroki budowania modelu

- Specyfikacja problemu: klasyfikacja, regresja, uczenie ze wzmocnieniem, grupowanie, inżynieria cech.
- Dane: przetwarzanie wstępne, podział na zestawy treningowy, walidacyjny i testowy.
- Cechy: ekstrakcja cech, selekcja cech, redukcja wymiarów.
- Wybór modelu: ustawienie hiperparametrów, inicjalizacja parametrów.
- Trening: budowanie modelu na zestawie treningowym, wykorzystanie CV (Cross validation).
- Testowanie: ocena modelu (np. wydajność) na zestawie testowym używając określonych parametrów.
- Dostrajanie modelu: modyfikacja hiperparametrów (manualnie, grid search, random search, bayesian search) albo zacznij od początku.

## steps to use the model

- problem specification: prediction / decision making / feature selection / problem reformulation
- data
  - ✓ new/unknown samples
  - ✓ preprocessing
- features
  - ✓ feature extraction / selection
  - ✓ dimension reduction
- model preparation
  - ✓ initializing determined parameters (e.g. weights in the neural network)
- model usage
  - ✓ result: class, value / making a decision / new feature set / solution

1

## Klasyfikacje algorytmów uczenia maszynowego

- Modele liniowe: np. regresja logistyczna.
- Maszyny wektorów nośnych (SVM): np. SVM liniowe, SVM z jądrem RBF.
- Sztuczne sieci neuronowe: np. perceptrony wielowarstwowe.
- Modele drzew i reguł: np. drzewa decyzyjne.
- Modele grafowe: np. sieci bayesowskie.
- Ensembles: np. lasy losowe.
- Uczenie na przykładach (instance-based learning): np. K-Najbliższych Sąsiadów (KNN)

## Komponenty algorytmów uczenia maszynowego

- Reprezentacja,
- Optymalizacja,
- Ocena.
- representation: which hypotheses can be represented given a certain algorithm class
- optimization: optimization metric that is used to fit the model



**criteria** (they are not exclusive)

- human supervision: ML supervised, unsupervised, semisupervised, reinforcement learning
- how much training data is used at once: all data (batch learning) or incremental data (online learning)
- how training data is used: by comparing it directly with the new data (instance-based learning) or by detecting patterns in the training data and building a predictive model for the new data (model-based learning)

Uczenie nadzorowane, nienadzorowane, wsadowe, wzmocnione, online

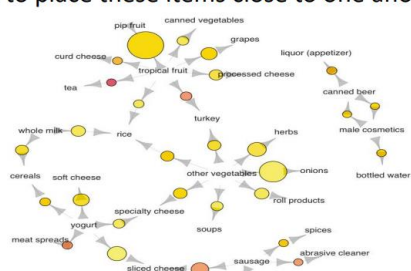
- Uczenie nadzorowane – przewidywanie etykiet na podstawie cech. Może być używane do klasyfikacji (etykiety dyskretne) lub regresji (etykiety ciągłe).
- Uczenie nienadzorowane – dane treningowe są bez etykier, odkrywanie ukrytych struktur w danych np. grupowanie, redukcja wymiarów, wykrywanie anomalii.

### association rule learning

- goal: to dig into large amounts of data and discover interesting relations between attributes
- example: people who purchase barbecue sauce and potato chips also tend to buy steak → you may want to place these items close to one another

- types

- ✓ Apriori
- ✓ Eclat



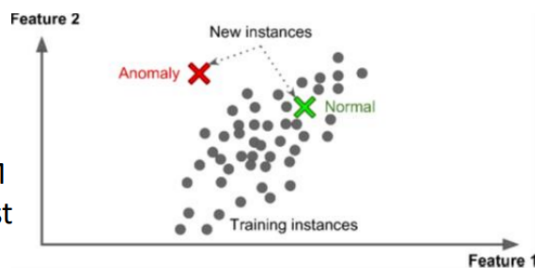
### dimensionality reduction (DR)

- goal: to simplify data without losing too much information
- it is often a good idea to try to reduce the dimension of the training data before feeding it to another ML algorithm
  - ✓ faster learning
  - ✓ less disk and memory space for data
  - ✓ (in some cases) better performance
- simple DR method: to merge several correlated features into one
- example (feature extraction)  
a car's feature1="mileage" may be strongly correlated with its feature2="age" → after dimensionality reduction: feature="car wear and tear"

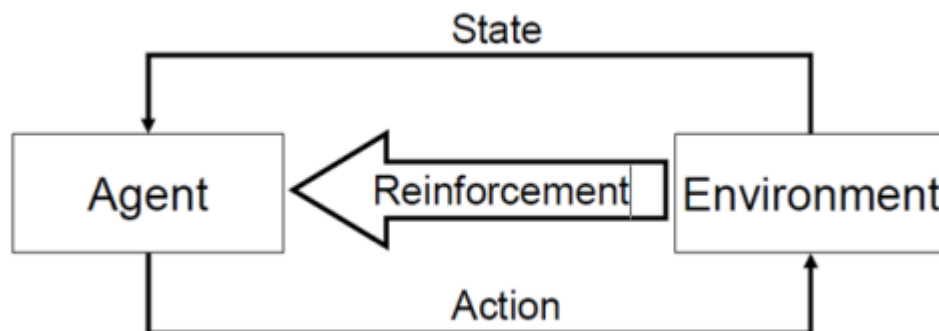
34

### anomaly/novelty detection

- algorithms
  - ✓ One-class SVM
  - ✓ Isolation Forest
- examples
  - ✓ detecting unusual credit card transactions to prevent fraud
  - ✓ catching manufacturing defects
  - ✓ automatically removing outliers from a dataset before feeding it to another learning algorithm



- Uczenie wzmacnione: Agent – uczy się strategii maksymalizującej nagrody w dynamicznym środowisku (obserwuje środowisko, wybiera i wykonuje akcje, dostaje w zamian nagrody/kary) np. robot uczący się chodzić, program AlphaGo.



- Uczenie wsadowe (batch learning) – model trenowany na całym zbiorze danych, co jest czasochłonne i wymaga dużych zasobów. Najczęściej wykonywane offline.

- Uczenie przyrostowe (incremental learning) – model trenowany na bieżąco na nowych danych (na małych grupach danych zwanych mini-batch) co jest szybsze i oszczędniejsze, szczególnie przy dużych strumieniach danych.
- proces: algorithm loads part of the data, runs a training step on that data, and repeats the process until it has run on all of the data

### **Uczenie półnadzorowane**

**Etykietowanie danych jest zazwyczaj czasochłonne i kosztowne.**

**Niektóre algorytmy mogą radzić sobie z danymi, które są częściowo etykietowane.**

**Większość algorytmów uczenia półnadzorowanego to kombinacje algorytmów nienadzorowanych i nadzorowanych.**

**Przykład: sieci głębokiego przekonania (DBN):**

- DBN bazują na nienadzorowanych komponentach zwanych ograniczonymi maszynami Boltzmanna (RBM) ustawionymi jedna na drugiej.
- RBM są trenowane sekwencyjnie w trybie nienadzorowanym, a następnie cały system jest dostrajany przy użyciu technik uczenia nadzorowanego.

### **Redukcja Wymiarów**

- Metody: Analiza głównych składowych (PCA), analiza czynnikowa, t-SNE, ISOMAP, autoenkodery.
- Cel: Upraszczanie danych przy minimalnej utracie informacji, co prowadzi do szybszego uczenia i mniejszego zużycia zasobów.

## learning rate

- important parameter of online learning systems
- how fast a system should adapt to changing data
- high learning rate
  - ✓ system adapts to new data, but also tends to quickly forget the old data
  - ✓ negative example: you don't want a spam filter to flag only the latest kinds of spam it was shown
- low learning rate
  - ✓ system learns more slowly, but is also less sensitive to noise in the new data or to sequences of nonrepresentative data points (outliers)

## Ważne Pojęcia

- **Równowaga interpretowalności i dokładności:** Prostota modelu vs. jego dokładność.
- **Brzytwa Ockhama:** Preferowanie prostszych modeli, jeśli są równie dokładne.
- **Teoria "No Free Lunch":** Nie ma jednego najlepszego algorytmu dla wszystkich problemów.

## Aspekty Teoretyczne

### ML vs Programowanie

- Tradycyjne programowanie: tworzenie programów do automatyzacji procesów.
- Uczenie maszynowe: przekształcenie danych w programy, komputery uczą się bez potrzeby programowania ich wprost.

### Uczenie maszynowe

- **Definicja:** Algorytmy, które zdobywają wiedzę z przykładów treningowych i używają jej do klasyfikacji, oceny lub przewidywania

nieznanych przypadków testowych. Algorytmy, które generalizują na podstawie przykładów.

## Zastosowanie uczenia maszynowego

- **Rozpoznawanie wzorców:** Klasyfikacja, regresja, kategoryzacja, metody aproksymacji.
- **Optymalizacja:** Modele mające (meta)parametry do optymalizacji.
- **Generowanie wzorców:** Generowanie sygnałów, np. obrazy syntetyczne.
- **Predykcja:** Przyszłe ceny akcji, kursy walut itp.
- **Wykrywanie anomalii:** Niezwykłe wzorce w danych, np. transakcje kartą kredytową.

## Historia Uczenia Maszynowego

- **Era przedkomputerowa:** Intuicja i analiza manualna.
- **Lata 70-te:** Języki programowania statystycznego.
- **Lata 90-te:** Oprogramowanie statystyczne z interfejsem graficznym.
- **Lata 2000:** Zoptymalizowane środowisko obliczeniowe.
- **Lata 2010:** Pakiety Python do manipulacji danymi i uczenia modeli.
- **Lata 2020:** Chmura obliczeniowa.

## Wyzwania uczenia maszynowego

- Główne przyczyny: złe dane, zły algorytm.
- Niewystarczająca ilość danych treningowych: proste problemy wymagają tysięcy przykładów, a rozpoznawanie obrazów czy mowy milionów.
- Nieprzedstawicielskie (nonrepresentative) dane treningowe: dane muszą być reprezentatywne dla nowych przypadków, aby model generalizował poprawnie. Nieprzedstawicielskie dane mogą być wynikiem przypadku, szumu, zbyt małej próby. Model wytrenowany na

nieprzedstawicielskim zbiorze treningowym prawdopodobnie nie będzie dokonywał dokładnych przewidywań.

- Słaba jakość danych: niskiej jakości dane utrudniają wykrywanie wzorców, co pogarsza wydajność modelu.
  - advice: clean up your training data  
(in ML a lot of the time is spent on data cleaning)
  - outliers: discard or try to fix the errors in the model manually
  - some instances are missing a few features: ignore this attribute altogether, ignore these instances, fill in the missing values (mean/median), or train one model with the feature and one model without it
- Zbyt wiele lub zbyt mało cech: Wybór i ekstrakcja cech są kluczowe dla wydajności modelu.
- Wybór odpowiedniego modelu.
- Overfitting: model działa dobrze na danych treningowych, ale nie generalizuje dobrze na nowych danych. Dzieje się tak, gdy model jest zbyt skomplikowany w stosunku do ilości i szumu danych treningowych.
  - Rozwiązania: redukcja liczby cech, regularizacja modelu, zgromadzenie większej ilości danych treningowych, redukcja szumu w danych treningowych, wybór modelu z mniejszą liczbą parametrów.
- Underfitting: występuje, gdy model jest zbyt prosty, aby nauczyć się podstawowej struktury danych.
  - Rozwiązania: wybór modelu z większą ilością parametrów, dostarczanie lepszych cech do algorytmu uczącego się (inżyniera cech), zmniejszenie ograniczeń nałożonych na model.
- Regularizacja: uproszczenie modelu/wprowadzenie ograniczeń na model, zmniejsza ryzyko przeuczenia (overfitting), ilość regularizacji stosowanej podczas uczenia może być kontrolowana przez hiperparametr.

## Wybór i walidacja modelu

- Przetestowanie modelu: podział danych na zestaw treningowy i testowy, ocena błędu generalizacji.
- Walidacja krzyżowa: używanie wielu małych zestawów walidacyjnych dla dokładniejszej oceny modelu.
- Dostrajanie modelu: manualne dostrajanie hiperparametrów, wyszukiwanie siatki, wyszukiwanie losowe, metody bayesowskie.

### MODEL FINE-TUNING

#### manual search

- play with the hyperparameters manually, until you find the best combination
- very tedious work, it takes a lot of time to explore all/many combinations

#### grid search

- automatic check of many combinations
- if no hints for hyperparameters values → try out consecutive powers of 10
- is fine when you are exploring relatively few combinations

41

#### randomized search

- preferable when the hyperparameter search space is large

#### Bayesian search

- minimize a scalar objective function in a bounded domain
- the function can be deterministic or stochastic

#### ensemble methods

- types: bagging, boosting, random forests, ...
- the ensemble (group) of models often performs better than the best individual model

42

## Trudności w uczeniu maszynowym

Kłątwa wymiarowości: Wzrost wymiarowości powoduje, że dane stają się rzadsze. Konieczność posiadania reprezentatywnych przykładów z wszystkich istotnych wariacji.

## Regresja

### Przykład Regresji

- Dane: (PKB na mieszkańca, satysfakcja z życia)
- Hipoteza: regresja liniowa.
- Model:  $\text{satysfakcja\_z\_zycia} = O_0 + O_1 \times \text{PKB}$ .
- Funkcja wydajności: funkcja kosztu mierząca **odległość między przewidywaniami modelu liniowego, a przykładami treningowymi**.
- Trening: znalezienie wartości parametrów  $O_0$  i  $O_1$  minimalizujących funkcję kosztu.
- Predykcja na nowych przypadkach (inference):
  - Przykład: Cypr
  - $\text{PKB\_na\_mieszkańca} = 22,587 \$$
  - $\text{Satysfakcja\_z\_zycia} = 4.85 + 22,587 \times 4.91 \times 10^{-5} = 5.96$ .

### Model regresji liniowej

- Regresja liniowa: jeden z najprostszych modeli.
- Formulacja problemu: znaleźć parametry  $\theta_j$ ,  $j \in [1, m]$  z danych treningowych  $x(i)$ ,  $i \in [1, n]$  aby najlepiej dopasować:
  - $y(i) = \theta_0 + \theta_1 x_1(i) + \theta_2 x_2(i) + \dots + \theta_m x_m(i)$
- Użycie modelu (testowanie): znaleźć oszacowane  $y$  dla danego wektora  $x = [x_1, x_2, \dots, x_m]$ :
  - $y = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \dots + \theta_m x_m$
- **Zmienne**:
  - $y$  - przewidywana wartość
  - $x_j$  - wartość  $j$ -tej cechy



- $\theta_j$  - parametr j-tego modelu ( $\theta_0$  - przesunięcie/bias,  $\theta_1, \theta_2, \dots, \theta_m$  - wagi cech)
- $m$  - liczba cech
- $n$  - liczba próbek
- $\alpha$  - współczynnik uczenia

## Rozwiązania Analityczne

- Problem optymalizacyjny:
  - Funkcje kosztu:
    - RMSE (Root Mean Square Error)
    - MSE (Mean Squared Error): prostsza do minimalizacji
    - MAE (Mean Absolute Error)

## optimization task

- find the value of  $\theta$  that minimizes MSE

$$L(y, \hat{y}) = \text{MSE}(y, \hat{y}) = \frac{1}{n} \sum_{i=1}^n (y^{(i)} - \hat{y}^{(i)})^2$$

## Podjęcie Algebraiczne przez Równanie Normalne

- System równań liniowych przez w notacji macierzowej:

$$X\theta = Y$$

$$X = \begin{pmatrix} x_0^{(1)} & x_1^{(1)} & x_2^{(1)} & \cdots & x_m^{(1)} \\ x_0^{(2)} & x_1^{(2)} & x_2^{(2)} & \cdots & x_m^{(2)} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ x_0^{(n)} & x_1^{(n)} & x_2^{(n)} & \cdots & x_m^{(n)} \end{pmatrix}$$

$$\theta = \begin{pmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_m \end{pmatrix} \quad Y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}$$

- Problemy:  $XTX$  może nie być odwracalne; rozwiązanie może nie istnieć bądź nie być unikalne.

Podejście Algebraiczne przez SVD (Rozkład wartości osobliwych)

- Rozwiązanie:
  - decomposition:  $X = U\Sigma V^T$ ,  $\dim X = \dim \Sigma$   
 $U = XX^T$  orthogonal matrix  
 $V = X^T X$  orthogonal matrix  
 $\Sigma$ : rectangular diagonal matrix of square roots of the non-zero eigenvalues of  $U$  or  $V$  (are identical)  
 $\sigma_i = \sqrt{\lambda_i}$  in descending order  $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_m \geq 0$   
 (add zero to the remaining rows/columns)

$$\begin{aligned} X\theta &= Y \rightarrow \theta = X^{-1}Y \\ X^{-1} &= [U\Sigma V^T]^{-1} = V\Sigma^{-1}U^T \\ \theta &= V\Sigma^{-1}U^T Y \end{aligned}$$

$$\Sigma^{-1} = [\text{diag}(\Sigma^T)]^{-1}$$

invert only non-zero elements of the diagonal of  $\Sigma^T$  <sup>24</sup>

- Testowanie: predykcje są bardzo szybkie, liczba instancji –  $O(n)$ , liczba cech –  $O(m)$ , jest efektywniejsze od równania normalnego.

Rozwiązanie SVD: zalety

- standardowa technika faktoryzacji macierzy

- pseudoodwrotność jest zawsze zdefiniowana
- jest bardziej efektywna niż obliczanie równania normalnego

### Rozwiązanie przez Gradient Descent

- Uczenie maszynowe: nie obliczamy bezpośrednio, lecz trenujemy model, aby najlepiej dopasować go do zbioru treningowego.
- Metoda: minimalizacja funkcji kosztu.
- Podejście: optymalizacja iteracyjna.
- Aktualizacja parametrów.
- **update the current value of  $\theta_j$ :**

$$\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} L(\theta)$$


---

- Działa dobrze przy dużej liczbie próbek (n).
- Dokładność zależy od liczby iteracji.
- Może działać z dowolną funkcją straty i dowolnym wynikiem modelu.
- Potrzebne jest dobra współczynnik uczenia.

### Regresja Wielomianowa

- **Regresja wielomianowa:**
  - $y = f(x)$  jest wielomianem n-tego stopnia w x
  - Istnieje nieliniowa zależność między y a x
  - Jest modelem liniowym: y jest liniowe w nieznanych parametrach, które są szacowane z danych
  - Specjalny przypadek regresji liniowej wielokrotnej

### Ocena Złożoności Modelu

- **Ocena ogólnej wydajności modelu:**
  - Przez walidację krzyżową
  - **Krzywe uczenia:**
    - Trenuj model kilka razy na różnych rozmiarach podzbiorów zbioru treningowego

- Wykreśl wydajność modelu jako funkcję rozmiaru zbioru treningowego (lub iteracji treningowej)

## Złożoność Uczenia i Testowania

- **Normalne równanie i SVD:**
  - Bardzo wolne przy  $m > 10^5$
  - Efektywne przy dużych zbiorach treningowych, pod warunkiem, że mieszczą się w pamięci

## Najważniejsze Pojęcia

- **Funkcja kosztu:** miara jakości modelu, np. MSE
- **Regresja liniowa:** przewidywanie wartości rzeczywistych na podstawie liniowej kombinacji cech
- **Regresja wielomianowa:** przewidywanie wartości rzeczywistych na podstawie wielomianowej kombinacji cech
- **Optymalizacja iteracyjna (Gradient Descent):** stopniowe dostosowywanie parametrów modelu w celu minimalizacji funkcji kosztu

## Gradient Descent (Metoda spadku gradientu)

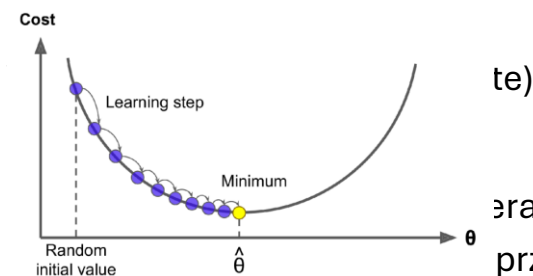
### Ogólne informacje

- Gradient Descent (GD) - metoda optymalizacji parametrów modelu.
  - Mierzy lokalny gradient funkcji kosztu względem parametrów modelu.
  - Strategia: Iteracyjne poruszanie się w kierunku opadającego gradientu funkcji kosztu.
  - Gradient zerowy oznacza minimum.
  - Wielkość kroku uczenia proporcjonalna do gradientu funkcji kosztu.
- Algorytm GD:

- Inicjalizacja modelu z wartościami losowymi,
- Krok uczenia: zmniejszenie funkcji kosztu,
- Warunek końcowy: algorytm zmierza do minimum.

## GD w regresji liniowej

- Algorytm:
  - Wypełnienie parametrów  $\theta$  losowymi wartościami.
  - Krok uczenia: zmniejszenie funkcji kosztu MSE.
  - Warunek końcowy: algorytm zmierza do minimum.



te)

tracą do konwergencji, długi czas.

przeskakiwać optymalne rozwiązanie, brak

znajdzonej rozwiązania.

## Pułapki funkcji kosztu

- Lokalne minimum: możliwość konwergencji do lokalnego minimum.
- Płaskowyzę: długie uczenie.
- Zbyt wczesne zatrzymanie: brak optymalnego rozwiązania.
- Dziury, grzbiety, nieregularne tereny: trudna konwergencja.

## Pożądana funkcja kosztu

- Funkcja wypukła: segment łączący dowolne dwa punkty na krzywej nigdy jej nie przecina.
- Jedno globalne minimum, brak lokalnych minimów.
- Funkcja ciągła z łagodnym nachyleniem.

## Skalowanie cech

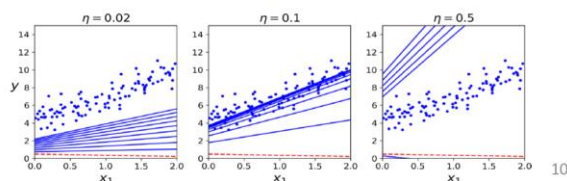
- Różne skale cech: wolna konwergencja, wolne zmierzanie do minimum.
- Podobna skala cech: szybka konwergencja.

## Batch Gradient Descent

- Funkcja kosztu: Średni Błąd Kwadratowy (MSE).
- Algorytm: używa całego zbioru treningowego w każdym kroku.
- Wolny na bardzo dużych zbiorach treningowych, ale dobrze skalowalny z liczbą cech.

Formuła GD:  $\theta$  (next step) =  $\theta - \eta \nabla \theta \text{MSE}(\theta)$

- learning rate  $\eta$ 
  - ✓ too low: long time to reach the optimal solution
  - ✓ good: convergence in a few iterations
  - ✓ too high: algorithm diverges at every step
  - ✓ method to find an optimal one: grid search



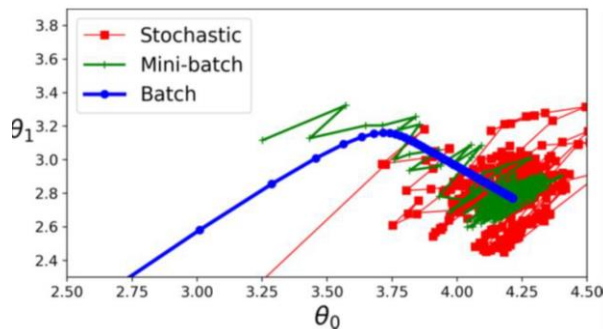
## Stochastic Gradient Descent (SGD)

- Przetwarza instancje treningowe niezależnie, po jednej na raz.
- „Stochastic”: wybiera losową instancję w każdym kroku.
- Bardzo szybki, dobry do nauki online.
- Może być implementowany jako algorytm „out-of-core”.
- Wady: koszt funkcji skacze w górę i w dół, zmniejszając się tylko średnio; jeśli się zatrzyma, końcowy parametr jest dobry, ale nie optymalny.
- Zalety: zdolność ucieczki z lokalnych minimów; jest lepszy w poszukiwaniu globalnego minimum od GD.

## Mini-Batch Gradient Descent

- Oblicza gradienty na małych, losowych zestawach instancji zwanych mini-batchami.

- Zalety: optymalizacja sprzętowa operacji macierzowych (GPU), mniej chaotyczny postęp algorytmu, bliżej minimum.
- Wady: trudniej uciec z lokalnych minimów.



## Regularyzacja

Ogólne informacje

- Regularyzacja – ~~używana do ograniczenia/redukcji parametrów modelu~~ (np. współczynniki funkcji dopasowania).
  - Redukuje overfitting poprzez zmniejszenie stopni swobody (wariancji modelu).
  - Metody: Ridge regression, Lasso regression, Elastic net.

Funkcja straty/kosztu podczas fazy treningowej/testowej

- Funkcja straty/kosztu w fazie treningowej **różni się od metryki wydajności w fazie testowej**.
- Trening:
  - Funkcja straty/kosztu powinna mieć przyjazne dla optymalizacji pochodne.
  - Dodawana regularyzacja.
- Testowanie:
  - Metryka wydajności powinna być jak najbliższa końcowemu celowi.
  - Bez terminu regularyzacyjnego.

## Ridge Regression

- Znana również jako Tikhonov regularization.
- Dodaje termin regularyzacyjny do funkcji kosztu:

$$\alpha \sum_{i=1}^m \theta_i^2$$

- Używa normy L2 wektora wag.
- Utrzymuje wagi modelu jak najmniejsze.
- Powinna być dodawana do funkcji kosztu tylko w fazie treningu.
- Przykład:
  - Funkcja kosztu treningowego:  $L(\theta) = \text{MSE}(\theta) + \alpha/2 * \text{suma}(\theta_i^2)$ .
  - Parametr  $\alpha$  kontroluje siłę regularyzacji.
- Gradient Descent:
  - $\nabla \theta L(\theta) = 2/n * X^T * (X\theta - y) + \alpha\theta$ .
- Przed przeprowadzeniem Ridge Regression dane muszą być skalowane.

## Lasso Regression

- Least Absolute Shrinkage and Selection Operator Regression.
- Dodaje termin regularyzacyjny do funkcji kosztu:

$$\alpha \sum_{i=1}^m |\theta_i|$$

- Używa normy L1 wektora wag.



- Skłania się do ustawiania wag najmniej istotnych cech na zero, automatycznie wykonując selekcję cech i wyjściem jest model sparsowany.
- Przykład:
  - Funkcja kosztu treningowego:  $L(\theta) = \text{MSE}(\theta) + \alpha * \text{suma}(|\theta_i|)$ .
- Gradient Descent:
  - Funkcja kosztu Lasso nie jest różniczkowalna w  $\theta_i = 0$ . Dla takich  $\theta_i$  używany jest wektor subgradientu.

### Elastic Net

- Kombinacja **terminów regularizacyjnych Ridge i Lasso.**
- Współczynnik mieszania  $r$ :
  - $r = 0$ : Elastic Net = Ridge Regression.
  - $r = 1$ : Elastic Net = Lasso Regression.
- Funkcja kosztu:  $L(\theta) = \text{MSE}(\theta) + r\alpha * \text{suma}(|\theta_i|) + (1-r) / 2 * \alpha * \text{suma}(\theta_i^2)$ .

### Early Stopping

- Prosta i efektywna technika regularyzacji.
- Błąd (RMSE) na zbiorze treningowym maleje wraz z epokami.
- Błąd (RMSE) na zbiorze walidacyjnym maleje, osiąga minimum, a następnie wzrasta (model zaczyna overfitować).
- Zatrzymać trening, gdy błąd walidacyjny osiągnie minimum.

### **main differences Ridge ↔ Lasso**

- in Ridge the gradients get smaller as the parameters approach the global optimum → GD naturally slows down → no bouncing around → good convergence
- increased  $\alpha$ 
  - ✓ optimal parameters (red square) get closer to the origin
  - ✓ in Ridge parameters can never be eliminated

## Podsumowanie regularizacji regresji liniowej

- **Ridge:** Dobry domyślny wybór.
- **Lasso:**
  - Dobrze, gdy tylko kilka cech jest użytecznych (redukuje wagi nieużytecznych cech do zera).
  - Może zachowywać się nieregularnie, gdy liczba cech jest większa niż liczba instancji treningowych lub gdy kilka cech jest silnie skorelowanych.
- **Elastic Net:**
  - Preferowany, gdy tylko kilka cech jest użytecznych (redukuje wagi nieużytecznych cech do zera).

## **Multi-class/Label/Output**

### Klasyfikacja wieloklasowa (Multi-Class Classification)

- Znana również jako klasyfikacja wielomianowa.
- Każda próbka może należeć do jednej z  $C > 2$  klas.
- Sieć neuronowa:  $C$  neurony wyjściowe.
- Wektor docelowy  $y(i)$  to wektor one-hot z jedną klasą pozytywną ( $=1$ ) i  $C-1$  klasami negatywnymi ( $=0$ ).
- Zadanie traktowane jako pojedynczy problem klasyfikacji próbek w jednej z  $C$  klas.

### **Klasyfikatory obsługujące wiele klas natywnie:**

- SGD (Stochastic Gradient Descent)
- Random Forest
- Naive Bayes

### **Ściśle binarne klasyfikatory:**

- Logistic Regression
- SVM (Support Vector Machine)

## Strategie używania binarnych klasyfikatorów do klasyfikacji wieloklasowej:

- **One-Versus-the-Rest (OvR) / One-Versus-All:**
  - Preferowana metoda dla algorytmów klasyfikacji binarnej.
  - Przykład: klasyfikacja obrazów cyfr (10 klas: 0-9). Trenuje się 10 binarnych klasyfikatorów, każdy dla jednej cyfry.
  - Klasyfikacja: decyzje od każdego klasyfikatora, wybór klasy z najwyższym wynikiem.
- **One-Versus-One (OvO):**
  - N klas:  $N \times (N-1) / 2$  klasyfikatorów.
  - Zaleta: każdy klasyfikator trenuje się tylko na części zbioru treningowego dla dwóch klas, które musi rozróżniać.
  - Preferowane dla algorytmów, gdzie szybciej trenuje się wiele klasyfikatorów na małych zbiorach treningowych (np. SVM).
  - Przykład: trenuje się binarny klasyfikator dla każdej pary cyfr. Klasyfikacja: testowanie 45 klasyfikatorów i wybór cyfry, która wygra najwięcej pojedynków.

## Klasyfikacja wieloetykietowa (Multi-Label Classification)

- Każda próbka może należeć do więcej niż jednej klasy.
- Sieć neuronowa: C neurony wyjściowe.
- Wektor docelowy  $y$  może mieć więcej niż jedną klasę pozytywną, więc będzie to wektor  $C$ -wymiarowy z 0 i 1.
- Zadanie traktowane jako  $C$  różne i niezależne problemy klasyfikacji binarnej ( $y_{i,k} = 0$  lub  $y_{i,k} = 1$ ), gdzie każdy neuron wyjściowy decyduje, czy próbka należy do klasy, czy nie.

Przykład: Rozpoznawanie twarzy

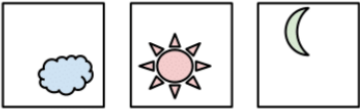

- Wiele osób na tym samym zdjęciu: przypisanie jednej etykiety na rozpoznaną osobę.
- Klasyfikator uczony do rozpoznawania trzech twarzy (A, B, C). Pokazuje się zdjęcie z A i C, powinien zwrócić  $[1, 0, 1]$ .

## Klasyfikacja wielowyjściowa (Multi-Output Classification)

- Znana również jako klasyfikacja wielowyjściowo-wieloklasowa.
- Uogólnienie klasyfikacji wieloetykietowej, gdzie każda etykieta może być wieloklasowa (może mieć więcej niż dwie możliwe wartości).

Przykład: Odzyskiwanie obrazów

- Wejście: obraz cyfry z szumem.
- Wyjście: czysty obraz cyfry, reprezentowany jako tablica intensywności pikseli.
- Wyjście klasyfikatora: wieloetykietowe (jedna etykieta na piksel), każda etykieta może mieć wiele wartości (intensywność piksela w zakresie od 0 do 255).

	Multi-Class	Multi-Label
C = 3	<p>Samples</p>  <p>Labels (t)</p> <p>[0 0 1] [1 0 0] [0 1 0]</p>	<p>Samples</p>  <p>Labels (t)</p> <p>[1 0 1] [0 1 0] [1 1 1]</p>

## Funkcje straty (Loss functions)

### Motywacje

- Uczenie maszynowe:
  - Paradygmat: zbudować model na podstawie danych i użyć go na nowych danych.
  - Faza 1: Trening – ustawienie parametrów modelu.
  - Faza 2: Testowanie – przewidywanie na nowych danych.

- Cel: zminimalizować funkcję straty (kosztu).
- Pomiar wydajności modelu podczas testowania za pomocą metryki wydajności (maksymalizacja).
- Funkcja straty (kosztu):
  - Różne opisy:
    - Używana podczas treningu do optymalizacji modelu.
    - Definiuje cel, względem, którego oceniany jest model.
    - Mapuje decyzje modelu na związane z nimi koszty.
    - Mierzy, jak daleko estymowana wartość jest od rzeczywistej wartości.
  - Wymagania:
    - Powinna mieć przyjazne dla optymalizacji pochodne.
    - Powinna być możliwa do traktowania z terminami regularyzacji.
  - Zachowanie:
    - Jest wysoka, jeśli przewidywania modelu są całkowicie błędne.
    - Jest minimalna (0), jeśli przewidywania modelu są równe celom.
- Czynniki wpływająca na wybór funkcji straty:
  - Typ algorytmu ML.
  - Łatwość obliczania pochodnych.
  - Procent wartości odstających w zbiorze danych.
  - Brak uniwersalnej funkcji straty.
- Metryki wydajności
  - Nazywane "kosztami testowymi".
  - Powinny być jak najbliżej ostatecznego celu.
  - Nie są używane podczas treningu (nie można z nich wyprowadzić "gradientów").
  - Są określane po ustaleniu parametrów modelu.
- Klasyfikacja funkcji straty
  - Regresja:

- MSE: Mean Squared Error
- MAE: Mean Absolute Error
- MAPE: Mean Absolute Percentage Error
- MSLE: Mean Squared Logarithmic Error
- Cosine Similarity
- Huber
- Log Cosh
- Prawdopodobieństwo:
  - Categorical Crossentropy
  - Binary Crossentropy
  - Sparse Categorical Crossentropy
  - Poisson
  - KL Divergence
- Hinge losses dla klasyfikacji "maximum-margin":
  - Hinge
  - Squared Hinge
  - Categorical Hinge

## Funkcje straty regresji

- MSE (Mean Squared Error):
  - Najczęściej używana funkcja straty dla regresji.
  - Wrażliwa na wartości odstające.
- RMSE (Root Mean Square Error):
  - Mniej powszech niż MSE.
  - Minimalizacja RMSE również minimalizuje MSE.
- MAE (Mean Absolute Error)
  - Prosta, ale odporna na wartości odstające (bez kwadratu).
  - Wymaga bardziej skomplikowanych narzędzi do obliczania gradientów.
- MSLE (Mean Squared Logarithmic Error)
  - Penalizuje niedoszacowania bardziej niż przeszacowania.
- Huber Loss
  - Kombinacja L1 i L2, odporny na wartości odstające.

- Poison Loss
  - Używana do danych licznikowych.

## Klasyfikacja

- Cross-Entropy: Używana w klasyfikacji binarnej i wieloklasowej.
- Hinge: Stosowana w maszynach wektorów nośnych (SVM).

## Przykłady Funkcji Straty

- MSE:
  - Inne nazwy: Quadratic Loss, L2 Loss.
  - Łatwe obliczanie gradientów.
  - Wrażliwa na wartości odstające.
- MAE:
  - Inne nazwy: L1 Loss.
  - Odporniejsza na wartości odstające niż MSE.
  - Używana w przypadku multimodalnej dystrybucji danych.
- Huber:
  - Funkcja częściowa.
  - Łączy stabilność MSE z odpornością MAE na wartości odstające.
- Cross-Entropy:
  - Miara różnicy losowości między dwoma zmiennymi losowymi.
  - Stosowana w zadaniach klasyfikacji wieloklasowej.

## Opis Pojęć

- **Funkcja Straty (Loss Function)**: Narzędzie używane **do oceny, jak dobrze model przewiduje rzeczywiste wartości**. Funkcja straty mapuje decyzje modelu na związane z nimi koszty.
- **MSE (Mean Squared Error)**: Średnia kwadratowa różnica między wartościami przewidywanymi a rzeczywistymi. Powszechnie używana w regresji.

- **MAE (Mean Absolute Error):** Średnia absolutna różnica między wartościami przewidywanymi a rzeczywistymi. Odporniejsza na wartości odstające niż MSE.
- **Cross-Entropy:** Miara różnicy losowości między dwoma zmiennymi losowymi, często używana w klasyfikacji binarnej i wieloklasowej.
- **Huber Loss:** Funkcja częściowa, która łączy zalety MSE i MAE, będąc bardziej odporna na wartości odstające.
- **Poisson Loss:** Używana w regresji dla modelowania danych licznikowych, np. liczby klientów w sklepie na dany dzień.
- **Metryki Wydajności (Performance Metrics):** Miary używane do oceny wydajności modelu na nowych danych, np. dokładność, AUC, precyzja, recall.

## Metryki (Metrics)

### Ogólne Kryteria Oceny Modelu

- **Dokładność przewidywania**
- **Efektywność obliczeniowa i skalowalność:**
  - Czas uczenia
  - Zużycie pamięci
- **Odporność:**
  - Na szum
  - Na brakujące wartości
- **Zdolność wyjaśniania:** np. drzewa decyzyjne vs. sieci neuronowe
- **Złożoność hipotezy:** np. rozmiar drzewa, rozmiar sieci neuronowej

### Dlaczego Używamy Różnych Metryk?

- Każda metryka jest odpowiednia dla konkretnego problemu, społeczności lub raportu.



- Przykład: Zdefiniowanie efektywności sklepu internetowego wymaga formalizacji, co to znaczy "efektywność" i określenia metryki do jej mierzenia.

## Porady

- Optymalizuj model dokładnie pod kątem metryki, która jest używana do jego oceny.

## Typy Metryk

- **Metryki dokładności:**
  - Accuracy, Binary Accuracy, Categorical Accuracy, Top-K Categorical Accuracy
- **Metryki probabilistyczne:**
  - Binary Crossentropy, Categorical Crossentropy, KL Divergence, Poisson
- **Metryki regresji:**
  - MSE, RMSE, MAE, MAPE, MSLE, Cosine Similarity, Log Cosh Error
- **Metryki segmentacji obrazów:**
  - Mean IoU
- **Metryki klasyfikacji oparte na TP/TN/FP/FN:**
  - TP, TN, FP, FN, Recall, Specificity, Precision, AUC

## Popularne Metryki

- **Regresja:** MSE, RMSE, MAE, MSLE
- **Klasyfikacja:** error, confusion matrix, accuracy, precision, recall, MCC, F score, Youden, ROC AUC, lift

## Metryki Regresji

- **MSE (Mean Squared Error):**
  - Najczęściej używana metryka regresji.

- Symetryczna i kwadratowa, odpowiednia dla szumów Gaussa.
- Wrażliwa na wartości odstające.
- **RMSE (Root Mean Squared Error):**
  - Korzystniejsza od MSE, ponieważ unika wartości bezwzględnych.
- **MAE (Mean Absolute Error):**
  - Prosta i łatwa do interpretacji.
  - Równe ważenie wszystkich różnic.

#### MAE vs. RMSE

- two most popular metrics for continuous variables
- RMSE has the benefit of penalizing large errors
- generally,  $RMSE \geq MAE$
- $RMSE = MAE$  when all the differences are equal or zero
- one distinct advantage of RMSE over MAE:  
RMSE avoids the use of taking the absolute value

#### Metryki Klasyfikacji

- **Confusion Matrix:**
  - TP (True Positive), TN (True Negative), FP (False Positive), FN (False Negative)
- **Precision (PPV):**
  - Wartość prawdziwych pozytywnych wyników względem wszystkich pozytywnych wyników.

$$PRE = PPV = \frac{TP}{TP + FP}$$

- **Negative Predicted Value (NPV)**

$$NPV = \frac{TN}{TN + FN}$$

- **Recall (Sensitivity):**

- Odsetek prawdziwych pozytywnych wyników wykrytych przez model.

$$\text{sensitivity} = \frac{TP}{TP + FN}$$

- **Specificity:**

- Odsetek prawdziwych negatywnych wyników.

$$\text{specificity} = \frac{TN}{TN + FP}$$

- **False Positive Rate (FPR)**

$$FPR = \frac{FP}{FP + TN}$$

- **False Negative Rate (FNR)**

$$FNR = \frac{FN}{FN + TP}$$

- **F1 Score:**

- Harmoniczna średnia precision i recall.

$$F \text{ score} = 2 \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

- **AUC (Area Under the ROC Curve):**
  - Miernik jakości modelu klasyfikacji binarnej.
  - Im bliżej 1, tym lepszy model.

### Przykłady Obliczeń

- **Accuracy:**
  - Procent poprawnie sklasyfikowanych próbek.
  - Nie zawsze odpowiednia dla niezrównoważonych danych.
- **Lift:**
  - Miara efektywności modelu względem losowej predykcji.
  - Popularna w marketingu.
- **ROC Curve:**
  - Wykres przedstawiający czułość (sensitivity) względem współczynnika fałszywie pozytywnych (FPR).
  - Pomocny w ocenie jakości klasyfikatora.

### Opis Pojęć

- **Confusion Matrix:** Tabela używana do oceny wydajności algorytmu klasyfikacji, zawierająca wartości True Positives, True Negatives, False Positives i False Negatives.
- **Precision (PPV):** Miara dokładności pozytywnych przewidywań modelu ( $TP / (TP + FP)$ ).
- **Recall (Sensitivity):** Miara zdolności modelu do wykrywania wszystkich pozytywnych przypadków ( $TP / (TP + FN)$ ).
- **F1 Score:** Harmoniczna średnia precision i recall, używana do równoważenia tych dwóch metryk.
- **AUC (Area Under the ROC Curve):** Miara jakości modelu klasyfikacji binarnej, reprezentująca wydajność modelu dla wszystkich możliwych progów.

- **Lift:** Miara efektywności modelu predykcyjnego względem losowej predykcji, szczególnie użyteczna w analizie marketingowej.
- **ROC Curve:** Wykres przedstawiający związek między czułością (sensitivity) a współczynnikiem fałszywie pozytywnych (FPR), używany do oceny wydajności klasyfikatora binarnego.

## Uczenie Bayesowskie

### Wprowadzenie do podejmowania decyzji

#### Metody wnioskowania (reasoning methodologies)

- **Logika binarna:** Tradycyjna logika prawdy/fałszu.
- **Logika rozmyta:** Obsługuje niepewne lub niejasne informacje.
- **Logika trójwartościowa:** Rozszerza logikę binarną o trzecią wartość (np. nieznane).
- **Teoria Dempstera-Shafera:** Alternatywa do teorii prawdopodobieństwa dla modelowania niepewności.
- **Wnioskowanie niemonotoniczne:** Wnioskowanie, które może zmieniać się w odpowiedzi na nowe informacje.

### Wnioskowanie

#### Wnioskowanie (inference)

- **Dedukcyjne:** Oparte na solidnych, absolutnych regułach wnioskowania (nigdy nie jest błędne).
  - Przykład: Jeśli wiemy, że Dave lubi ciasteczka, a Oreo to ciasteczka, to Dave lubi Oreo.
- **Indukcyjne:** Oparte na obserwacjach, mogą być prawdziwe lub fałszywe.
  - Przykład: Wiemy, że 82% testów z AI zawiera pytanie o Regułę Bayesa, więc zakładamy, że takie pytanie będzie na teście.

## *Podejmowanie racjonalnych decyzji przy niepewności*

### Teoria prawdopodobieństwa

- Reprezentacja wiedzy i niepewności.
- Optymalne aktualizowanie wiedzy na podstawie nowych informacji.
- Wnioskowanie.

### Teoria decyzji

- Połączenie teorii prawdopodobieństwa z teorią użyteczności.
- Problem: jak wykorzystać informacje, aby osiągnąć maksymalną oczekiwaną użyteczność.

## *Funkcja użyteczności*

### Funkcja użyteczności

- Określa preferencje dotyczące decyzji, które obejmują niepewność.
- Transformuje zysk na skalę odzwierciedlającą preferencje decydenta.
- Rodzaje:
  - **Unikanie ryzyka:** Funkcja wypukła (np.  $u(x) = \ln(x)$ )
  - **Poszukiwanie ryzyka:** Funkcja wklęsła (np.  $u(x) = x^2$ ).
  - **Neutralność wobec ryzyka:** Funkcja liniowa (np.  $u(x) = x$ ).

## Teoria Bayesa

### Twierdzenie Bayesa

- Umożliwia obliczenie prawdopodobieństwa warunkowego, zwanego prawdopodobieństwem posteriori, bez konieczności posiadania pełnych prawdopodobieństw wspólnych.
- $$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)}$$

## Kluczowe pojęcia

### Prawdopodobieństwo priorytetowe (prior probability)

- Prawdopodobieństwo niezależne od jakichkolwiek innych czynników.

### Prawdopodobieństwo warunkowe (conditional probability)

- Prawdopodobieństwo jednego (lub więcej) zdarzenia przy założeniu wystąpienia innego zdarzenia.

### Niezależność (independence)

- Znajomość wartości B nie mówi nam nic o wartości A i vice versa.

### Dystrybucja prawdopodobieństwa (probability distribution)

- Wszystkie wartości prawdopodobieństwa dla pewnej zmiennej.

### Reguła łańcuchowa (chain rule)

- $P(A \cap B) = P(A|B) * P(B)$ .

## Klasyfikatory probabilistyczne

### Klasyfikatory probabilistyczne

- Metody zwracające "prawdopodobieństwa"  $P(y/x)$  (modelują niepewność).
- Modele:
  - **Dyskryminacyjne (discriminative)**
  - **Generatywne (generative)**

## *Klasyfikatory Bayesowskie*

### MAP (Maximum A Posteriori Estimation)

- Znajduje najbardziej prawdopodobną hipotezę, biorąc pod uwagę dane treningowe.

### **NBC (Naive Bayes Classifier)**

- Zakłada, że cechy są warunkowo niezależne względem klasy.
- Powszechnie używany w klasyfikacji tekstu i problemach z dużą liczbą zmiennych.

### ***Bayesian Optimal Classifier (BOC)***

#### **Bayesian Optimal Classifier (BOC)**

- Znajduje najbardziej prawdopodobną klasyfikację dla nowych danych poprzez maksymalizowanie prawdopodobieństwa poprawnej klasyfikacji każdej obserwacji.
- Metoda maksymalnego prawdopodobieństwa posteriori (MAP) jest uproszczonym wariantem BOC, który wybiera najbardziej prawdopodobną hipotezę na podstawie danych treningowych.

### ***Optymalizacja Bayesowska***

#### **Optymalizacja Bayesowska (Bayesian optimization)**

- Metoda globalnej optymalizacji wykorzystująca twierdzenie Bayesa do znalezienia wejścia, które minimalizuje/maksymalizuje koszt funkcji celu.
- Proces budowania probabilistycznego modelu funkcji celu, znanego jako funkcja zastępcza, aby efektywnie przeszukiwać funkcję celu za pomocą funkcji akwizycji.

### ***Bayesian Belief Networks (BBN)***

#### **Bayesian Belief Networks (BBN)**

- Probabilistyczny model graficzny reprezentujący zmienne losowe oraz ich zależności.



- Umożliwia wizualizację modelu probabilistycznego, przeglądanie relacji oraz rozumowanie o prawdopodobieństwach przyczynowych na podstawie dostępnych dowodów.

### ***Minimum Description Length (MDL)***

#### **Minimum Description Length (MDL)**

- Kryterium wyboru hipotezy, która prowadzi do najlepszego skompresowania danych treningowych.
- Opiera się na zasadzie brzytwy Ockhama: wybieranie najprostszej hipotezy, która wyjaśnia dane.

### ***Naive Bayes Classifier (NBC)***

#### **Naive Bayes Classifier (NBC)**

- Zakłada, że cechy są warunkowo niezależne względem klasy (dlatego nazywany "naiwnym").
- Popularny w klasyfikacji tekstu i problemach z dużą liczbą zmiennych.
- Trening: Priorytety są oceniane przez liczenie liczby klas w danych, a  $p(x_j/y)$  jest oceniane jako częstość obserwacji wartości cechy  $x_j$  wśród wektorów klasy  $y$ .

### ***Wnioski***

#### **Praktyczne zastosowania Bayesowskich metod**

- Optymalizacja hipotez i klasyfikacji w kontekście uczenia maszynowego.
- Wykorzystanie sieci Bayesa do modelowania złożonych domen i podejmowania decyzji przy niepewności.
- Porównywanie różnych metod optymalizacji i klasyfikacji za pomocą teorii Bayesa.

### **Podsumowanie**

Metody uczenia maszynowego oparte na teorii Bayesa dostarczają solidnych ram do podejmowania decyzji przy niepewności. Teoria Bayesa

pozwała na efektywne aktualizowanie prawdopodobieństw w świetle nowych danych, co jest kluczowe w dynamicznie zmieniających się środowiskach. Narzędzia takie jak klasyfikatory Naive Bayes czy Bayesian Belief Networks znajdują szerokie zastosowanie w wielu dziedzinach, od diagnostyki medycznej po analizę ryzyka i prognozowanie finansowe.

## Regresja Logistyczna (Logistic Regression)

### Wprowadzenie do regresji logistycznej

#### Regresja logistyczna (logit regression):

- Używana do estymacji prawdopodobieństwa przynależności instancji do określonej klasy.
- Klasyfikator binarny: jeśli oszacowane prawdopodobieństwo jest większe niż 50%, model przewiduje przynależność do klasy pozytywnej (etykieta "1"), w przeciwnym razie do klasy negatywnej (etykieta "0").

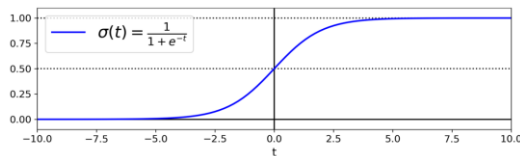
### Model regresji logistycznej

#### Krok 1: Regresja liniowa

- step1: regression
  - ✓  $\hat{t} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \dots + \theta_m x_m$
  - $\hat{t}$  predicted value
  - $x_j$  j-th feature value,  $x = [1, x_1, x_2, \dots, x_m]$
  - $m$  number of features
  - $\theta_j$  j-th model parameter  
( $\theta_0$  bias/intercept,  $\theta_1, \theta_2, \dots, \theta_m$  feature weights)

#### Krok 2: Funkcja logistyczna

- step 2: logistic regression
  - ✓ sigmoid function:  $\hat{p} = \sigma(\hat{t}) = \frac{1}{1+e^{-\hat{t}}}$
  - ✓ S-shaped
  - ✓ outputs a number between 0 and 1
- origin of the name
  - ✓ function:  $\hat{t} = \log\left(\frac{\hat{p}}{1-\hat{p}}\right) \equiv \text{logit}(\hat{p})$   
is the inverse of the logistic function
  - ✓  $t$  is called: logit, log-odds  
(since it is the log of the ratio between the estimated probability for the positive class and the estimated probability for the negative class)
- once the logistic regression model has estimated the probability  $\hat{p} = \sigma(\theta^T x)$  that an instance  $x$  belongs to the positive class, it can make prediction
 
$$\hat{y} = \begin{cases} 0 & \text{if } \hat{p} < 0.5 \\ 1 & \text{if } \hat{p} \geq 0.5 \end{cases} = \begin{cases} 0 & \text{if } \theta^T x < 0 \\ 1 & \text{if } \theta^T x \geq 0 \end{cases}$$



## Trening modelu

- **Cel:** Ustalenie wektora parametrów  $\theta$ , tak aby model estymował wysokie prawdopodobieństwa dla pozytywnych instancji ( $y=1$ ) i niskie dla negatywnych ( $y=0$ ).

## Funkcja kosztu

**Dla jednej instancji:**

- resulting cost function for a training instance  $x$ :

$$L^{(i)}(\theta) = \begin{cases} -\log(\hat{p}) & \text{if } y = 1 \\ -\log(1 - \hat{p}) & \text{if } y = 0 \end{cases}$$

- $\hat{p} \rightarrow 0: L^{(i)}(\theta) \rightarrow \infty$  large cost for positive instance
- $\hat{p} \rightarrow 1: L^{(i)}(\theta) \rightarrow 0$  small cost for positive instance
- $\hat{p} \rightarrow 1: L^{(i)}(\theta) \rightarrow \infty$  large cost for negative instance
- $\hat{p} \rightarrow 0: L^{(i)}(\theta) \rightarrow 0$  small cost for negative instance
- cost function over the whole training set  $\rightarrow$  log loss  
(average cost over all training instances)

$n$

**Dla całego zbioru treningowego (log loss):**

- cost function over the whole training set  $\rightarrow$  log loss  
(average cost over all training instances)

$$L(\theta) = -\frac{1}{n} \sum_{i=1}^n [y^{(i)} \log(\hat{p}^{(i)}) + (1 - y^{(i)}) \log(1 - \hat{p}^{(i)})]$$

### Rozwiązanie

Brak znanej zamkniętej formy równania do obliczenia wartości  $\theta$  minimalizującej funkcję kosztu.

### **Gradient Descent (GD):**

- Funkcja jest wypukła, więc można znaleźć globalne minimum.
- Pochodne cząstkowe funkcji kosztu względem  $j$ -tego parametru modelu  $\theta_j$ :

$$\frac{\partial}{\partial \theta_j} L(\theta) = \frac{1}{n} \sum_{i=1}^n (\sigma(\theta^T x^{(i)}) - y^{(i)}) x_j^{(i)}$$

Metody treningowe

- **Batch Gradient Descent (Batch GD)**
- **Stochastic Gradient Descent (SGD)**
- **Mini-batch Gradient Descent**

## Regularizacja

- **L1 Regularization**
- **L2 Regularization**

## Wprowadzenie do regresji softmax

### Regresja softmax (multinomial regression):

- Jest to rozszerzenie regresji logistycznej na przypadki, gdzie istnieje wiele klas (więcej niż dwie).
- Służy do klasyfikacji wieloklasowej, gdzie każda instancja może należeć tylko do jednej z wielu możliwych klas.

## Model regresji softmax

### Krok 1: Obliczanie wyników dla każdej klasy

- Dla instancji  $x$  obliczany jest wynik (score) dla każdej klasy  $k$ :

✓ for an instance  $x$  compute a score for each

class  $k$  (regression):  $s_k(x) = \theta^{(k)T}x$

$$= \theta_0^{(k)} + \theta_1^{(k)}x_1 + \theta_2^{(k)}x_2 + \dots + \theta_m^{(k)}x_m$$

$s_k(x)$  scores = logits = log-odds (although they are actually unnormalized log-odds)

$x_j$   $j$ -th feature value,  $j=0, 1, \dots, m$

$\theta_j^{(k)}$   $j$ -th model parameter for class  $k$

feature weights:

$(\theta_0^{(k)} \text{ bias/intercept}, \theta_1^{(k)}, \theta_2^{(k)}, \dots, \theta_m^{(k)})$

$\theta$  parameter matrix of parameter vectors  $\theta^{(k)}$

- ✓ apply the softmax function (=softmax, = normalized exponential)  $\hat{p}_k = \sigma(s_k(x))$

## Krok 2: Funkcja softmax

- Funkcja softmax normalizuje wyniki, przekształcając je w prawdopodobieństwa przynależności do klas:

- probability that the instance belongs to class k

$$\hat{p}_k = \sigma(s(x))_k = \frac{\exp(s_k(x))}{\sum_{l=1}^C \exp(s_l(x))}$$

- C number of classes
- s(x) vector containing the scores of each class for the instance x
- $\sigma(s(x))_k$  estimated probability that the instance x belongs to class k, given the scores of each class for that instance

## Predykcja

- Model regresji softmax przewiduje klasę z najwyższym oszacowanym prawdopodobieństwem:

$$\begin{aligned}\hat{y} &= \underset{k}{\operatorname{argmax}} \sigma(s(x))_k = \underset{k}{\operatorname{argmax}} s_k(x) \\ &= \underset{k}{\operatorname{argmax}} \left( (\theta^{(k)})^T x \right)\end{aligned}$$

- Model przewiduje tylko jedną klasę na raz (multiclass, nie multioutput), więc powinien być używany tylko z klasami wzajemnie wykluczającymi się (np. różne gatunki roślin).

## Funkcja kosztu

**Cel:** Estymować wysokie prawdopodobieństwo dla klasy docelowej i niskie dla innych klas.

**Funkcja kosztu (kategoryczna entropia krzyżowa):**

- Mierzy, jak dobrze zestaw oszacowanych prawdopodobieństw klas pasuje do klas docelowych:

$$L(\theta) = -\frac{1}{n} \sum_{i=1}^n \sum_{k=1}^C y_k^{(i)} \log(\hat{p}_k^{(i)})$$

$y_k^{(i)}$  target probability that the i-th instance belongs to class k (= 1 or 0, depending on whether the instance belongs to the class or not)

### Rozwiązanie

Brak znanej zamkniętej formy równania do obliczenia wartości  $\theta$ , które minimalizują funkcję kosztu.

#### Gradient Descent (GD):

- Gradient Descent (GD)
  - ✓ gradient vector of the cost function with regard to  $\theta^{(k)}$  i.e. for every class
 
$$\nabla_{\theta^{(k)}} L(\theta) = \frac{1}{n} \sum_{i=1}^n (\hat{p}_k^{(i)} - y^{(i),k}) \mathbf{x}^{(i)}$$
  - ✓ partial derivatives
 
$$\frac{\partial}{\partial \theta_j} L(\theta) = \frac{1}{n} \sum_{i=1}^n \sum_{k=1}^C (\hat{p}_k^{(i)} - y^{(i),k}) x_j^{(i)}$$
  - ✓ find the parameter matrix  $\theta$  that minimizes the cost function

### Wyjaśnienia kluczowych pojęć

**Regresja logistyczna:** Metoda estymacji prawdopodobieństwa przynależności instancji do określonej klasy, stosowana do binarnej klasyfikacji.

**Logit (log-odds):** Logarytm stosunku prawdopodobieństwa przynależności do klasy pozytywnej do prawdopodobieństwa przynależności do klasy negatywnej.

**Funkcja sigmoidalna:** Funkcja logistyczna, która przekształca wynik regresji liniowej na prawdopodobieństwo (wartość między 0 a 1).

**Gradient Descent (GD):** Algorytm optymalizacji używany do minimalizacji funkcji kosztu poprzez iteracyjne przesuwanie się w kierunku przeciwnym do gradientu funkcji kosztu.

**Funkcja kosztu (log loss):** Funkcja mierząca błąd modelu, gdzie duże koszty są przypisywane dużym błędom predykcji.

## PCA (Principal Component Analysis)

### Wprowadzenie do PCA

- PCA (Analiza Głównych Składowych) to metoda redukcji wymiarów poprzez przekształcenie zmiennych wejściowych w nowe, bardziej użyteczne zmienne zwane głównymi składowymi.
- Zastosowania: redukcja wymiarowości, poprawa cech, lepsza interpretacja danych.

### Redukcja wymiarów

- Przykłady redukcji: 2D do 1D, 3D do 2D.
- Znaczenie redukcji wymiarów w interpretacji danych i analizy statystycznej.

### Analiza czynnikowa

- Statystyczna metoda opisu zmienności obserwowanych cech za pomocą mniejszej liczby zmiennych ukrytych zwanych czynnikami.

### Główne składniki: wymagania

- Wektory ortogonalne (niezależne).
- Kombinacja liniowa zmiennych wejściowych.
- Składniki ułożone według malejącej wariancji.



## Interpretacja geometryczna

- Dane jako n-punkty w przestrzeni m-wymiarowej.
- Macierz kowariancji zawierająca relacje między punktami.
- Transformacja zmiennych oryginalnych na główne składniki przez rotację układu współrzędnych.

## Założenia PCA

- Dane powinny być jednorodne, bez outlierów.
- Standaryzacja danych jest obligatoryjna.
- Minimalna liczba próbek zależna od liczby cech.

## Kryteria wyboru liczby składników

- Kryterium wartości własnej (Kaisera).
- Kryterium wyjaśnionej wariancji.
- Kryterium wykresu Catella (scree plot).

## Algorytm PCA

- Metody oparte na macierzy kowariancji, macierzy korelacji, SVD.
- Transformacja danych do przestrzeni głównych składników i analiza w przestrzeni zredukowanej.

## Przykłady zastosowania PCA

- Analiza zestawu danych "Miasta": wyjaśnienie zmienności przy użyciu głównych składników.

## Podsumowanie

- PCA jako narzędzie do redukcji wymiarów, poprawy interpretacji danych i wyjaśnienia zmienności w danych.

## Kluczowe pojęcia

## Principal Component Analysis (PCA)

- PCA to technika redukcji wymiarów, która przekształca zmienne wejściowe w zestaw nieskorelowanych zmiennych zwanych głównymi składowymi, uporządkowanych według ich ważności.

### **Redukcja wymiarowości**

- Proces przekształcania danych o dużej liczbie wymiarów (zmiennych) do przestrzeni o mniej liczbie wymiarów, zachowując jak najwięcej informacji.

### **Analiza czynnikowa**

- Technika statystyczna używana do opisu zmienności obserwowanych zmiennych za pomocą mniejszej liczby zmiennych ukrytych (czynników).

### **Główne składniki**

- Nowe zmienne, powstałe z kombinacji liniowych zmiennych wejściowych, które maksymalizują wariancję danych. Pierwsza główna składowa wyjaśnia największą możliwą część zmienności danych, druga – kolejną największą i tak dalej.

### **Macierz kowariancji**

- Macierz zawierająca kowariancje pomiędzy wszystkimi parami zmiennych w danych. Jest używana do określenia korelacji pomiędzy zmiennymi i znalezienia głównych składowych.

### **Standaryzacja danych**

- Proces przekształcania danych, aby miały średnią 0 i odchylenie standardowe 1. Jest to kluczowe w PCA, aby wszystkie zmienne miały porównywalne wagi.

### **Kryterium wartości własnej (Kaisera)**

- Kryterium używane do wyboru liczby głównych składowych do zachowania. Sugeruje zachowanie tylko tych składowych, które mają wartości własne większe niż 1.

### Kryterium wyjaśnionej wariancji

- Kryterium wyboru liczby głównych składowych na podstawie sumy wariancji wyjaśnionej przez te składowe. Często używa się progu 80-95% wyjaśnionej wariancji.

### SVD (Singular Value Decomposition)

- Technika matematyczna używana w PCA do dekompozycji macierzy danych na macierz jednostkową, macierz wartości szczególnych i transpozycję macierzy jednostkowej. Umożliwia efektywne obliczenie głównych składowych.

### SVM (Support Vector Machine)

#### Wprowadzenie do SVM

- SVM (Support Vector Machine) to binarny, liniowy klasyfikator w wielowymiarowej przestrzeni cech generowanej przez funkcję jądra.
- Algorytm maksymalizuje margines między klasami.
- Kluczowe zastosowania: klasyfikacja, klasteryzacja, wykrywanie anomalii, ranking, czyszczenie danych.
- Obszary zastosowań: bioinformatyka, wizja komputerowa, kategoryzacja tekstów, rozpoznawanie pisma ręcznego, analiza szeregów czasowych.

#### Historia

- 1963: Wprowadzenie SVM przez Vapnika (liniowe SVM).

- Lata 90.: Rozszerzenie przez Bosera, Cortesa, Vapnika o metody jądra.
- 1995: Rozszerzenie na funkcję ciągłą (regresja)

### **Cel SVM**

- Znalezienie optymalnej hiperplany oddzielającej, która maksymalizuje margines między klasami.
- SVM jest problemem źle uwarunkowanym, ponieważ istnieje wiele możliwych hiperplanów oddzielających.

### **Dane i separacja liniowa**

## linear separator

- data set is linearly separable if there is a hyperplane  $(w, b)$  such that:

$$w \cdot x^{(i)} + b = 0$$

$$w \in \mathbb{R}^d$$

$$b \in \mathbb{R} \quad \text{offset vector}$$

- equation  $(w, b)$  is redundant (arbitrary)  $\rightarrow$  without loss of generality one can assume the canonical hyperplane:

$$\min_i |x^{(i)} \cdot w + b| = 1$$

- finally, for the two classes of points:

$$x^{(i)} \cdot w + b \geq +1 \text{ for } y^{(i)} = +1$$

$$x^{(i)} \cdot w + b \leq -1 \text{ for } y^{(i)} = -1$$

hence

$$y^{(i)}(w \cdot x^{(i)} + b) \geq 1 \\ i = 1, 2, \dots, l$$

( $l$ : number of support vectors)

### **problem: functional minimization ('hard margin')**

- searching for an optimal hyperplane  $w \cdot x + b = 0$   
means minimization of the functional  $\Phi = \frac{1}{2} w \cdot w$   
subject to:  $y^{(i)}(w \cdot x^{(i)} + b) \geq 1, i = 1, \dots, l$   
( $l$ : number of support vectors)
- functional  $\Phi$  does not depend on  $b$   
(if the equation for the separation hyperplane  $w \cdot x + b = 0$  is satisfied, a shift in the direction perpendicular to the hyperplane does not change the margin, but the hyperplane ceases to be optimal  
→ it approaches one of the classes)

### **Margines i wektory nośne**

- Problem optymalizacji funkcji celu za pomocą mnożników Lagrange'a, minimalizacja w przestrzeni wektorów.

- standard optimization technique:

$$L(w, b, \Lambda) = \frac{1}{2} w \cdot w - \sum_{i=1}^l \alpha^{(i)} [y^{(i)}(x^{(i)} \cdot w + b) - 1]$$

$\Lambda^T = (\alpha^{(1)}, \dots, \alpha^{(l)})$  vector of Lagrange multipliers  
 $\forall_i \alpha^{(i)} \geq 0$

it is a quadratic optimization problem subject to linear constraints:

- ✓ convexity
- ✓ unique minimum (no local minima)
- ✓ polynomial time complexity

### **Formulacja dualna i algorytm SVM**

### algorithm

- solve the system of  $(l+1)$  equations against:  $\alpha_i, b$ :

$$\sum_{i=1}^l \alpha^{(i)} y^{(i)} = 0$$

for all positive support vectors:

$$\sum_{i=1}^l \alpha^{(i)} y^{(i)} K(x^{(i)}, x^{(r+)}) + b = +1$$

for all negative support vectors:

$$\sum_{i=1}^l \alpha^{(i)} y^{(i)} K(x^{(i)}, x^{(s-)}) + b = -1$$

$$\alpha^{(i)} \geq 0, \quad i = 1, \dots, l, \quad r + s = l$$

$l$ : number of support vectors

linear, separable case:  $K(x^{(i)}, x^{(j)}) = x^{(i)} \cdot x^{(j)}$

if there is no degeneration in the system of  $(l+1)$  equations:

- calculate coefficients:

$$w = \sum_{i=1}^l \alpha^{(i)} y^{(i)} x^{(i)}$$

- calculate separation hyperplane:

$$wx + b = 0$$

- calculate margin:

$$m = \frac{2}{\|w\|} = \frac{2}{\sqrt{\sum_i w_i^2}}$$

- 
- Problem optymalizacji kwadratowej z ograniczeniami liniowymi.

### Nieliniowa separacja

- Przejście do wyższej przestrzeni wymiarowej za pomocą funkcji transformacji (**patrz poniżej**).
- Trik jądrowy: (**patrz poniżej**)

## concept

- transition to higher dimensional space where the data will be separable
- transformation function  $\Phi: \mathbf{R}^d \mapsto \mathcal{H}$   
 $\mathbf{x}^{(i)} \cdot \mathbf{x}^{(j)} \rightarrow \Phi(\mathbf{x}^{(i)}) \cdot \Phi(\mathbf{x}^{(j)})$
- kernel trick: since in the SVM algorithm only the product  $\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$  is calculated, no explicit form of  $\Phi$  is required  $\rightarrow$   
$$\Phi(\mathbf{x}^{(i)}) \cdot \Phi(\mathbf{x}^{(j)}) = K(\mathbf{x}^{(i)} \cdot \mathbf{x}^{(j)})$$

## Funkcje jądra

- Typy: Liniowe, wielomianowe, radialne funkcje bazowe (RBF), sigmoidalne.
- Wybór funkcji jądra wpływa na decyzję granicy i efektywność klasyfikacji.

## Zadania optymalizacyjne SVM

- Klasyfikacja wektorów nośnych (C-SVC, v-SVC).
- Detekcja jednowartościowa (one-class SVM).
- Regresja wektorów nośnych ( $\epsilon$ -SVR, v-SVR).

## Straty SVM

- Hinge Loss: funkcja strat dla maksymalnej marginesowej klasyfikacji.

## Kluczowe Pojęcia

## Support Vector Machine (SVM)



- SVM to algorytm nadzorowanego uczenia maszynowego używany głównie do klasyfikacji, ale także do regresji. Jego celem jest znalezienie optymalnej **hiperplany (ang. hyperplan)** oddzielającej klasy w przestrzeni cech.

### Hiperplana oddzielająca

- Jest to hiperplana, która oddziela różne klasy w danych. W przypadku dwuwymiarowym jest to linia, w przypadku trójwymiarowym – płaszczyzna, a w wyższych wymiarach – hiperplana.

### Margines

- Margines to odległość między hiperplaną oddzielającą a najbliższymi punktami danych obu klas (wektorami nośnymi). SVM maksymalizuje ten margines, aby zapewnić jak największą separację między klasami.

### Wektory nośne

- Są to punkty danych, które leżą najbliżej hiperplany oddzielającej i mają decydujące znaczenie w wyznaczaniu tej hiperplany. Tylko te punkty wpływają na kształt i położenie hiperplany.

### Trik jądrowy

- Trik jądrowy pozwala SVM działać w przestrzeniach wielowymiarowych bez eksplicytnego przekształcania danych do tych przestrzeni.

### Funkcje jądra

- Liniowe
- Wielomianowe
- Radialne funkcje bazowe (RBF)
- Sigmoidalne

### Hinge Loss

- Funkcja strat używana w SVM. Mierzy, jak dobrze klasyfikator oddziela dane. Strata wynosi 0, jeśli punkt jest poprawnie sklasyfikowany i znajduje się poza marginesem, a rośnie liniowo, jeśli punkt znajduje się wewnątrz marginesu lub jest błędnie sklasyfikowany.

### **C-SVC, v-SVC, $\epsilon$ -SVR, v-SVR**

- Różne warianty SVM używane do klasyfikacji (C-SVC, v-SVC) i regresji ( $\epsilon$ -SVR, v-SVR), różniące się sposobem radzenia sobie z marginesami, błędami i regularyzacją.

## Drzewa decyzyjne

### Struktura drzewa decyzyjnego

#### **Podstawowe elementy:**

- **Drzewo decyzyjne:** skierowany graf acykliczny. Istnieje tylko jedna ścieżka pomiędzy dwoma węzłami.
- **Korzeń:** bez rodzica.
- **Węzeł:** reprezentuje atrybut.
- **Liść:** węzeł decyzyjny, reprezentuje klasę decyzji.
- **Krawędź:** gałąź, odpowiada wartości atrybutu.

### Atrybuty

#### **Typy atrybutów:**

- Binarne, nominalne, numeryczne/porządkowe, ciągłe.
- Drzewa decyzyjne efektywniej działają z atrybutami dyskretnymi.

### Możliwe porządki rozwijania węzłów

#### **W głąb (in depth)**

- Dla każdej gałęzi bieżącego węzła konstruowane jest kompletne poddrzewo, aż do osiągnięcia liścia, zanim zostanie wzięta pod uwagę kolejna gałąź.
- Algorytmy rekurencyjne są oparte na tej metodzie.

### **Wszerz (in width)**

- Dla każdej gałęzi bieżącego węzła tworzona jest tylko jedna warstwa węzłów potomnych lub liści na raz.

### **Najpierw najlepszy (best first)**

- Na każdym poziomie rozważana jest tylko jedna nierozwinięta gałąź, wybrana na podstawie funkcji jakości.

### **Kryteria podziału w węzłach**

#### **Kryteria podziału w węzłach:**

- **Heurystyka:** preferencja atrybutów, które prowadzą do najczystszych pod-węzłów.
- **Podział dla atrybutów numerycznych/ciągłych:** wartość progowa do podziału na dwie części.

### **Główne problemy**

#### **Główne problemy:**

- Wybór atrybutu początkowego.
- Kiedy przestać rozbudowywać drzewo (nadmierne dopasowanie, trudność analizy drzewa).
- Co zrobić z atrybutami o zbyt wielu wartościach?
- Wpływ sprzecznych/zakłóconych/danych brakujących na rozwój drzewa.

### **Algorytmy i metryki**

#### **Algorytmy i metryki:**

- **ID3**: entropia, przyrost informacji.
- **C4.5**: stosunek zysku.
- **CART**: Gini.
- **CHAID**:  $\chi^2$ .

### **decision trees as a method of ML**

- supervised
- nonparametric (i.e. have no assumptions about the space distribution and the classifier structure)
- not prescriptive (i.e. do not assume a normal distribution)
- non-linear classifiers
- batch learning: trees process all training instances at once rather (incremental learning would require rebuilding the whole tree with all the former data)
- do not require any domain knowledge
- effective learning both with small training data and large amount of data (data mining, exploratory knowledge discovery)

- both for classification and regression
- decision trees are invariant under all monotone transformations of individual ordered variables (due to splitting nodes by thresholding)
- are relatively robust to outliers and misclassified points (they do not calculate any mean from the data points)
- time for building a tree may be higher than another type of classifier, testing/classification is fast
- a decision tree has a hierarchical structure of rules/questions about the attributes (from the root down to the leaves)
- inductive bias: "prefer shortest (minimal depth) tree"
- local optimization
- creating an exactly minimal decision tree is a difficult problem

15

### Proste drzewa

- Mają większe szanse na odkrycie interesujących informacji.
- Mają niższą wariancję i wyższy błąd: nadmierne dopasowanie nie jest takie łatwe.
- Dzielą przestrzeń cech na kilka dużych grup: lepszą generalizacją.

### Złożone drzewa

- Mogą pasować do danych przypadkowo.
- Mają niższy błąd i wyższą wariancję: łatwo o nadmierne dopasowanie.
- Dzielą przestrzeń cech na wiele małych grup (nawet oddzielna grupa dla każdej próbki treningowej): gorszą generalizacją.

## how to assess the quality of the decision tree

- by size: the lower the better
  - ✓ a small number of nodes
  - ✓ low height
  - ✓ few leaves
- by classification accuracy
  - ✓ on train set
  - ✓ on test set
- example
$$Q(T) = \alpha \cdot \text{size}(T) + \beta \cdot \text{accuracy}(T, D)$$

$\alpha, \beta$ : real numbers  
Q: classification quality, T: tree, D: data

### Zalety i wady

#### Zalety drzew decyzyjnych:

- Możliwość obsługi danych nieliniowych.
- Brak potrzeby specjalnego przetwarzania danych.
- Dobry w wykrywaniu ważnych zmiennych.

#### Wady drzew decyzyjnych:

- Nie wszystkie koncepcje mogą być reprezentowane za pomocą drzew decyzyjnych.
- Możliwość generowania dużych drzew.
- Niska stabilność w obliczu małych zmian w danych treningowych.

### Źródła szumu

#### Źródła szumu:

- Dane: różne instancje tego samego wektora cech mają różne klasy, brakujące wartości, nieodpowiednie instancje.

- Atrybuty: nieodpowiedni zestaw atrybutów, duża liczba atrybutów, wartości ciągłe.
- Struktura: założenie o podziałach jednego atrybutu, założenie o podziale przestrzeni na wzajemnie wykluczające się regiony.
- Algorytm: metoda "dziel i zwyciężaj", nadwrażliwość na nieistotne atrybuty i szum, niewłaściwa metryka.

### Kiedy używać drzew decyzyjnych

- Gdy potrzebna jest zarówno symboliczna reprezentacja, jak i (stosunkowo) dobra wydajność klasyfikacji.
- Problem nie zależy od wielu atrybutów.
- Skromny podzbiór atrybutów zawiera istotne informacje.
- Liniowe kombinacje cech nie są krytyczne.
- Ważna jest szybkość uczenia.

### **Drzewo decyzyjne -> Reguły decyzyjne:**

- Każde drzewo decyzyjne można zamienić na równoważny zestaw reguł.
- Tworzona jest jedna reguła dla każdej ścieżki od korzenia do liścia.
- Liczba liści jest równa liczbie reguł.
- Zestaw reguł może nie zawsze być optymalny.

### **Zestaw reguł -> Drzewo decyzyjne:**

- Jest mniej trywialne.
- Nie każdy zestaw reguł ma równoważne drzewo.
- Wiele koncepcji ma krótszy opis jako zestaw reguł.

### Typy atrybutów

#### **Typy atrybutów:**

- Kategoryczne/nominalne.

- Numeryczne/porządkowe.
- Ciągłe.

### Kategoryczne/nominalne atrybuty

#### **Kategoryczne/nominalne atrybuty:**

- Standardowe cechy dla algorytmów drzew decyzyjnych.
- Testy tożsamości są używane.
- Drzewo ma gałęzie zgodnie ze wszystkimi różnymi wartościami danej cechy.

### Atrybuty numeryczne/ciągłe

#### **Atrybuty numeryczne/ciągłe:**

- Nie mogą być traktowane jako wartości kategoryczne.
- Metryki mogą faworyzować atrybuty z dużą liczbą wartości.

### Procedura dla atrybutów numerycznych/ciągłych

#### **Procedura dla atrybutów numerycznych/ciągłych:**

- Muszą być podzielone na najbardziej optymalne grupy kategoryczne.
- Należy znaleźć optymalną wartość progową do podziału.

### Klasy testów

#### **Klasy testów:**

- Testy operujące na pojedynczym atrybucie.
- Testy operujące na kombinacji atrybutów.

### Typy testów

#### **Typy testów:**

- Testy tożsamości dla atrybutów nominalnych.
- Testy nierówności dla atrybutów numerycznych/ciągłych.



## Brakujące wartości atrybutów

### Brakujące wartości atrybutów:

- Przypisywanie nowej wartości atrybutu oznaczonej jako: unknown, ?, - , N/A.
- Przypisywanie najbardziej powszechnej wartości atrybutu.
- Przypisywanie średniej ważonej wartości atrybutu.

## Metryki

### Metryki:

- Kryteria optymalizacji drzewa.
- Funkcje zanieczyszczenia.
- Miary oparte na entropii, statystykach, stopniu wariacji danych.

## Przeuczenie (Overfitting)

### Przeuczenie:

- Może wystąpić, gdy zestaw danych zawiera szum lub nie jest reprezentatywną próbą danych.
- Charakteryzuje się małym błędem na danych treningowych i dużym błędem na danych testowych.
- Podstawowy algorytm drzewa decyzyjnego:
  - Rozwija drzewo wystarczająco głęboko, aby poprawnie klasyfikować przykłady treningowe.
  - Jest odpowiedni dla idealnych danych i deskryptywnego podejścia do odkrywania wiedzy.

### Jak unikać przeuczenia i radzić sobie z szumem w danych?

- Mniejsze drzewa mają niższe ryzyko przeuczenia.
- Zatrzymywanie:
  - Przestań rozwijać drzewo wcześniej, zanim osiągnie punkt, w którym idealnie klasyfikuje dane treningowe.

- Trudne.
- Definiowane przez niektóre interpretywalne hiperparametry.
- Przycinanie:
  - Przycinanie w przód: zatrzymaj podział węzłów.
  - Przycinanie wstecz: utwórz drzewo, które (nadmiernie) dopasowuje się do danych treningowych, a następnie uprość je.
  - Przekształć drzewo w reguły decyzyjne i zastosuj przycinanie reguł (C4.5).

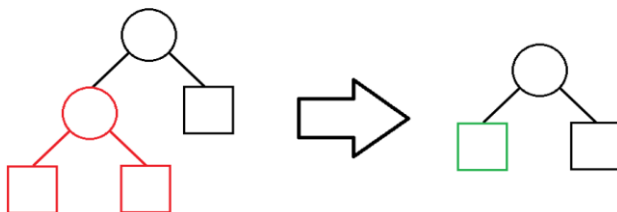
## Przycinanie (Pruning)

### Dlaczego potrzebujemy przycinania?

- Metody rozwijania drzewa są zachłanne.
  - Zły lub brak podziału na jednym etapie może prowadzić do bardzo dobrych podziałów w przyszłości.
  - Patrzenie naprzód na wiele kroków nie rozwiąże fundamentalnie tego problemu.
- Strategia 'rozwijanie drzewa z kryterium zatrzymania' (zwykle) nie jest zadowalająca.

#### what to do?

- order of pruning: bottom up  
(allows you to apply mild changes)
- remove branches that contain only a few instances
- evaluate optimal tree complexity on validation set



### Konsekwencje przycinania:

- Mniejsze drzewo z mniejszą liczbą podziałów może prowadzić do niższej wariancji modelu i nieco wyższego błędu.
- Mniejsze drzewa mają mniejsze ryzyko przeuczenia.
- Mniejsze drzewa mogą lepiej generalizować (prowadzić do lepszej interpretacji danych).

### Rodzaje przycinania:

- Przycinanie w przód.
- Przycinanie wstecz.
- Metody mieszane: kombinacja obu.

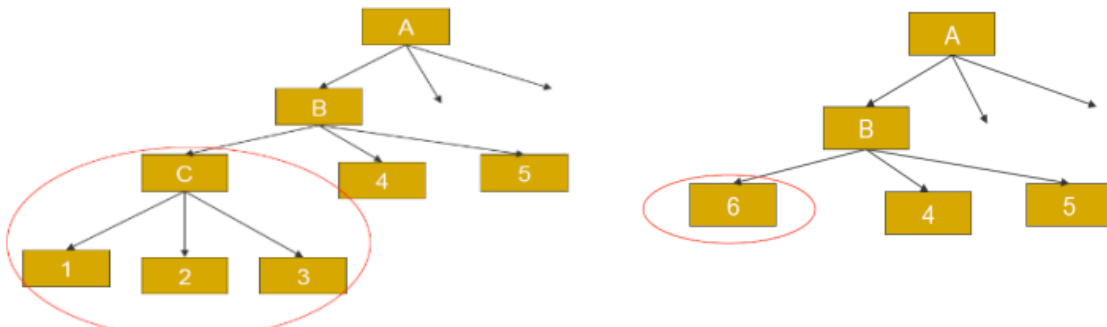
### post-pruning techniques

- subtree replacement
- subtree raising

94

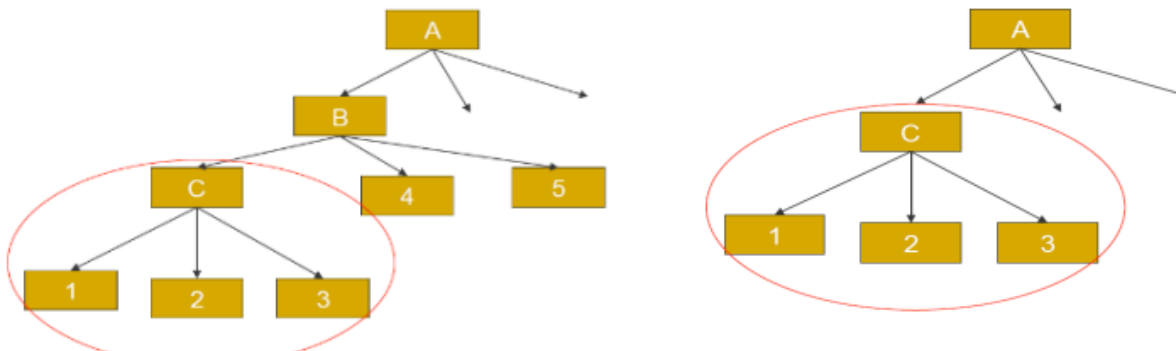
### subtree replacement

- entire subtree is replaced by a single leaf node
- label leaf with the majority class of the remaining data
- bottom-up strategy (check the possibility of replacing the subtree only when the trees below have already been checked)
- simple and often used



## subtree raising

- a node is replaced with one of its child nodes (entire subtree is raised onto another node) = delete the node (in the middle) and split the instances into the subtrees below
- requires the reorganization of the subtree structure
- it gives the most optimal results
- very complex and time consuming
- slower than subtree replacement



## Drzewa decyzyjne obojętne (Oblivious Decision Trees)

### Drzewa decyzyjne obojętne:

- Drzewa decyzyjne, dla których wszystkie węzły na tym samym poziomie testują tę samą cechę.
- Minimalizują ogólny podzbiór atrybutów wejściowych (co prowadzi do redukcji wymiarów).
- Są skuteczne w selekcji cech.
- Zwykle budowane przez algorytm zachłanny, który stara się maksymalizować miarę wzajemnej informacji na każdym poziomie.
- Przeszukiwanie rekurencyjne kończy się, gdy nie ma atrybutu, który wyjaśniałby cel ze statystyczną istotnością.

## Fuzzy Decision Trees

### Fuzzy Decision Trees:

- Mogą jednocześnie przypisywać więcej niż jedną gałąź tej samej instancji z pewnością stopniową.
- Zachowują symboliczną strukturę drzewa i jego zrozumiałość.
- Mogą reprezentować koncepcje o stopniowanych cechach, produkując wyjścia wartości rzeczywistych z stopniowymi przejściami.
- Mogą radzić sobie z szumem lub niepewnościami w danych zbieranych w systemach przemysłowych.

### Drzewa decyzyjne oparte na algorytmach ewolucyjnych

#### Drzewa decyzyjne oparte na algorytmach ewolucyjnych:

- Rozwijanie zestawu drzew (z różnymi parametrami) do rozwiązania problemu.
- Wymiana "genomu" (fragmentów drzew) między indywidualnymi modelami.
- Mutacja:
  - Losowe zastąpienie niektórych węzłów w drzewie.
  - Losowa zmiana kryterium podziału w węźle.
  - Losowe rozwijanie części drzewa (kilka kolejnych poziomów decyzji).
- Krzyżowanie:
  - Jednopunktowe, dwupunktowe, jednolite.
  - Bardziej dokładne drzewa przekazują swoje fragmenty mniej dokładnym drzewom.

## Ensembling Methods:

- Komitety drzew:
  - Homogeniczny zestaw drzew decyzyjnych analizujących dany problem pod różnymi podejściami.
  - Końcowa decyzja wynika z:
    - (ważonego) uśredniania.
    - Większościowego/ważonego głosowania.
    - Dodatkowego algorytmu.
- Boosting:
  - Iteracyjnie poprawia model klasyfikacji.
  - Początkowo instancje danych mają przypisane równe wagi, następnie przypadki trudne do klasyfikacji otrzymują wyższe wagi, algorytm koncentruje się bardziej na tych przypadkach.
- Bagging (bootstrap aggregation):
  - Generowanie wielu replik bootstrap na losowych (z zamianą) małych podzbiorach danych.
  - Model testowany na próbkach out-of-bag (OOB) (instancje nieużywane w uczeniu modelu).

## Kompromis obciążenie - wariancja

### *Podstawowe formalizmy*

#### Założenia:

- **X**: przestrzeń wejściowa, elementy  $x$ .
- **Y**: przestrzeń wyjściowa, elementy  $y$ .
- **Z**: proces stochastyczny nad  $X * Y$ , definiowany przez nieznaną wspólną zmienną losową  $p(x, y)$ .
- Zbiór danych treningowych:  $\{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)\}$ , który zawiera  $N$  próbek z  $Z$  (zawiera szum).

#### Model predykcyjny ( $f$ ):

- $f(x, \alpha)$ , gdzie  $x$  to dane, a  $\alpha$  to parametry modelu.
- Model uczący się na podstawie zbioru treningowego.
- Przybliża prawdziwą funkcję  $y$ .

### Koszt predykcji ( $L(y, z)$ ):

Funkcja kosztu predykcji  $z$ , gdy prawdziwa wartość to  $y$ .

Typy funkcji kosztu:

- **Kwadratowy koszt (squared loss):**  $L(y, z) = (y - z)^2$ .
- **Bezwzględny koszt (absolute loss):**  $L(y, z) = |y - z|$ .
- **Koszt zero-jedynkowy (zero-one loss):**  $L(y, z) = 0$ , gdy  $y = z$ ;  $L(y, z) = 1$ , gdy  $y \neq z$ .

### Rozkład błędu

#### Rozkład błędu:

- Analiza błędu generalizacji jako suma trzech składników:
  - Bias (obciążenie).
  - Variance (wariancja).
  - Irreducible error (nieredukowalny błąd).

#### Obciążenie (Bias):

- Różnica między oczekiwaną predykcją modelu a prawdziwą wartością:  $\text{Bias}[z] = (E[y] - E[z])^2$ .
- Model z wysokim obciążeniem może nie uchwycić istotnych zależności (underfitting).

#### models with low bias

- may represent a large noise component in the training set, making predictions less accurate (overfitting)
- predicted data points are close to the target
- suggest less assumptions about the form of the target function
- are usually more complex (e.g. higher-order regression polynomials)
- a zero-bias approach has poor generalization to new situations

### **models with high bias**

- can miss the relevant relations between features and target outputs (underfitting)
- predicted data points are far from the target
- suggest more assumptions about the form of the target function
- typically produce simpler models (low-order or even linear regression polynomials)
- pay very little attention to the training data - always lead to a high error on training and test data

### **Wariancja (Variance):**

- Zmienność predykcji modelu dla różnych zbiorów treningowych:  
 $\text{Var}[z] = E [(z - E[z])^2]$ .
- Model z wysoką wariancją może nadmiernie dopasować się do danych treningowych (overfitting).

### **models with high variance**

- pay a lot of attention to training data and can model noise, so does not generalize well on unseen data (overfitting)
- suggest large changes to the estimate of the target function with changes to the training dataset

### **models with low variance**

- data points are close to each other as a result close to the function
- the model suggests small changes to the estimate of the target function with changes to the training dataset

### **Nieredukowalny błąd (Irreducible Error):**



- Szum w danych, którego nie można zredukować przez model:  
 $\text{Var}[y] = E [(y - E[y])^2]$ .

## Redukcja wariancji

### Metody redukcji wariancji:

- Przycinanie drzew decyzyjnych (**pruning**).
- Wczesne zatrzymanie (**early stopping**).
- Regularizacja (**regularization**): Lasso, Ridge.
- Metody zbiorcze (**bagging, random forest**).

## Regresja a klasyfikacja

### Regresja (loss kwadratowy):

- $E [(y - z)^2] = \text{Var}[y] + \text{Bias}[z]^2 + \text{Var}[z]$ .

### Klasyfikacja (loss zero-jedynkowy):

- Misclassification risk:  $r(y') = E [L(f, y)]$ .
- Prawdopodobieństwo błędu klasyfikacji:  $P(z \neq z_m)$ .

## Metryki oceny modeli

### Metryki:

- $R^2$ , adjusted  $R^2$ .
- AIC, BIC.
- Hold-out set.
- Cross-validation (n-fold CV).
- Leave-One-Out Cross-Validation (LOO CV).

## Logika zdań (CNF/DNF)

## Wprowadzenie do logiki

### Definicja logiki:

- Etymologia: z greckiego "logikos" - dotyczący rozumowania.
- Nauka o rozumowaniu, dowodzeniu, myśleniu lub wnioskowaniu.
- Systematyczne badanie formy argumentów.

### Historia:

- **Syllogizm Arystotelesa i Stoików:** dedukowanie wniosków z ogólnych i szczegółowych stwierdzeń.
- **Logika zdań:** Peter Abelard (XII wiek).
- **Logika symboliczna:** Gottfried Leibnitz, George Boole, August De Morgan.
- **Logika predykatów:** Gottlob Frege.
- **Dedukcja naturalna:** Gerhard Gentzen, Jan Łukasiewicz.
- **Drzewa prawdy:** Evert Willem Beth.
- **Tablice prawdy:** Ludvig Wittgenstein, Emil Post.

## Zastosowania logiki

### Zastosowania logiki:

- Formułowanie obserwacji.
- Definiowanie pojęć.
- Formalizacja teorii i dowodzenie twierdzeń.
- Wyprowadzanie wniosków z istniejących informacji.

### Zastosowania logiki w informatyce:

- Tworzenie języków formalnego rozumowania (np. PROLOG).
- Rozwój systemów ekspertowych (np. MYCIN, MISTRAL).
- Obliczanie złożoności obliczeniowej (np. NP-zupełność).
- Efektywne wyszukiwanie i ocenianie zapytań w bazach danych.
- Formalna weryfikacja poprawności sprzętu i oprogramowania.
- Poszukiwanie hipotez w uczeniu maszynowym.

## Rodzaje logiki

### Rodzaje logiki:

- **Logika zdań (propositional logic):** fakty, wartości true/false.
- **Logika predykatów (predicate logic):** fakty, obiekty, relacje, wartości true/false/unknown.
- **Logika temporalna (temporal logic):** fakty, obiekty, relacje, czasy, wartości true/false/unknown.
- **Logika rozmyta (fuzzy logic):** stopień prawdziwości w przedziale  $[0, 1]$ , wartości znane jako interwały.

## Problemy w zastosowaniu logiki

### Problemy w zastosowaniu logiki:

- Eksplozja kombinatoryczna procedury dowodowej.
- Bariera Goedla: formalne systemy zawierające arytmetykę nie mogą mieć wszystkich trzech właściwości: kompletności, spójności i efektywnej aksjomatyzacji.
- Problem ramy: trudność wnioskowania o zmianach w świecie.
- Trudność łączenia metod formalnych z dostępną heurystyczną informacją.
- Wykorzystanie niekompletnych i niepewnych informacji.
- Proces rozumowania człowieka różni się od tradycyjnej logiki monotonicznej.

## Definicje podstawowe

### Kalkulator logiczny (calculus):

- System formalny składający się z dobrze uformowanych formuł, aksjomatów i zestawu reguł formalnych.

### System logiczny (logical system):

- System formalny, w którym wyrażenia to zdania, a reguły są prawdziwościowe (zachowujące prawdę) i mogą zawierać aksjomaty.

## Logika zdań (propositional logic)

### Logika zdań:

- Najbardziej abstrakcyjna, używana do manipulowania prostymi, deklaratywnymi zdaniami.
- Operuje na zdaniach boolowskich o wartościach prawda/fałsz.
- Składnia jest niezależna od kontekstu, dopuszcza częściowe/zanegowane informacje.

### Ograniczenia logiki zdań:

- Bardzo ograniczona moc wyrazu: tylko proste fakty.
- Brak właściwości lub relacji do modelowania rzeczywistych obiektów.

## Składnia logiki zdań

### Składnia:

- Legalne formy/zdania języka.

### Symbole języka logiki zdań:

- Formuły atomowe (litery zdań/propozycje): zestaw prymitywnych symboli.
- Operatory logiczne (spójniki): zestaw symboli operatorów.
- Nawiasy "(", ")": używane do grupowania wyrażeń.

### Propozycja (proposition):

- Deklaratywne zdanie, które może być prawdziwe lub fałszywe (nie może być oba jednocześnie).
- Może być zdaniem atomowym lub złożonym.

### Propozycje atomowe:

- Reprezentowane przez pojedyncze zmienne propozycyjne.

- Nie można ich logicznie rozbić na mniejsze zdania, muszą być prawdziwe lub fałszywe.

### **Propozycje złożone:**

- Powstają przez łączenie propozycji atomowych za pomocą spójników logicznych.
- Mogą być zagnieżdżone, a ich wartość prawdziwości można obliczyć na podstawie wartości prawdziwości propozycji atomowych.

### **Dobrze uformowana formuła (WFF):**

- Może być zbudowana z formuł atomowych za pomocą symboli operatorów zgodnie z regułami gramatyki.
- Każda dobrze uformowana formuła ma unikalne znaczenie i jest albo prawdziwa, albo fałszywa.

### **Spójniki logiczne**

#### **Spójniki logiczne:**

- Łączą propozycje w celu tworzenia bardziej złożonych propozycji.
- Typy spójników: negacja (NOT), koniunkcja (AND), alternatywa (OR), implikacja (IF-THEN), równoważność (IFF).

### **Semantyka logiki zdań**

#### **Semantyka:**

- Określa znaczenie zdania, przypisując wartości prawdziwości {T, F} zmiennym propozycjonalnym.
- Zasady semantyczne są oparte na składni i stosowane rekurencyjnie.

#### **Reguły semantyczne:**

- Używane do określenia prawdziwości zdania w danej interpretacji.
- Notacja:  $\text{holds}(p, i)$  oznacza, że zdanie  $p$  jest prawdziwe w interpretacji  $i$ .

## Prawa logiczne i transformacje

### Identyczności logiczne:

- Komutatywność, asocjatywność, rozdzielność, idempotencja, podwójna negacja, prawa De Morgana, uproszczenie, implikacja, kontrapozycja, równoważność, wyłączenie środka, sprzeczność.

### Konwersja do CNF (Conjunctive Normal Form):

- Eliminacja implikacji i równoważności, redukcja zakresu negacji, konwersja do koniunkcji alternatyw przy użyciu praw asocjatywnych i rozdzielnych.

### Konwersja do DNF (Disjunctive Normal Form):

- Eliminacja implikacji i równoważności, redukcja zakresu negacji, konwersja do alternatywy koniunkcji przy użyciu praw asocjatywnych i rozdzielnych.

## Dowodzenie w logice zdań

### Dedukcja naturalna (Natural Deduction):

- System dowodzenia, który zaczyna się od zbioru przesłanek i kończy konkluzją.
- Właściwości: poprawność (soundness) i kompletność (completeness).

### Reguły wnioskowania:

- Algorytmy niezależne od dziedziny, opisujące, jak można wyprowadzić nową wiedzę z istniejącej bazy wiedzy.
- Typy reguł: Modus Ponens, Modus Tollens, eliminacja koniunkcji, wprowadzenie koniunkcji, eliminacja alternatywy, wprowadzenie alternatywy, wprowadzenie negacji, eliminacja podwójnej negacji, wprowadzenie sprzeczności, eliminacja sprzeczności.

### Dowodzenie twierdzeń (Theorem Proving):

- Proces udowadniania prawdziwości formuł na podstawie zbioru aksjomatów i reguł wnioskowania.
- **Drzewa dowodzenia:**
  - Struktura hierarchiczna używana do przedstawienia dowodu.
  - Korzeń drzewa to twierdzenie, które chcemy udowodnić.
  - Węzły wewnętrzne reprezentują reguły wnioskowania.
  - Liście reprezentują aksjomaty lub twierdzenia, które już zostały udowodnione.

### **Metoda rezolucji (Resolution):**

- Technika dowodzenia twierdzeń w logice zdań.
- Opiera się na regule rezolucji, która pozwala na łączenie dwóch klauzul, aby usunąć zmienną.
- **Algorytm rezolucji:**
  - Konwersja wszystkich formuł do postaci koniunkcyjno-normalnej (CNF).
  - Iteracyjne stosowanie reguły rezolucji w celu uproszczenia formuł.
  - Dowód kończy się sukcesem, jeśli uzyskamy pustą klauzulę, co oznacza sprzeczność.

## **Analiza Dyskryminacyjna (Discriminant Analysis)**

### **Fisher's Linear Discriminant (FLD)**

- **Cel:** Redukcja wymiarowości przy zachowaniu jak największej ilości informacji dyskryminacyjnej między klasami.
- **Metoda:** Znalezienie wektora projekcji, który maksymalizuje różnicę między średnimi klas, znormalizowaną przez miarę rozproszenia wewnątrzklasowego.

### **Linear Discriminant Analysis (LDA)**

- **Założenia:** Funkcje gęstości prawdopodobieństwa w klasach są normalnymi rozkładami wielowymiarowymi  $N(\mu_c, \Sigma)$ , z równymi macierzami kowariancji wewnątrzklasowych  $\Sigma$ . **(patrz poniżej)**
- ✓ probability density functions in the classes are d-dim normal distributions  $N(\mu_c, \Sigma)$
- ✓ equal within-class covariance matrices across classes  $\Sigma_1 = \Sigma_2 \equiv \Sigma$
- **Cel:** Maksymalizacja różnicy między średnimi klas, przy jednoczesnej minimalizacji rozproszenia wewnątrzklasowego.
- **Decomposition:** Całkowita macierz kowariancji  $S$  jest sumą macierzy rozproszenia wewnątrzklasowego  $W$  i międzyklasowego  $B$ .

### Quadratic Discriminant Analysis (QDA)

- **Cel:** Klasyfikacja z różnymi macierzami kowariancji dla każdej klasy.
- **Metoda:** Użycie powierzchni kwadratowych do oddzielenia klas.
  - **Funkcja dyskryminacyjna:** Wykorzystuje dystans Mahalanobisa, uwzględniając różne macierze kowariancji dla klas.

### LDA vs. QDA

- common practice: instead of using QDA apply LDA to an extended set of predictors  $\rightarrow$  it works similarly to QDA  

$$(x_1, \dots, x_m, x_1x_2, \dots, x_{m-1}x_m, x_1^2, \dots, x_m^2)$$
- despite their simplicity, LDA and QDA often work better than many more sophisticated learning paradigms

### Macierz kowariancji (Covariance Matrix)



- **Definicja:** Miara tego, jak bardzo dwie zmienne zmieniają się razem.

### Analiza wariancji (ANOVA)

- **Cel:** Użycie zmiennych niezależnych kategorycznych i ciągłych zmiennych zależnych.
- **Różnica z LDA:** ANOVA używa zmiennych kategorycznych jako zmienne niezależne, podczas gdy LDA używa ciągłych zmiennych niezależnych i kategorycznych zmiennych zależnych.

### Analiza czynnikowa (Factor Analysis)

- **Cel:** Znalezienie liniowych kombinacji cech, które najlepiej wyjaśniają dane.
- **Różnica z LDA:** Analiza czynnikowa bazuje na korelacjach cech (bez etykiet), podczas gdy LDA używa etykiet i modeluje klasy.

### Naive Bayes Classifier (NBC)

- **Cel:** Klasyfikacja oparta na regule Bayesa, zakładająca niezależność atrybutów.

### Transformacje LDA

- **Zależne od klasy:** Maksymalizacja stosunku wariancji międzyklasowej do wariancji wewnątrzklasowej dla każdej klasy.
- **Niezależne od klasy:** Maksymalizacja stosunku całkowitej wariancji do wariancji wewnątrzklasowej, używając jednego kryterium optymalizacji.

### *Najważniejsze punkty:*

- **Fisher's Linear Discriminant:** Metoda liniowej dyskryminacji opracowana przez R.A. Fishera, która znajduje linię projekcji maksymalizującą separację między klasami.

- **Linear Discriminant Analysis (LDA):** Technika statystyczna używana do znalezienia liniowej kombinacji cech, która najlepiej oddziela dwie lub więcej klas.
- **Quadratic Discriminant Analysis (QDA):** Uogólnienie LDA, które pozwała na różne macierze kowariancji dla każdej klasy.
- **Kowariancja wewnątrzklasowa i międzyklasowa:** Kluczowe elementy LDA, używane do maksymalizacji separacji klas.
- **Analiza wariancji (ANOVA) vs. LDA:** ANOVA skupia się na zmiennych niezależnych kategoriycznych, podczas gdy LDA na zmiennych ciągłych.
- **Naive Bayes Classifier (NBC):** Prosty, ale skuteczny klasyfikator oparty na regule Bayesa, zakładający niezależność atrybutów.
- **Transformacje LDA:** Metody zależne i niezależne od klasy używane do maksymalizacji separacji klas.