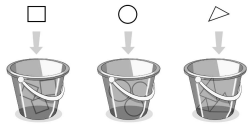
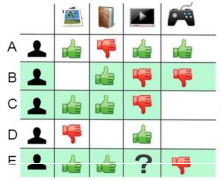


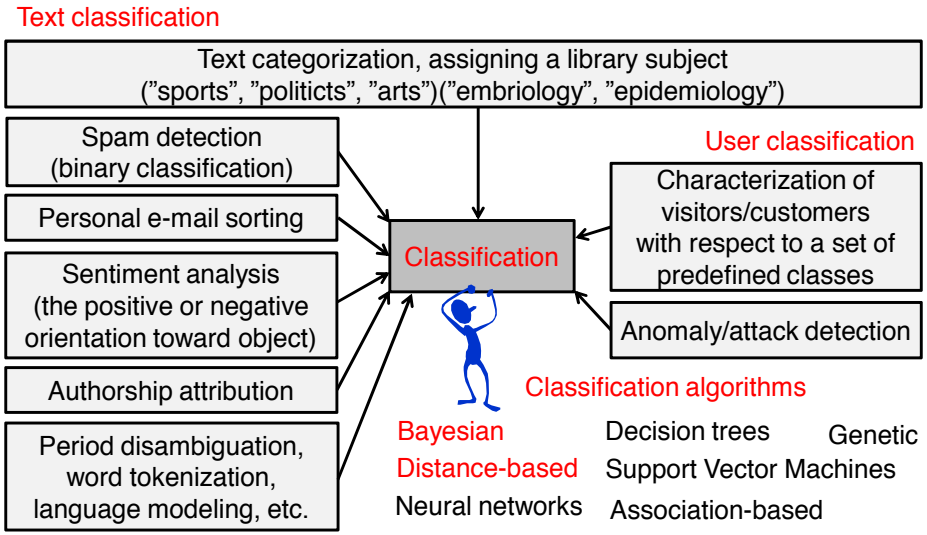


Web Content and Usage Analysis: Classification and Collaborative Filtering

Miłosz Kadziński
Institute of Computing Science
Poznan University of Technology, Poland
www.cs.put.poznan.pl/mkadzinski/wpi

[1] Wykład będzie łączył aspekty typowe dla analizy zawartości oraz użytkowania sieci. Skupimy się na podstawowych zagadnieniach eksploracji danych w zakresie klasyfikacji oraz predykcji. Jeśli chodzi o klasyfikację, omówimy dwie sławne grupy algorytmów: najbliższych sąsiadów oraz Bayesowski. Nie będziemy poświęcać im nadmiernej uwagi, ponieważ mam świadomość, że zadanie klasyfikacji było lub będzie omawiane też na wielu innych przedmiotach. W związku z tym, większa część wykładu zostanie poświęcona predykcji oraz społecznemu filtrowaniu. Co istotne, zrozumienie dedykowanych algorytmów jest tu o wiele łatwiejsze, jeśli w pierwszej kolejności zapoznać się z zasadą działania podstawowych procedur klasyfikujących. Omówimy cztery grupy algorytmów predykcji i zobaczycie, że albo pomysł jest tu bardzo podobny, jak w metodach klasyfikacji, albo poszczególne kroki są z nich zaczerpnięte.

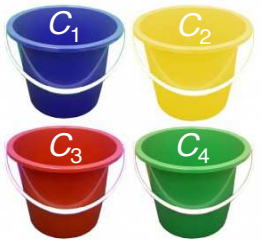
Real-world Examples of Classification Tasks



[2] Rozpoczniemy od klasyfikacji. Aby udowodnić, jak istotne jest to zagadnienie w zakresie przetwarzania informacji i eksploracji zakresów Internetu, podamy przykładowe praktyczne zastosowania w kontekście analizy tekstów i użytkowników. Standardowy przykład dotyczy dokumentów odnosi się do sytuacji, w której zadaniem jest przypisanie tzw. tematu bibliotecznego. Klasami mogłyby tu być choćby: sport, polityka, sztuka, rozrywka, biznes, technologie czy styl życia. W innych zastosowaniach klasy mogą być bardzo specyficzne. Tak jest choćby przy identyfikacji spamu, gdzie zadanie ma charakter binarny; każda wiadomość albo jest spamem albo nie. W definicji klas dużą rolę może pełnić użytkownik. Najlepszym przykładem jest tu sortowanie wiadomości e-mail. Charakterystyka klas może być jednak jeszcze zupełnie inna. Choćby w przypadku analizy nastawienia, celem jest wykrycie, czy użytkownik miał pozytywny czy negatywny stosunek do opisywanego obiektu; klasami mogą być też poszczególni autorzy, a zadanie polega na ich automatycznej identyfikacji; w przetwarzaniu tekstów problemów, które można sformułować w ramach klasyfikacji jest bez liku. Jeśli chodzi o użytkowników, to najczęściej klasyfikacja wykorzystywana jest bądź do tego, by określić, do jakiej predefiniowanej grupy należy klient sklepu albo do wykrycia pewnych anomalii, np. do wskazania, czy klient jest rzeczywisty czy nie. Istnieje bardzo wiele grup metod klasyfikacji, w tym klasycznych jak drzewa decyzyjne lub maszyna wektorów nośnych lub tych, opartych na wiodących trendach sztucznej inteligencji, jak algorytmy ewolucyjne lub sieci neuronowe. Podczas tego wykładu skupimy się jednak na dwóch podstawowych grupach, tj. algorytmach probabilistycznych (Bayesowskich) oraz bazujących na podobieństwie lub odległości między obiektami.

Classification Task

Given a **description of an instance**, $x \in X$, in terms of features or attributes in space X *and* a **fixed set of class** or category labels $C = \{C_1, C_2, \ldots, C_n\}$, a **classification task** is to determine the class/category of x: $c(x) \in C$, where $c(x)$ is a function whose domain is X and whose range is C



- *Organize and categorize data in distinct classes*
- Create model and use it classify new data (i.e., predict a discrete and nominal value (category))

[3] Ogólne sformułowanie zadania klasyfikacji jest następujące: niech będzie dany opis zbioru instancji (obiektów) na zbiorze cech lub atrybutów oraz zbiór predefiniowanych klas, kategorii lub etykiet; zadanie polega na przypisaniu każdej instancji do jednej klasy lub ich podzbioru z wykorzystaniem funkcji lub modelu, którego dziedziną jest zbiór instancji, a przeciwdziedziną - zbiór klas. Mówiąc krótko, będziemy chcieli zorganizować nasze dane i przydzielić je do różnych, z góry określonych klas. Z praktycznego punktu widzenia - będziemy chcieli stworzyć model, który będzie można wykorzystać do klasyfikacji nowych danych, tj. predykcji dyskretnej i nominalnej wartości nazywanej klasą, kategorią lub etykietą.

Classification: Three Steps

Model construction (learning)

- Each instance (example) is assumed to belong to a pre-defined class, as determined by one of the attributes called a **target attribute**
- The values of the target attribute are the **class labels**
- The set of all instances used for learning the model is called **training set**

Model evaluation (accuracy)

- **Estimate accuracy** rate of the model based on a **test set**
- The known labels of test instances are compared with the predicted class
- Test set is independent of training set (otherwise over-fitting will occur)

Model use (classification)

- The model is used to **classify unseen instances** (i.e., to predict the class labels for new unclassified instances)
- Predict the value of an actual attribute

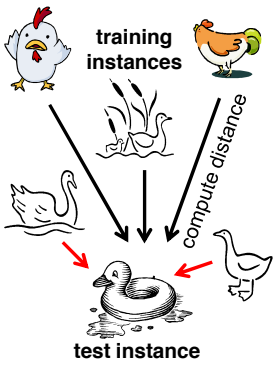
[4] W dużej mierze klasyfikację możemy postrzegać jako pracę z modelem. Ta praca podzielona jest na trzy etapy: uczenie, ocenę jakości oraz wykorzystanie. Proces uczenia polega na konstrukcji modelu na podstawie znanej klasyfikacji dla zbioru uczącego. Zbiór ten charakteryzuje się tym, że wartość atrybutu docelowego dla przykładów uczących jest znana, co oznacza, że etykieta klasy dla takich przykładów jest zdefiniowana i stanowi dla nas punkt odniesienia, który chcielibyśmy w uczeniu odtworzyć. Etap ewaluacji zakłada ocenę jakości działania modelu na zbiorze testowym, który nie był wykorzystywany podczas uczenia. Jest to bardzo ważne, bo w przeciwnym razie, tj. jeśli algorytm widziałby dane testowe już podczas uczenia mógłby do nich dopasować model, który w efekcie straciłby zdolności generalizacji. Formalnie nazywamy to efektem przeuczenia. Aby ocenić jakość modelu, potrzebujemy dobrze zdefiniowanych miar, które oddawałyby stopień zgodności między tym, co powinien przewidzieć a co przewiduje model dla danych testowych. Jedną z częściej wykorzystywanych miar jest tu jakość klasyfikacji. Wreszcie w ostatnim etapie już nauczony model, które dokładność została uprzednio zbadana, wykorzystujemy do klasyfikacji danych, dla których nominalna wartość atrybutu decyzyjnego jest nieznana i to właśnie model ma ją określić.

Distance-based Classification

- Classify new instances based on their similarity (distance) from instances we have seen before
- Sometimes called "**instance-based learning**"

Basic idea

- Save all previously encountered instances
- Given a new instance, find those instances that are *most similar* to the new one
- Assign new instance to the same class as these "nearest neighbors"

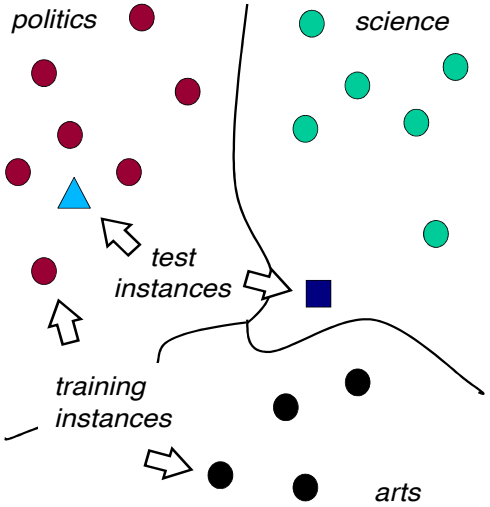


Lazy classifiers

- The approach defers all of the real work until new instance is obtained; no attempt is made to learn a generalized model from the training set
- Less data preprocessing and model evaluation, but more work has to be done at classification time

[5] Pierwsza grupa algorytmów klasyfikacji, którą omówimy, to metody bazujące na odległości lub podobieństwie. Ogólna zasada ich działania zakłada, że klasyfikacja nowych przykładów powinna opierać się na podobieństwie do przykładów, które już widziano, dla których algorytm zna klasę. Niekiedy takie metody nazywa się bazującymi na przykładach (instancjach). Jeśli myśleć o algorytmach klasyfikacji, to zazwyczaj pierwsze wyobrażenia odnoszą się do tego, że proces konstrukcji modelu musi być bardzo złożony. Nie w tym wypadku. Zadanie tu polega tylko na przechowywaniu przykładów, dla których klasyfikacja jest znana i odłożeniu całej pracy na później, tj. do momentu, w którym pojawiają się przykłady, które trzeba zaklasyfikować. Dopiero wtedy oblicza się podobieństwo takiego przykładu do przykładów uczących i identyfikuje te, które są najbardziej podobne. Jest to proces bardzo intuicyjny; tacy najbliżsi sąsiedzi są po prostu najlepszymi przykładami, od których można się uczyć. Z tego powodu, klasyfikujemy nowy przykład (instancję) do tej samej klasy, jak jego najbliżsi sąsiedzi. W kontekście ich zasady działania, algorytmu takie reprezentują uczenie leniwe, które nie robią nic na etapie uczenia, ale za to ich wykonanie jest bardzo kosztowne na etapie wykorzystania.

Rocchio Classification

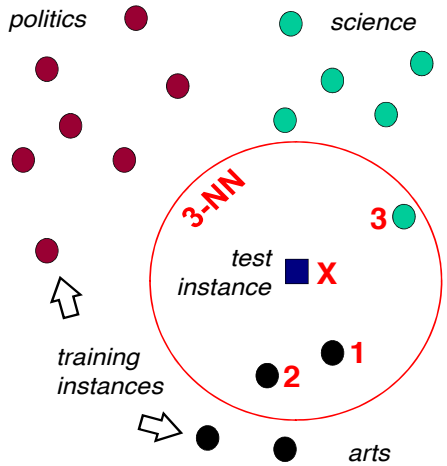


Basic Idea

- How to find separators?
- Separators equally distanced from the class centroids
- Assign instance to the class of training sample whose centroid is closest to it
- **Nearest centroid classifier**
- Similar to the Rocchio algorithms for relevance feedback
- No guarantee to reproduce classification for training instances

[6] Rozważmy przykład, w którym symbole na slajdzie reprezentują dokumenty, które w rzeczywistości są po prostu wektorami liczb (jak w reprezentacji Bag-of-words czy TF-IDF). Dysponujemy dokumentami z trzech klas: polityka, nauka oraz sztuka. Kilkanaście z nich stanowi zbiór uczący, tj. klasyfikacja jest dla nich znana, ale dwa - reprezentowane przez kwadrat i trójkąt - są dokumentami, dla którym etykieta jest nieznana. Pierwszy, prosty algorytm bazujący na odległości, który omówimy nazywa się klasyfikatorem Rocchio. Kryjący się za nim pomysł polega na narysowaniu granic między klasami. W szczególności, zakłada, że te granice są położone w równej odległości od centroidów klas. Dla przypomnienia, centroid to średnia reprezentacja obiektów z danej klasy. Granice narysowana na slajdzie. Dokument zostanie przypisany do tej klas, w której obszar wpada. Formalnie oznacza to, że klasę determinuje odległość od centroidów poszczególnych klas. Ta metoda ma dwie podstawowe wady. Po pierwsze, nie gwarantuje, że klasy dla wszystkich przykładów uczących zostaną odtworzone. Nie dzieje się to na prostym przykładzie na slajdzie, ale rzeczywistość jest bardziej skomplikowana i czasami klasy nie są tak dobrze separowalne. Po drugie, na poziomie bardziej intuicyjnym, jeśli spojrzymy na klasyfikację obiektu-trójkąta, to wydaje się ona w porządku, ponieważ jest on położony blisko obiektów z tej klasy. Ale jeśli przyjrzeć się kwadratowi, dla którego przewidywana klasa to nauka, to wydaje się to bardziej kontrowersyjne. Jest on bowiem położony dość daleko od dokumentów z tej klasy, a bliżej dokumentów dotyczących sztuki. Ta ostatnia obserwacja sugeruje nowe rozwiązanie.

k-Nearest Neighbor Classifier



Model

- Given instance X find its ***k* nearest neighbors**
- Variety of distance or similarity measures can be used (* *later*)
- Requires comparison of X with all training instances

Classification

- Find the class label for each of the *k* neighbors
- Use a voting approach to determine the majority class among the neighbors
- Assign the majority class to X

k is often taken to be odd in order to avoid ties (works in case of two classes)
if there are more than two classes, take k to be the number of classes plus one

[7] Zamiast analizowania odległości od centroidów, sensowniejszym pomysłem jest obliczenie dla danego przykładu jego odległości od wszystkich obiektów uczących. Potem powinnyśmy wyłuskać te, które są najbliższe lub najbardziej podobne. Załóżmy, że ostatecznej analizie poddana zostanie ograniczona liczba, k najbliższych sąsiadów. Na przykładzie na slajdzie k jest równe 3. Następnie, analizujemy klasy tych przykładów i wykorzystujemy pewien mechanizm głosowania, by określić klasę z najwyższym wsparciem. Taka klasa większościowa jest sugerowanym wynikiem dla przykładu, dla którego chcemy określić klasę. Ta idea leży u podstaw algorytmu k najbliższych sąsiadów, po angielsku k nearest neighbors lub w skrócie k-NN, gdzie k oznacza rozmiar sąsiedztwa. Ważne pytanie dotyczy ustalenia wartości k. Domyślne reguły mówią, że k nie powinno być równe 1, bo dane uczące zwykle nie są doskonałe i w takim wypadku propagowalibyśmy błędy. Nie powinno być też zbyt wysokie, bo wtedy zatracimy ideą najbliższych sąsiadów i zaczniemy wykorzystywać w klasyfikacji przykłady, które nie są już tak bardzo podobne. W związku z tym, często przyjmuje się, że k jest małą nieparzystą liczbą, jak 3 lub 5, albo gdy klas jest więcej niż dwie, to k powinno być równe liczbie klas powiększonej o 1. W praktyce taki rozmiar sąsiedztwa jest meta-parametrem algorytmu i często podlega uczeniu, tj. testuje się różne wartości i wybiera tą, które daje najlepsze wyniki we wstępnych testach.

Combination Functions

Once the nearest neighbors are identified, the "**votes**" of these neighbors must be **combined to generate a prediction**

- Regular voting (democracy): each neighbor has a single vote

  A: 1 (D1) + 1 (D2) = **2** **B**: 1 (D3) + 1 (D4) + 1 (D5) = **3** ⇨ conf. level **B (60%)**

Use 5-NN	**D1**	**D2**	**D3**	**D4**	**D5**	D6	D7	D8	D9	D10		**X**
Class	**A**	**A**	**B**	**B**	**B**	A	B	B	A	A		**?**
Similarity with X	**0.9**	**0.8**	**0.6**	**0.5**	**0.5**	0.4	0.4	0.2	0.1	0.0		**1.0**

- Closer neighbors get higher weights (proportional to the similarity)
- The weighted sum for each class gives the comibined score for that class

  A: 0.9 (D1) + 0.8 (D2) = **1.7** **B**: 0.6 (D3) + 0.5 (D4) + 0.5 (D5) = **1.6** ⇨ **A**
- Introduces enough variation to prevent ties in most cases

** if there is strong variation in results for different k, this an indication that the training set is not large enough*

[8] Po identyfikacji najbliższych sąsiadów i ich klas, wciąż pozostaje kwestia, jak zrealizować głosowanie. Do tego celu istnieją dwie podstawowe procedury. W pierwszej, każdy najbliższy sąsiad ma równą moc głosu (załóżmy 1). Załóżmy więc, że mamy 10 przykładów uczących, X jest dokumentem, który chcemy zaklasyfikować, do czego użyjemy algorytmu 5-NN (tj. najbliższych sąsiadów z K=5). Dokumenty od D1 do D5 są najbliższymi sąsiadami dla X. Dwa z nich wskazują na klasę A, a pozostałe trzy na klasę B. W efekcie B byłoby klasą rekomendowaną dla X jako klasa większościowa. Można jednak zauważyć, że dokumenty D1 i D2, które należą do klasy A, są bardziej podobne do X niż D3, D4 i D5. W drugim podejściu, zamiast założenia równych mocy głosów, przyjmujemy, że siły głosów zależą od miar podobieństwa, tj. bliżsi sąsiedzi dostają większe wagi, wyższą moc. W naszym przykładzie klasyfikacja dla X uległaby zmianie, ponieważ wsparcie dla klasy A będzie większe niż dla klasy B. Z praktycznego punktu widzenia, to podejście jest lepsze, bo pozwala w zdecydowanej większości przypadków uniknąć remisów. Dużo trudniej bowiem uzyskać te same wyniki, sumując liczby z przedziału od 0 do 1 niż sumując tylko same 0 i 1.

Bayesian Learning

Bayes's theorem plays a critical role in probabilistic learning and classification

- **Uses prior probability of each class** given no information about an item
- Classification produces **a posterior probability distribution over the possible classes given a description of an item**
- The models are incremental (each training example can incrementally increase or decrease the probability that a hypothesis is correct)
- Prior knowledge can be combined with observed data

- A *data instance X* with an unknown class label
- Hypothesis *H* that *X belongs to a specific class C*
- The *conditional probability* of hypothesis *H* given *observation X,* P*(HIX),* follows the **Bayes's theorem**:

$$P(H|X) = \frac{P(X|H) \cdot P(H)}{P(X)}$$

- Practical difficulty: requires initial knowledge of many probabilities

[9] Drugi klasyfikator, który omówimy, jest reprezentatywny dla uczenia probabilistycznego w oparciu o schemat uczenia Bayesowskiego. Nazwa sugeruje, że opiera on się na słynnym twierdzeniu Bayesa, co oznacza że będziemy estymować rozkład prawdopodobieństwa na zbiorze klas, dysponując opisem przykładu, który ma zostać zaklasyfikowany. Rozważmy przykład X z nieznaną etykietą klasy. Naszym zadaniem jest obliczenie prawdopodobieństwa hipotezy, że X należy do określonej klasy. Formalnie, będzie to prawdopodobieństwo warunkowe tej hipotezy przy danym obiekcie X. W związku z tym, że nie może ono być policzone bezpośrednio, skorzystamy z twierdzenia Bayesa. Mówi ono, że takie prawdopodobieństwo można obliczyć jako iloczyn prawdopodobieństwa warunkowego wystąpienia obiektu X przy danej hipotezie H oraz prawdopodobieństwa wystąpienia hipotezy H podzielonego przez prawdopodobieństwo wystąpienia przykładu X. Nawet jeśli to twierdzenie pozwoli nam uprościć obliczenia, wciąż wiele prawdopodobieństw składowych musi być znanych i/lub obliczonych.

Bayesian Classification (1)

- Let set of classes be $\{C_1, C_2, \ldots, C_n\}$
- Let X be description of an instance (e.g., vector representation)
- Determine class of X by computing for each class C_i

$$P(C_i|X) = \frac{P(X|C_i) \cdot P(C_i)}{P(X)}$$

- **Select C_i for which $P(C_i|X)$ is maximal**

- P(X) can be determined since classes are complete and disjoint:

$$\sum_{i=1,\ldots,n} P(C_i|X) = \sum_{i=1,\ldots,n} \frac{P(X|C_i) \cdot P(C_i)}{\boldsymbol{P(X)}} = 1$$

$$\mathbf{P(X)} = \sum_{i=1,\ldots,n} P(X|C_i) \cdot P(C_i)$$

- In practice, P(X) can be neglected (the same for $P(C_i|X)$, i=1,…,n):

$$P(C_1|X) = \frac{P(X|C_1) \cdot P(C_1)}{P(X)} \quad \cdots \quad P(C_n|X) = \frac{P(X|C_n) \cdot P(C_n)}{P(X)}$$

[10] Sformalizujmy nasze zadanie klasyfikacji. Rozważamy obiekt X, który de facto jest wektorem liczb oraz zbiór klas. Musimy określić prawdopodobieństwo warunkowe dla każdej z n klas, dysponując opisem obiektu. W związku z tym, że stosujemy klasyfikator probabilistyczny, wybierze on klasę, dla której odpowiednie prawdopodobieństwo warunkowe będzie największe. Ich obliczenie nie jest jednak zadaniem łatwym, ponieważ musimy znać prawdopodobieństwa wystąpienia X przy danej klasie oraz prawdopodobieństwa zarówno dla X, jak i klasy. W pierwszej kolejności skupimy się na prawdopodobieństwie wystąpienia obiektu X, które znajduje się w mianowniku. W ogólności, skoro zbiór klas jest zupełny i są one rozłączne, dałoby się policzyć prawdopodobieństwo dla X. Nie jest to jednak konieczne. Ostateczne zadanie bowiem polega na obliczeniu prawdopodobieństwa warunkowego dla wszystkich klas, a prawdopodobieństwa dla X znajduje się w mianowniku każdego z takich prawdopodobieństw. Skoro chcemy wskazać klasę, dla której prawdopodobieństwa jest najwyższe, to dzielenie przez taką samą liczbę (jakakolwiek by ona nie była), nie zmieni względnego porównania wyników. Pomińmy zatem mianownik

Bayesian Classification (2)

$$P(C_i|X) \approx P(X|C_i) \cdot P(C_i)$$

- $P(C_i)$ are easily estimated from data
- N_i of the N training examples are in C_i

⇨ $P(C_i) = N_i / N$

Problem: too many possible combinations (exponential in *m* being the number of features/atttributes) to estimate all $P(X|C_i)$
Assume instance is a **conjunction of binary features/attributes**:

	t_1	t_2	…	t_m
X	1	1	…	0

⇨ $X = t_1 \wedge t_2 \wedge \ldots \wedge t_m$

⇨ $X = (t_1=1) \wedge (t_2=1) \wedge \ldots \wedge (t_m=0)$

If we assume features/attributes of an instance are **independent** given the class (C_i) (*conditionally independent*)

$$P(X|C_i) = P(t_1 \wedge t_2 \wedge \ldots \wedge t_m | C_i) = \prod_{j=1,\ldots,m} P(t_j|C_i)$$

- We need to know $P(t_j|C_i)$ for each feature and category
- Estimated based on observed frequencies in training data
- If N_{ij} of N_i examples in class C_i contain attribute t_j: $P(t_j|C_i) = N_{ij} / N_i$

[11] Obliczenie prawdopodobieństw dla klas jest także łatwe. Wystarczy przeanalizować cały zbiór uczący i sprawdzić, ile przykładów (instancji) w nim należy do danej klasy. Niestety, obliczenia dla ostatniego typu prawdopodobieństwa są już bardziej skomplikowane. Dokładne obliczenie prawdopodobieństwa wystąpienia obiektu X przy danej klasie Ci wymagałoby, aby X było obecne w dokładniej takiej postaci w przykładach uczących tej klasy, a aby dostać różne wyniki dla różnych klas, to także wśród przykładów uczących dla innych klas, być może w różnej liczbie kopii. Nie brzmi to realistycznie, więc problem należy uprościć. W pierwszej kolejności załóżmy, że obiekt jest koniunkcją pewnych cech, w tym wypadku binarnych atrybutów. Jeśli założyć, że cechy te są niezależne (bardzo istotne i silne założenie!), to możliwa jest dekompozycja prawdopodobieństwa warunkowego jako iloczynu prawdopodobieństw warunków dla wszystkich atrybutów oddzielnie, pojawiających się w dokładnie takiej postaci jak w klasyfikowanym obiekcie, w obiektach klasy Ci. Dla dokumentu widocznego na slajdzie, oznacza to, że dla termów t1 i t2 jesteśmy zainteresowani prawdopodobieństwem ich wystąpienia w dokumentach uczących z klasy Ci, a dla termu tm interesować nas będzie prawdopodobieństwo jego niewystąpienia, dokładnie jak w dokumencie X. Musimy znać te prawdopodobieństwa dla każdej cechy (termu) i klasy, ale szczęśliwie można je obliczyć stosunkowo prosto. Wystarczy bowiem iloraz między liczbą dokumentów w klasie Ci zawierających tę cechę w dokładniej tej formie, jak występuje ona w X, oraz liczby wszystkich dokumentów w klasie Ci. Dzięki temu będziemy wiedzieli, na ile typowa jest ta cecha (w formie występującej w X) dla dokumentów tej klasy.

Bayesian Classification (3) - Spam Example

	t_1	t_2	t_3	t_4	SPAM
D1	1	1	1	0	no
D2	0	1	0	0	no
D3	1	1	1	0	no
D4	1	0	0	1	no
D5	1	0	1	1	yes
D6	0	1	1	1	yes

Training data

P(no) = 4/6 P(yes) = 2/6

Probabilities

	P(t|no)	P(t|yes)
t_1	3/4	1/2
t_2	3/4	1/2
t_3	2/4	2/2
t_4	1/4	2/2

New e-mail containing t_3 and t_4

	t_1	t_2	t_3	t_4
X	0	0	1	1

Should it be classified as spam = "yes" or spam = "no"?

$P(C_i|X) \approx P(C_i) \cdot P(t_1=0|C_i) \cdot P(t_2=0|C_i) \cdot P(t_3=1|C_i) \cdot P(t_4=1|C_i)$

P(no|X) ≈ 4/6 · (1-3/4) · (1-3/4) · 2/4 · 1/4 = **0.0052**

P(yes|X) ≈ 2/6 · (1-1/2) · (1-1/2) · 2/2 · 2/2 = **0.0833**

$P(\text{yes}|X) > P(\text{no}|X)$ ⇨ *spam*

[12] Rozważmy przykład, który wiele wyjaśni. Dysponujemy zbiorem 6 e-maili (dokumentów) uczących, które opisywane są w kontekście 4 termów. Przykładowo, D1 zawiera termy t1, t2 i t3, ale nie zawiera t4. Nasze zadanie to konstrukcja filtru spamowego, więc klasyfikacja jest binarna z klasami: spam (yes-spam) lub nie-spam (no-spam). Zadanie zostanie przeprowadzone dla instancji X, która zwiera termy t1 i t2, ale nie zawiera t3 i t4. Celem jest obliczenie prawdopodobieństw warunkowych dla klas no-spam i yes-spam przy opisie instancji X. W tym celu potrzebujemy prawdopodobieństwa tych klas oraz prawdopodobieństwa warunkowe wszystkich termów w formie, w jakiej pojawiają się lub nie w X w e-mailach uczących dla różnych klas. Jeśli chodzi o prawdopodobieństwa klas, to 4 z 6 e-maili uczących należą do klasy no-spam, a 2 pozostałe z 6 do yes-spam. W tym kontekście, szanse, że losowa wiadomość nie jest spamem są 2 razy wyższe. Przejdźmy teraz do reprezentacji cech X w przykładach należących do poszczególnych klas. Przykładowo, jakie jest prawdopodobieństwo wystąpienia termu t3 w klasie no-spam? Istnieją 4 dokumenty w tej klasie, ale tylko dwa z nich zawierają t3. Na podobnej zasadzie, prawdopodobieństwo niewystąpienie termu t1 w klasie no-spam, to (1-3/4), czyli 1/4, bo w 1 na 4 dokumenty no-spam term t1 nie występuje. Aby obliczyć ostateczne wyniki, musimy pomnożyć wiele prawdopodobieństw, tj. prawdopodobieństwa poszczególnych klas oraz prawdopodobieństwa warunkowego niewystąpienia t1, niewystąpienia t2, wystąpienia t3 i wystąpienia t4 w danej klasie. Analiza ostatecznych wyników sugeruje, że prawdopodobieństwo dla klasy yes-spam jest nieco wyższe, więc bardziej prawdopodobne jest, że X jest spamem, co jest ostateczną rekomendacja klasyfikatora.

Bayesian Classification (4) - Smoothing

Estimating probabilities from small training sets is error-prone:

- If due only to chance, a rare feature, t_k, is always false in the training data, $\forall C_i : P(t_k|C_i) = 0$
- If t_k then occurs in a test example, X, the result is that $\forall C_i: P(X|C_i) = 0$ and $\forall C_i : P(C_i|X) = 0$

- To account for estimation from small samples, probability estimates are adjusted or smoothed

Laplace smoothing using an m-estimate assumes that each feature is given a prior probability, p, that is assumed to have been previously observed in a "virtual" sample of size m:

$$P(t_j|C_i) = \frac{N_{ij} + m \cdot p}{N_i + m}$$

- For binary features, p is simply assumed to be 0.5

[13] Problem związany z wykorzystaniem naiwnego klasyfikatora Bayesa wiąże się z tym, że mnożymy bardzo wiele prawdopodobieństw. Wystarczy by jedno z nich było równe zero, aby końcowy wynik nie miał sensu. Wyobraźmy sobie, że jakaś cech nie występuje w dokumentach uczących danej klasy, a przykład, który mamy zaklasyfikować akurat ją posiada. Aby uniknąć wyzerowania prawdopodobieństwa w takiej sytuacji, stosuje się technikę dostosowywania lub wygładzania (ang. smoothing) prawdopodobieństw. W najprostszym przypadku, zamiast zera, przyjmuje się bardzo małą stałą. W trochę bardziej profesjonalnym wydaniu, stosujemy tzn. wygładzanie Laplace'a. W technice tej agregujemy to, co wiemy z danych z tym, co określa się mianem wirtualnej próbki, gdzie każda wartość cechy ma predefiniowane prawdopodobieństwa wystąpienia. Przykładowo, dla cechy binarnej, rozmiar próbki m wynosi 1, a prawdopodobieństwa wystąpienia albo 0 albo 1 jest równe 0.5. W rezultacie dodajemy dodatnią wartość zarówno do licznika, jak i mianownika, chroniąc ułamek przed wyzerowaniem. Warto zwrócić uwagę na fakt, że to, co dodajemy do licznika jest mniejsze od tego, co dodajemy do mianownika.

Bayesian Classification (5) - Smoothing

	Doc	Content	SPAM
training	D1	business proposal	yes
training	D2	gold gold	yes
training	D3	gold bank business	yes
training	D4	bank transfer	no
training	D5	business business bank	no
test	**D6**	business business gold	?

|IDICTI| = 5, $N_{yes\text{-}term}$ = 7, $N_{no\text{-}term}$ = 5

the number of occurences of t_j *in docs belonging to* class C_i

$$P(t_j|C_i) = \frac{N_{ij} + 1}{N_{i\text{-}term} + |IDICTI|}$$

"add one"

dictionary size

the number of all terms in docs belonging to class C_i

ADD ONE SMOOTHING

P(yes) = 3/5

P(businesslyes) = (2+1)/(7+5) = 3/12

P(goldlyes) = (3+1)/(7+5) = 4/12

P(yesID6) ≈ 3/5·(3/12)²·4/12 = 0.0125

yes

P(no) = 2/5

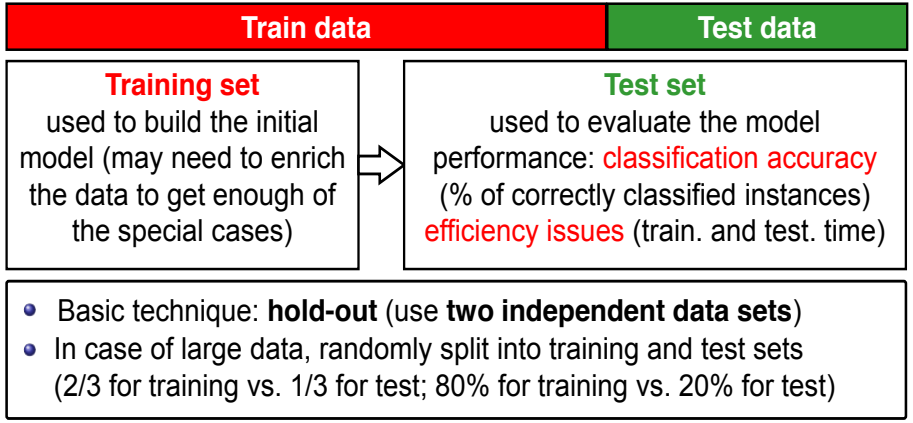
P(businesslno) = (2+1)/(5+5) = 3/10

P(goldlno) = (0+1)/(5+5) = 1/10

P(noID6) ≈ 2/5·(3/10)²·1/10 = 0.0036

[14] Pokażmy teraz, że klasyfikator Bayesa jest ogólną metodą, którą należy dostosować do konkretnego zastosowania i reprezentacji danych. Załóżmy, że wykorzystywana jest reprezentacja Bag-of-words oraz technika wygładzania. Dysponujemy 5 dokumentami uczącymi: 3 z klasy yes-spam oraz 2 z klasy no-spam. Dla dokument testowego: business business gold, klasa ma być dopiero określona. Prawdopodobieństwa dla klas liczymy jak poprzednio, ale w związku z tego, że zmieniła się reprezentacja dokumentów, to prawdopodobieństwa warunkowe dla poszczególnych termów w danej klasie obliczamy już w inny sposób. Dokładniej, dla każdej klasy tworzymy mega-dokument, który zawiera wszystkie dokumenty uczące, które do niej należą. Dla każdego termu z dokumentu testowego, prawdopodobieństwo policzylibyśmy jako iloraz liczby jego wystąpień w takim mega-dokumencie do liczby wszystkich termów, które w nim występują. Przykładowo, w dokumentach klasy yes-sapm jest 7 termów i 2 wystąpienia termu business, a w dokumentach klasy no-spam jest 5 termów i 0 wystąpień termu gold. Zastosujemy jednak od razu technikę wygładzania o nazwie add-one. Polega ona, jka sugeruje nazwa, na sztucznym zwiększeniu liczby wystąpień każdego termu w każdej klasie o 1, co skutkuje dodaniem 1 w liczniku. W związku z tym, że robimy to dla każdego termu ze słownika, to w mianowniku trzeba dodać jego rozmiar, a więc liczbę wszystkich różnych termów, które występują we wszystkich dokumentach uczących. Dzięki temu prawdopodobieństwo nie wyzeruje się. Aby obliczyć końcowe wyniki, należy pomnożyć prawdopodobieństwa składowe z zastrzeżeniem, że jeśli jakiś term pojawia się w dokumencie testowym więcej niż raz (a jest to ważne dla Bag-of-words), to odpowiednie prawdopodobieństwo jest uwzględniane (tj. mnożone) tyle razy, ile wynosi liczba wystąpień tego termu. Spójrz na term business, który występuje w dokumencie testowym 2 razy oraz term gold, który występuje tylko raz.

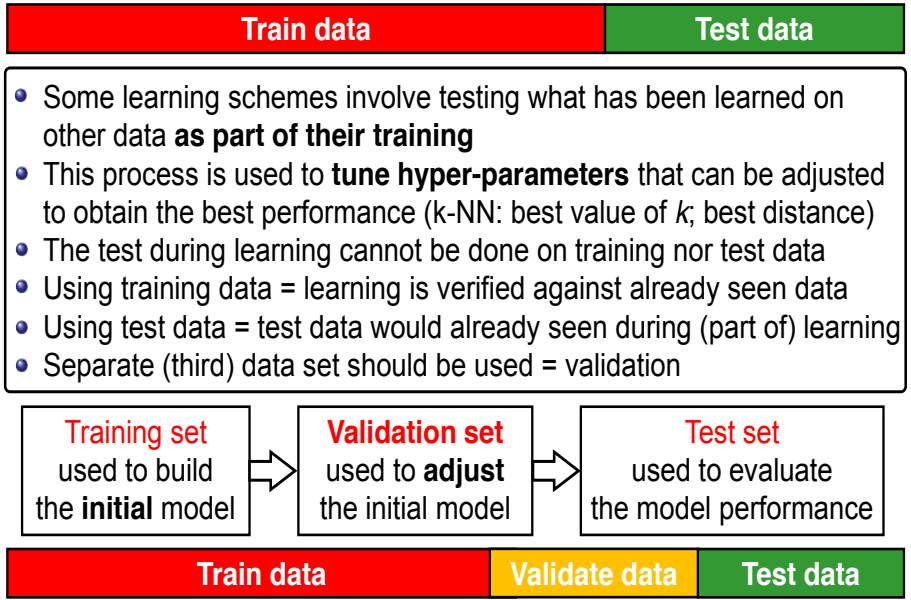
Classification - Evaluating Models



Końcowe prawdopodobieństwo warunkowe dla klasy yes-spam jest większe, więc byłaby ona rekomendowana dla rozważanego dokumentu.

[15] W standardowym schemacie uczenia algorytmów klasyfikacji dostępny zbiór przykładów uczących dzielony jest na dwa podzbiory: większy, treningowy i mniejszy, testowy. Ten pierwszy służy do konstrukcji, tj. właściwego uczenia, modelu. W związku z tym, musi być skonstruowany tak, aby dostępne w nim przykłady były zróżnicowane, w tym m.in. wszystkie klasy muszą w nim być reprezentowane. Drugi, zwykle dwa do cztery razy mniejszy od treningowy, służy do oceny jakości skonstruowanego modelu. Z jednej strony, ocena taka musi odnosić się do porównania przykładów, dla których pożądana klasyfikacja jest znana. Przykładowo, trafność klasyfikacji ujmuje procent przykładów, dla których prawidłowa klasa została przewidziana przez algorytm. Z drugiej strony, ogromnie ważne jest, by ten zbiór nie był wcześniej dostępny dla algorytmu uczącego, bo mógłby on się dostosować do tych danych. Jest to niewskazane. Klasyfikator powinien mieć zdolność generalizacji, tj. uogólniania sugerowanej klasyfikacji do innych danych niż te, które widział na etapie uczenia. Poza miarami jakości algorytmy ocenia się też pod względem aspektów wydajnościowych takich jak czas uczenia czy wykonania.

Classification - Validation



[16] W bardziej zaawansowanych schematach uczenia pojawia się trzeci podzbiór, tzw. zbiór walidujący. Korzysta się z niego wtedy, gdy algorytm wymaga dostosowania wartości meta-parametrów, jak k w k-NN czy definicja odległości. Walidacja jest de facto testowaniem w czasie uczenia i pozwala wybrać najlepsze wartości meta-parametrów. Niezmiernie istotne jest, by był to zbiór oddzielny od treningowego i testowego, bo z jednej strony nie można testować na danych wykorzystanych we właściwym uczeniu, a z drugiej, nie powinniśmy testować na danych, który były składową uczenia. Zwykle taki podział na trzy podzbiory stosuje się, gdy dostępny zbiór danych uczących jest duży. W przeciwnym razie, stosuje się bardziej zaawansowane techniki walidacji, jak walidacja krzyżowa (cross validation) albo ocena samowsporna (bootstrap validation).

What is Prediction?

Prediction vs. classification

- Prediction is **similar** to classification, because it uses a model to predict unknown (or future) value based on values of other attributes
- Prediction is **different** from classification, because the model is used to predict values of a numeric target attribute rather than categorical class label

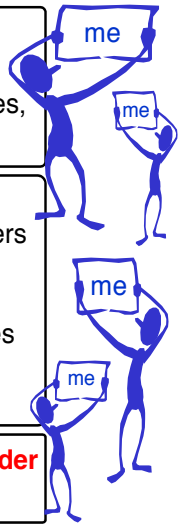
- Prediction models thought of as continuous-valued functions
- Most common approach: regression analysis (linear and multiple regression, non-linear regression, etc.)
- Most common application domains: **personalization**, credit scoring, predict customer loyalty, etc.

[17] Drugim zagadnieniem, które zostanie omówione podczas wykładu jest predykcja. Jej zestawienie z tematyką klasyfikacji jest nieprzypadkowe z co najmniej dwóch względów. Po pierwsze, zadanie predykcji jest bardzo zbliżone do klasyfikacji, ponieważ polega na przewidzeniu nieznanej lub przyszłej wartości pewnego atrybutu lub cechy na podstawie wartości znanych cech innych atrybutów. Różnica w stosunku do klasyfikacji polega na tym, że w predykcji przewidywana jest wartość numeryczna, a w klasyfikacji chcemy wypracować etykietę klasy. Po drugie, pewne kroki znane z algorytmów klasyfikacji będą albo służyć jako istotne składowe algorytmów predykcji albo te ostatnie opierać się będą na podobnej regule działania. Modele predykcji można sobie wyobrazić jako funkcje o charakterze ciągłym, a tradycyjne algorytmy opierają się tu na wykorzystaniu regresji. Z kolei najważniejsze dziedziny zastosowań odnoszą się do szeroko rozumianej personalizacji, estymacji ryzyka kredytowego, tj. szans na spłatę kredytu przez klienta, czy też poziomu lojalności klienta w relacji z firmą.

Personalization

The Problem

Dynamically serve customized content (books, movies, pages, products, tags, etc.) to users based on their profiles, preferences, or expected interests



Why we need it?

- Information spaces are becoming much more complex for users to navigate (huge online repositories, social networks, mobile applications, blogs, …)
- For businesses: need to grow customer loyalty / increase sales
- Industry research: successful online retailers are generating as much as 35% of their business from recommendations

The most common type of personalization systems: **recommender systems** (content-based filtering and collaborative filtering)

[18] Skupimy się na zagadnieniu personalizacji, którą rozumie się jako przedstawienie treści dynamicznie dostosowanej do użytkownika na podstawie jego profilu, preferencji, oczekiwań lub zainteresowań. Treści takie mogą mieć np. postać książek, filmów, stron internetowych lub produktów. Rozwój algorytmów personalizacji wypływa z potrzeb różnych grup. Po pierwsze, użytkownikom coraz trudniej poruszać się i nawigować w coraz większych przestrzeniach produktów takich jak sklepy internetowe, aplikacje mobilne lub sieci społecznościowe. Po drugie, wymagają ich także dostawcy usług, bo pozwalają im na zwiększenie lojalności klientów oraz zwiększenie sprzedaży. Szeroko dostępne są wyniki badań, które potwierdzają, że zysk ze sprzedaży produktów z personalizacji stanowi ponad 1/3 przychodów sklepów internetowych. W różnych kontekstach algorytmy personalizacji przyjmują specyficzną formę. My skupimy się na systemach rekomendacyjnych, a w szczególności na tzw. filtrowaniu na bazie zawartości czy wspólnym lub społecznym filtrowanie, co jest dość nieszczęśliwym (acz oficjalnym) tłumaczeniem angielskiego pojęcia 'collaborative filtering'.

Content-based Recommendation

- Give recommendations to a user based on items with “similar” content in the user’s profile
- Predictions for unseen (target) items are computed based on their similarity (in terms of content) to items in the user profile

[19] Rekomendacja bazująca na zawartości opiera się na analizie profilu zawartości użytkownika. Ogólną zasadę można by sformułować tak, że użytkownikowi polecane będą te produkty lub obiekty, które mają podobną zawartość do tych, które znajdują się w jego profilu. Kluczowym krokiem w realizacji predykcji dla obiektów, z którymi użytkownik nie miał jeszcze styczności, jest obliczenie ich podobieństwa - w kontekście zawartości - z tymi obiektami, z którymi taką styczność już miał. Przykładowo, moglibyśmy rozważyć użytkownika, którzy lubuje się w filmach Vegi takich, jak Botoks czy Pitbull (Nowe porządki) oraz Smarzowskiego, jak Dom zły czy Wesel. Filmy te - pomimo, że dotyczy różnej tematyki - są bardzo charakterystyczne i bez trudu można zorientować się, kto był ich reżyserem. Dla użytkownika, który widział niektóre z nich, nie będzie wielkim ryzykiem rekomendacja Kobiet mafii lub Drogówki, bo szanse, że będą one mu się podobać są ogromne. Warto tu podkreślić, że rekomendacja taka może różnić się co do poziomów (w przykładzie na slajdzie porównaj poziom wysoki względem średniego).

Recommendation Task

Basic formulation as a **prediction problem**:

Given a profile P_a for a user ***a***, and a target item i_p, predict the preference score of user ***a*** on item i_p

- Typically, the profile P_a contains **preference scores** by *a* on some other items, $\{i_1, \ldots, i_k\}$ different from i_p
- Preference scores on items $i_1, \ldots, i_k$ may have been obtained explicitly or implicitly

[20] Podstawowe sformułowanie problemu predykcji w tym kontekście jest następujące: dany jest profil użytkownika oraz obiekt docelowy; zadanie polega na predykcji oceny, którą wystawiłby użytkownik dla tego obiektu w ten sposób, by odzwierciedlała ona jego preferencje. W typowych zastosowaniach profil zawiera oceny wystawione przez użytkownika dla podzbioru różnych obiektów/produktów, przy czym wystawienie to mogło mieć charakter bezpośredni - wprost lub pośredni - niejako przy okazji korzystania z sieci lub serwisu.

Collaborative Filtering (1)

The most prominent approach to generate recommendations
- Used by large, commercial e-commerce sites
- Well-understood, various algorithms and variations exist
- Applicable in many domains (book, movies, DVDs, ...)

Approach to recommend items
- Use the "wisdom of the crowd"
- Give recommendations to a user based on preferences of “similar” users

Basic assumption and idea
- Users give ratings to catalog items (implicitly or explicitly)
- Customers who had similar tastes in the past, will have similar tastes in the future
- User preferences remain stable and consistent over time

[21] Najbardziej znaczące podejście do generacji rekomendacji opiera się na społecznym filtrowaniu. Jest ono wykorzystywane na największych serwisach, które mają szerokie grona użytkowników, jak sklepy internetowe, serwisy filmowe lub muzyczne. Wielka siła tego podejścia bierze się z mnogości istniejących algorytmów, których dziedzina zastosowań nie jest w żaden sposób ograniczona. Nie ma bowiem dla nich znaczenia, czy rozważanymi obiektami są książki, filmy, piosenki lub żarty. Ogólna zasada ich działania opiera się na wykorzystaniu mądrości tłumu, a więc wykorzystaniu do rekomendacji użytkowników o podobnych preferencjach do użytkownika, dla którego predykcja jest dokonywana. Taki sposób postępowania wymaga jednak spełnienia kilku założeń. Po pierwsze, użytkownicy muszą dokonywać ocen obiektów lub produktów. Po drugie, zakłada się, że gusta użytkowników nie podlegają wielkim zmianom wraz z upływem czasu. Wreszcie - co związane jest z poprzednim, ale w kontekście analizy zbiorowości - użytkownicy o podobnych preferencjach w przeszłości, powinni też zgadzać się co do swoich przyszłych wyborów.

Collaborative Filtering (2)

Input: *only a matrix of given user–item ratings*

	Wesele	Dom zły	Kobiety mafii	Pitbull
Agata	7	6	3	7
Karolina	7	4	4	6
Natalia	3	7	7	2
Asia	4	4	6	2
Julia	7	4	3	?

Can we predict Julia's rating on the unseen item Pitbull?

Output types:

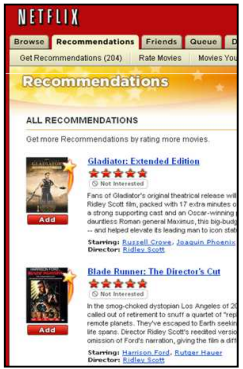
- **A (numerical) prediction** indicating to what degree the current user will like or dislike a certain item (not yet used/seen/rated)
- **A top-N list of recommended items**

[22] Aby zrozumieć, na czym polega zadanie rekomendacji rozważmy przykład, w którym użytkownicy dokonują ocen filmów. Wejściem dla działania naszego algorytmu jest macierz ocen 5 użytkowników dla 4 filmów. Oczywiście w praktycznych zastosowaniach macierz ta byłaby zdecydowanie większa. Niektóre z komórek macierzy są wypełnine. Przykładowo, korzystając ze skali od 1 do 7, Julia oceniła film Dom zły na 4. Innych ocen jednak brakuje. Przykładowo, nie wiemy czy i w jakim stopniu Indze będzie podobał się Pitbull. Wyniki, których możemy spodziewać się w zadaniu predykcji są dwojakiego typu. Z jednej strony, chodzić może po prostu o predykcję numerycznej wartości, wyrażającej stopień, w jakim danemu użytkownikowi podoba się (lub nie) konkretny film. Z drugiej, jeśli takich predykcji dokonać by dla wszystkich filmów, których użytkownik nie widział, to celem może być utworzenie rankingu takich filmów i wyłuskanie z niego k filmów, dla których predykcja oceny jest najwyższa.

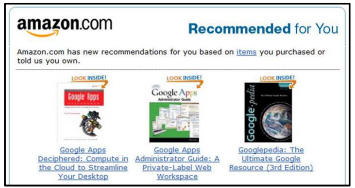
Collaborative Filtering (3) - Famous Sites

- Many examples in real world applications
- Do not need a representation for items
- Compare user profiles instead

Netflix, MovieLens, Amazon, Spotify, Jester (jokes), EachMovie, BookCrossing



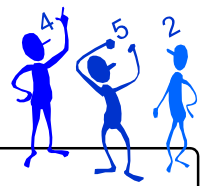
Famous Netflix competition in 2006-09 grand prize of $1,000,000



[23] Aby potwierdzić przydatność tego typu algorytmów wystarczy wspomnieć nazwy serwisów, które korzystają ze społecznego filtrowania (patrz prawy górny róg). Co istotne, przedmiotem ich zainteresowania są różne obiekty, od filmów przez książki i muzykę do żartów. W wielu wypadkach nie jest potrzebna tu formalna reprezentacja obiektów. Wystarczy porównywać profile użytkowników, tj. ich opinie wyrażone w sposób jawny lub pośredni. Jednym z najsłynniejszych przykładów serwisu przejawiającego głębokie zainteresowanie algorytmami rekomendacji od bardzo wielu lat jest Netflix. Już w latach 2008-9 serwis ten urządził otwarty konkurs, w którym udostępnił bazę kilkunastu milionów ocen i obiecał drużynie, która opracuje algorytm istotnie lepszy, do tego czym dysponowali w tamtych czas, nagrodę w wysokości miliona dolarów.

Collaborative Filtering (4) - Explicit Ratings

- Most commonly used (movie, book, music, joke ratings)
- Probably the most precise ratings
- 1-5, 1-7 Likert response scales



Research topics

- Optimal granularity of scale; indication that 10-point scale is better accepted in movie domain
- A more fine-grained scale was chosen (–10 to +10 and a graphical bar) in the joke recommender, where a continuous scale were used
- Multidimensional ratings (multiple ratings per movie such as ratings for actors and sound)

Main problems

- Users not always willing to rate many items (number of available ratings too small, sparse rating matrices, poor recommnedation quality)
- How to stimulate users to rate more items?

[24] Zanim przejdziemy do omówienia algorytmów, skupmy się jeszcze przez chwilę na pochodzeniu ocen. Pierwsze ich źródło to noty wystawione wprost przez użytkowników z wykorzystaniem precyzyjnie zdefiniowanej skali. To jest najczęściej stosowane podejście przy ocenach filmów lub książek, gdzie stosowana jest tzw. skala Likerta z przedziałem zmienności np. od 1 do 5 lub od 1 do 10. W tym kontekście prowadzony jest szereg badań, które zmierzają do tego, by określić, jaki typ skali jest odpowiedni dla konkretnej dziedziny zastosowań. Przykładowo, stwierdzono, że dla filmów powinno się stosować skalę 10-punktową, a w kontekście oceny jakości żartów powinna być dopuszczona skala z ocenami ujemnymi. Z tego typu ocenami wiążą się też liczne problemy. Przykładowo, zbyt rzadka macierz ocen może prowadzić do rekomendacji o zbyt słabej jakości. W tym kontekście, rozważa się np. w jaki sposób stymulować użytkowników do wystawienia ocen, dla produktów, z których korzystali, filmów, które widzieli, książek, które przeczytali, itd.

Collaborative Filtering (5) - Implicit Ratings

- Typically collected by the web shop or application in which the recommender system is embedded
- When a customer buys an item, for instance, many systems interpret this behavior as a positive rating
- **Clicks, page views, downloads, time spent on a page**
- Can be collected constantly and do not require additional efforts from the side of the user

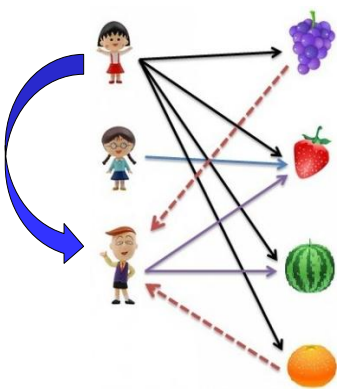
Main problem

- One cannot be sure whether the user behavior is correctly interpreted
- For example, a user might not like all the books he or she has bought; the user also might have bought a book for someone else

[25] Oceny mogą być także wystawiane w sposób pośredni, co jest powszechnie wykorzystywane w systemach i sklepach internetowych, których częścią składową są moduły rekomendacyjne. Przykładowo, jeśli użytkownik kupuje jakiś produkt, system może to interpretować jako pozytywną ocenę tego produktu. Takie niejawne “oceny" mogą więc pochodzić z kliknięć, wyświetleń strony, ściągnięć lub czasu spędzonego na oglądaniu jakiegoś produktu. Zaletą tego sposobu kolekcji ocen jest fakt, że mogą być one zbierane w sposób ciągły i nie wymagają dodatkowej interakcji ze strony użytkownika, bo zbieramy je przy okazji obserwacji jego zachowania. Podstawowy problem wiąże się tu z brakiem pewności co do interpretacji takiego zachowania. Przykładowo, sam fakt, że użytkownik oglądnął jakiś film albo kupił książkę, nie oznacza, że podobały mu się one. W porównaniu z ocenami wyrażonymi w sposób pośredni, wyzwaniem, z którym trzeba się tu zmierzyć jest więc brak precyzji.

User-based Collaborative Filtering (1)

User-based collaborative filtering



- Imagine that we want to recommend a movie to *Julia*
- We could assume that similar people will have similar taste
- Suppose that *Julia* and *Asia* have seen the same movies, and they rated them all almost identically
- *Julia* has not seen *Pitbull* and *Asia* did
- If *Asia* loves that movie, it sounds logical to think that *Julia* will too
- We could create an artificial rating based on the similarity between their ratings

”If you like it, I’ll like it as well”

[26] Przechodzimy do omówienia czterech algorytmów służących do predykcji ocen w ramach społecznego filtrowania. Pierwszy z nich nazywa się predykcją bazującą na analizie (ocen) innych użytkowników, po angielsku user-based collaborative filtering. Ta metoda opiera się na założeniu, że podobni użytkownicy mają podobne preferencje. Wyobraźmy więc sobie, że chcemy polecić film Indze. Załóżmy, że Inga i Julia widziała te same filmy i oceniły je niemal identycznie. Co więcej, Julia widziała już Pitbulla, a Inga jeszcze nie. Jeśli Julii podobał się ten film, to sensowne jest, by założyć, że będzie się podobał także Indze. Moglibyśmy więc skonstruować sztuczną ocenę Pitbulla dla Ingi na podstawie analizy podobieństwa ocen wystawionych przez Julię i Ingę. Algorytm ten można więc wytłumaczyć jednym zdaniem: jeśli ty coś lubisz, to ja też będę to lubił, bo generalnie mamy podobne preferencje. Dla przykładu po lewej stronie: rozważmy trzech użytkowników i cztery owoce. Użytkownik, dla którego przeprowadzamy analizę lubi truskawki i arbuza. Zastanawiamy się, jakie inne owoce może lubić. Sensownie byłoby spojrzeć na osoby, które też lubią truskawki i arbuzy. Warunek ten spełnia górny użytkownik, który lubi też winogrona i pomarańcze. W tym kontekście, może te owoce rekomendować też użytkownikowi dolnemu.

User-based Collaborative Filtering (2)

- Predictions for unseen (target) items are computed based the other users' with similar interest scores on items in user *a*'s profile
- Users with similar tastes (aka "k-nearest-neighbors")

	Wesele	Dom zły	Kobiety mafii	Pitbull
Agata	7	6	3	7
Karolina	7	4	4	6
Natalia	3	7	7	2
Asia	4	4	6	2
Julia	7	4	3	?

- Given an active user (Julia) and an item (Pitbull) not yet seen by Julia
- **Find a set of users (peers/nearest neighbors)** who like the same items as Julia and who have rated Pitbull
- **Predict a rating**, to check if Julia will like Pitbull
- Do the same for all items Julia has not seen (Botoks, Cicha Noc, Twój Vincent, Listy do M., …) and recommend the best-rated ones

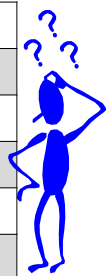
[27] Predykcji oceny dla obiektów nieznanych użytkownikowi chcemy dokonać na podstawie analizy ocen innych użytkowników, którzy mają podobne do niego preferencje. Z pomocą przychodzi tu ponownie pojęcie k najbliższych sąsiadów. Sposób postępowania przy predykcji oceny Ingi dla Pitbulla będzie więc następujący. Najpierw znajdziemy k użytkowników o najbardziej podobnych do Ingi preferencjach, którzy - co ważne - ocenili Pitballa. Następnie ich oceny wykorzystamy do wypracowania predykcji oceny, którą mogłaby wystawić Inga. To samo można by zrobić dla innych filmów, których Inga nie widziała (jak Botoks czy Cicha Noc), a potem wyłuskać wśród nich ograniczoną liczbę takich filmów, dla których przewidywana ocena jest najwyższa.

User-based Collaborative Filtering (3)

Three crucial questions

- How do we **measure similarity/distance** between user *a* and other users according to interest scores or ratings?
- How **many neighbors** should we consider?
- How do we **generate a prediction** from the neighbors' ratings?

	Wesele	Dom zły	Kobiety mafii	Pitbull
Agata	7	6	3	7
Karolina	7	4	4	6
Natalia	3	7	7	2
Asia	4	4	6	2
Julia	7	4	3	?



[28] Ogólna idea algorytmu jest bardzo prosta. Pojawiają się jednak kluczowe pytania, na których odpowiedzi będziemy udzielać przy okazji omówienia kolejnych slajdów. Po pierwsze, jak mierzyć podobieństwo lub odległość między użytkownikami. Po drugie, ilu najbliższych sąsiadów wziąć pod uwagę. Po trzecie, jak ostatecznie obliczyć predykcję nieznanej oceny. Nie ma na te pytania jednej, uniwersalnie najlepszej odpowiedzi. Postaramy się więc omówić podstawowe możliwości.

User-based Collab. Filter. (4) - User Similarity (1)

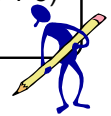
Rating are seen as vectors in n-dimnesional space

- *a*, *b*: users
- $r_{a,p}$: rating of user *a* for item *p*
- P: set of items rated by both *a* and *b*

Cosine similarity measures the angle between the vectors

$$sim(a,b) = \frac{\sum_{p \in P} r_{a,p} \cdot r_{b,p}}{\sqrt{\sum_{p \in P} r_{a,p}^2} \cdot \sqrt{\sum_{p \in P} r_{b,p}^2}}$$

- 1 means very similar, 0 - no similarity, -1 - dissimilar
- Works well in case of user ratings (at least a range of 1-5)
- Produces better results in item-to-item filtering



	W	D	K	T
Asia	4	4	6	2
Julia	7	4	3	?

$$sim(A,J) = \frac{4 \cdot 7 + 4 \cdot 4 + 6 \cdot 3}{\sqrt{4^2 + 4^2 + 6^2} \cdot \sqrt{7^2 + 4^2 + 3^2}} = 0.874$$

[29] Rozpoczniemy od obliczenia podobieństwa między użytkownikami. Będzie to w pewnym sensie powtórka sprzed kilku wykładów, gdy zajmowaliśmy się podobieństwem między dokumentami. Załóżmy, że rozważani są użytkownicy a oraz b, dane są ich oceny dla wybranych obiektów, a podzbiór obiektów, które ocenili zarówno a, jak i b, oznaczymy przez P. Podobieństwo między użytkownikami jest rozumiane jako podobieństwo między wektorami wystawionych przez nie ocen. Do jego obliczenia możemy więc wykorzystać miarę kosinusową, która jest zdefiniowana jako iloczyn wektorowy (suma iloczynów) przez iloczyn skalarny (iloczyn długości). Ważne jest przy tym, że ograniczamy się tu tylko do podzbioru obiektów P, a więc tych, które użytkownicy ocenili wspólnie. Oczywiście, 1 oznacza tu pełne podobieństwo, a -1 jego zupełny brak. Taka realizacja obliczeń świetnie sprawdza się, gdy wykorzystywane skale ocen mają stosunkowo nieduży zakres zmienności (np. od 1 do 5), ale praktyka pokazuje, że w praktyce lepiej się sprawdza, gdy porównuje się obiekty, a nie użytkowników. Dla porównania Uli i Ingi wykorzystywane są oceny, które wystawiły dla obiektów W, D i K. Wynik wskazuje na bardzo wysokie podobieństwo, bliskie 1. Intuicyjnie czujemy, że w kontekście porównania użytkowników jest ono odrobinę za wysokie. Widzimy bowiem, że gdy ocena Ingi jest wysoka, to Uli niska, a gdy Uli wysoka, to Ingi niska. To sugeruje właściwe rozwiązanie.

User-based Collab. Filter. (5) - User Similarity (2)

A popular similarity measure in user-based CF: **Pearson correlation**

$$sim(a,b) = \frac{\sum_{p \in P} (r_{a,p} - \bar{r}_a) \cdot (r_{b,p} - \bar{r}_b)}{\sqrt{\sum_{p \in P} (r_{a,p} - \bar{r}_a)^2} \cdot \sqrt{\sum_{p \in P} (r_{b,p} - \bar{r}_b)^2}}$$

- 1 means very similar, 0 means no correlation, -1 means dissimilar
- Works well in case of user ratings (where there is at least a range of 1-5)
- Can be seen as **adjusted cosine similarity** (accounts for the average ratings)

	W	D	K	T
Asia	4	4	6	2
Julia	7	4	3	?

Consider only items evaluated jointly by Asia and Julia

⇦ avarage rating = 14/3 = 4.67 (neglect rating for T)

⇦ avarage rating = 14/3 = 4.67

$$sim(A,J) = \frac{(4 - 4.67)\cdot(7 - 4.67) + (4 - 4.67)\cdot(4 - 4.67) + (6 - 4.67)\cdot(3 - 4.67)}{\sqrt{(4 - 4.67)^2+(4 - 4.67)^2+(6 - 4.67)^2} \cdot \sqrt{(7 - 4.67)^2+(4 - 4.67)^2+(3 - 4.67)^2}} = -0.69$$

[30] Najczęściej wykorzystywaną miarą w kontekście podobieństwa między użytkownikami jest współczynnik korelacji Pearsona, czyli kowariancja przez iloczyn odchyleń standardowych. Tu - podobnie jak w przypadku miary kosinusowej - 1 oznacza całkowity podobieństwo, a -1 jego zupełny brak. Można bowiem postrzegać współczynnik korelacji jako delikatnie zmienioną miarę kosinusową, która uwzględnia odchylenia od ocen średnich. Obliczenie zrealizowane dla porównania Ingi i Uli potwierdzają, że w istocie ich preferencje bardzo się od siebie różnią. Wartość miary podobieństwa jest silnie ujemna. Ponownie podkreślmy, że przy obliczeniach brane są pod uwagę tylko te filmy, które oceniły Inga i Ula. Tym samym ignorowany jest np. film T.

User-based Collab. Filter. (6) - User Similarity (3)

A popular similarity measure in user-based CF: **Euclidean distance**

$$sim(a,b) = \sqrt{\sum_{p \in P} (r_{a,p} - r_{b,p})^2}$$

- Ordinary straight-line distance between two points in Euclidean space
- A special cased of L-norm (L=2) (L=1 – Manhattan, L= infty – Chebyshev)
- Non-standardized – difficult to interpret for a single pair
- Distance rather than similarity (need for further transformation)

Natalia	3	7	7	2
Asia	4	4	6	2
Julia	7	4	3	?

$$sim(N,J) = \sqrt{(3-7)^2 + (7-4)^2 + (7-3)^2} = 6.40$$

$$sim(A,J) = \sqrt{(4-7)^2 + (4-4)^2 + (6-3)^2} = 4.24$$

[31] Drugą, często wykorzystywaną w kontekście porównywania użytkowników miarą jest odległość Euklidesowa. Nie mówimy już tu o średnich, tylko o najprostszej odległości między punktami w wielowymiarowej przestrzeni. Odległość Euklidesowa jest specjalnym przypadkiem tzw. L-normy dla L równego 2. Gdy L jest równe 1, to mówimy o odległości taksówkowej, a dla L równego nieskończoność o odległości Czebyszewa. Dwa drobne problemy związane z wykorzystaniem odległości Euklidesowej są następujące. Po pierwsze, odległość ta nie jest standaryzowana, więc ciężko interpretować wyniki tylko dla pojedynczej pary, bez szerszego kontekstu. Po drugie, jest to odległość, a nie podobieństwo, więc trzeba pamiętać, że im jej wartość jest mniejsza, tym lepiej i tym bardziej podobni użytkownicy. W tym kontekście, jeśli porównywać Ingę z Ulą i Faustyną, to bardziej podobna co do preferencji jest do niej Ula, bo odległość Euklidesowa jest tu mniejsza niż dla Faustyny.

User-based Collab. Filter. (7) - User Similarity (4)

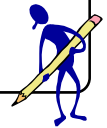
- Not always possible to compute a distance based on rating
- In some situations we may only have implicit binary values, e.g., whether a user did or did not select a document, book, movie, etc.

Tanimoto distance (based on Jaccard's coefficient)

- a, b: users
- $r_{a,p}$: "binary rating" of user a for item p (0 or 1)

How many items have been viewed jointly by both users as compared to the number of items which have been viewed by these users?

$$sim(a,b) = \frac{\sum_p r_{a,p} \cdot r_{b,p}}{\sum_p r_{a,p} + \sum_p r_{b,p} - \sum_p r_{a,p} \cdot r_{b,p}}$$



Natalia	1	0	0	1
Asia	1	1	1	0
Julia	1	1	0	?

$sim(N,J) = (1 \cdot 1 + 0 \cdot 1 + 0 \cdot 0)/(1+2-1) = 1/2$

$sim(A,J) = (1 \cdot 1 + 1 \cdot 1 + 1 \cdot 0)/(3+2-2) = 2/3$

[32] W niektórych przypadkach nie mamy dostępnych ocen na wielostopniowej skali, a tylko binarną informację o tym, czy użytkownik z czegoś skorzystał, coś kupił, coś wyświetlił, coś obejrzał, itd. Potrzebujemy więc miar, które byłyby stosowalne w takich kontekstach. Odpowiedzią jest tu odległość Tanimoto, która bazuje na współczynniku Jaccarda. W kontekście filmów mierzyłaby ona, ile filmów było widzianych przez dwóch użytkowników łącznie spośród tych, które sumarycznie widzieli oni oddzielnie. Dla porównania Ingi i Faustyny, podzielilibyśmy więc 1 film, który widziała jedna i druga, przez 2, które widziały one w sumie. W kontekście porównania Ingi i Uli, analogicznie podzielilibyśmy 2 filmy, które widziały razem, przez 3, które widziały w sumie.

User-based Collab. Filter. (8) - Making Predictions

Making predictions

- Find users who like the same items as *Julia* and who have rated *Pitbull*
- k-nearest-neighbor strategy
- Use an aggregate of their ratings to predict if *Julia* will like *Pitbull*

	Wesele	Dom zły	K. mafii	Pitbull	**Pearson**	Euclidean	Cosine
Agata	7	6	3	7	**0.85**	2.00	0.983
Karolina	7	4	4	6	**0.97**	1.00	0.995
Natalia	3	7	7	2	**-0.97**	6.40	0.787
Asia	4	4	6	2	**-0.69**	4.24	0.874
Julia	7	4	3	?	**1.00**	0.00	1.000

The simplest strategy: an average predicted rating (of K nearest neighbors)

$$predction(r_{a,i}) = \frac{\sum_{u=1,\ldots,K} r_{u,i}}{K}$$

K=1 ⇨ $r_{J,P} = 6$

K=2 ⇨ $r_{J,P} = (6+7)/2 = 6.5$

K=3 ⇨ $r_{J,P} = (6+7+2)/3 = 5$*

$r_{J,P} = (6+7)/2 = 6.5$

* *it does not make sense to account for users with dissimilar preferences (Asia vs. Julia)*

[33] Przejdźmy teraz do udzielenia odpowiedzi na inne pytanie, odnoszące się do sposobu obliczenia predykcji oceny. Wróćmy do przykładu z Ingą i Pitbullem. Zbadawszy podobieństwo Ingi do użytkowników, którzy ocenili Pitbulla, musimy wśród nich zidentyfikować k najbardziej podobnych, czyli k najbliższych sąsiadów. Potem zagregujemy, w mądry sposób, oceny, które wystawili, aby ocenić czy są szanse na to, by Indze podobał się Pitbull. Podstawowy pomysł na agregację to obliczenie średniej z ocen k najbliższych sąsiadów. W tabeli zaprezentowano podobieństwo między Ingą oraz 4 innymi użytkownikami. Przedstawiono wyniki dla trzech miar, ale my wykorzystamy do dalszych obliczeń współczynnik korelacji Pearsona. Jeśli założylibyśmy, że K jest równe 1, to weźmiemy pod uwagę tylko najbardziej podobnego użytkownika, czyli Julię i za predykcję oceny dla Ingi uznamy ocenę Julii dla Pitbulla, czyli 6. Jeśli K byłoby równe 2, to uwzględnilibyśmy oceny Julii i Karoliny, uśredniając 6 i 7 (predykacja 6.5). Gdy K jest równe 3, to trzeba by też uwzględnić ocenę Uli, ale nie miałoby to żadnego sensu, bo Ula nie jest podobna w swoich preferencja do Ingi (miara podobieństwa jest silnie ujemna). Nie liczy się więc prostej średniej z ocen k najbliższych sąsiadów, a tylko spośród tych sąsiadów, którzy spełniają minimalny próg podobieństwa. Może on być definiowany przez użytkownika albo domyślny (w przypadku współczynnika korelacji - jego wartość musi być dodatnia). Ostatecznie, nawet jeśli przyjąć K=3, to ocena Uli nie byłaby uwzględniona przy dokonaniu predykcji dla Ingi.

User-based Collab. Filter. (9) - Making Predictions

Closer neighbors count more:

Non-weighted average for K = 2: 6.5

$$predction(r_{a,i}) = \frac{\sum_{u=1,\ldots,K} r_{u,i} \cdot sim(a,u)}{\sum_{u=1,\ldots,K} sim(a,u)}$$

Weighted average for K = 2:

$r_{I,P} = (6 \cdot 0.97 + 7 \cdot 0.85)/(0.97+0.85) = 6.46$

	Wesele	Dom zły	K. mafii	Pitbull	Pearson	Average
Agata	7	6	3	7	**0.85**	5.33
Karolina	7	4	4	6	**0.97**	5.00
Natalia	3	7	7	2	-0.97	5.67
Asia	4	4	6	2	-0.69	4.67
Julia	7	4	3	?	1.00	4.67

- Check if the neighbors' rating are higher/lower than their average
- Add/subtract the neigbors' bias from the active user's average

$$predction(r_{a,i}) = \bar{r}_a + \frac{\sum_{u=1,\ldots,K} (r_{u,i} - \bar{r}_u) \cdot sim(a,u)}{\sum_{u=1,\ldots,K} sim(a,u)}$$

mean rating for user a

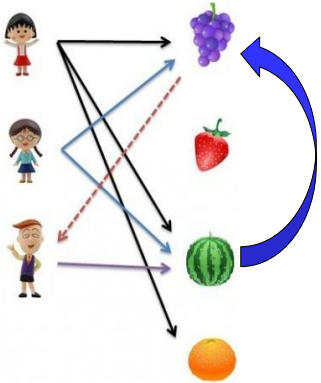
weighted average of deviations from the neighbors' mean ratings

For K = 2: $r_{J,P} = 4.67 + \frac{0.97 \cdot (6 - 5) + 0.85 \cdot (7 - 5.33)}{0.97 + 0.85} = 4.67 + 0.71 = 5.38$

[34] Bardziej zaawansowane modele predykcji wykorzystują miary podobieństwa w agregacji. Zamiast zwykłej średniej można więc liczyć średnią ważoną, gdzie w liczniku oceny k najbliższych sąsiadów mnożymy przez podobieństwo, a w mianowniku dzielimy przez sumę podobieństw. Jeśli dokonać predykcji oceny dla Ingi przy K=2, to zamiast 6.5 wyszłoby 6.46. Dlaczego? Ano przy obliczeniu średniej ważonej odrobinę bardziej istotna byłaby ocena Julii, czyli 6, niż Faustyny, czyli 7. Świetnie jednak wiadomo, że różni użytkownicy stosują różne punkty odniesienia przy wystawianiu ocen. Znacie takie osoby, dla których ocena 8 na skali od 1 do 10 jest czymś bardzo słabym, ale też inne, dla których 6 na takiej skali jest już wielkim wyróżnieniem. Przy agregacji ocen warto by wziąć pod uwagę taki punkt odniesienia, jakim jest średnia ocen wystawionych przez danego użytkownika. To prowadzi nas do ostatecznej, najbardziej sensownej propozycji dla wypracowania rekomendacji. Obliczając predykcję oceny, wychodzimy ze średniej ocen danego użytkownika i modyfikujemy ją o średnią ważoną od średnich ocen k najbliższych sąsiadów dla danego obiektu. Przykładowo dla Ingi i K=2, startujemy z poziomu jej średniej - 4.67 i modyfikujemy ją o ważone odchylenia od średnich ocen Julii i Karoliny dla Pitbulla. W związku z tym, że zarówno Julia, jak i Karolina, oceniły Pitbulla wyżej od średniej, to predykcja oceny dla Ingi też jest wyższa niż jej średnia (ostatecznie 5.38).

Item-based Collaborative Filtering (1)

Item-based collaborative filtering



- Imagine that we want to recommend a movie to *Julia*
- We focus on what items from all the options are similarly rated to what we know she enjoys
- Suppose that *Pitbull* and *Wesele* have been rated almost identically
- *Julia* has not seen *Pitbull but she has seen Wesele*
- If others rate *Pitbull* high/low, it sounds logical to think that *Julia* will rate them similarly too

"If others like that item, I'll like this item"

[35] Drugim algorytmem predykcji, który omówimy, jest filtrowanie społecznie opierające się nie na użytkownikach, a na obiektach, po angielsku item-based collaborative filtering. Znów rozważmy przykład z rekomendacją filmów dla Ingi. Tym razem skupimy się tu analizie obiektów, które są podobnie oceniane do tego, co Inga lubi. Załóżmy, że Pitbull i Wesele są oceniane podobnie przez wszystkich użytkowników. Wtedy - jeśli Inga nie widziała Pitbulla, a widziała Wesele, a użytkownicy podobnie ocenialiby te dwa filmy, to sensownie byłoby założyć, że jeśli inni użytkownicy ocenili Pitbulla wysoko/nisko, to podobnie oceniłaby go Inga. W jednym zdaniu algorytm ten można więc wyjaśnić następująco: jeśli inni lubią ten obiekt, to ja też będę go lubić, bo preferencje dla tych dwóch obiektów są bardzo podobne. Dla przykładu z lewej strony, użytkownik dolny lubi arbuza i zastanawiamy się, jakie inne owoce może jeszcze lubić. Spójrzmy na innych użytkowników, którzy lubią arbuza. Aż dwóch z nich lubi też winogrona. Sensowne jest więc założyć, że dolny użytkownik też będzie je lubił.

Item-based Collaborative Filtering (2)

Predictions for unseen (target) items are computed **based the user's ratings for items with similar interest scores for all users**

- Items with similar ratings (aka "nearest neighbors")
- *k*-nearest-neighbor (*k-NN*) strategy or simply account for all items

	Wesele	Dom zły	Kobiety mafii	Pitbull
Agata	7	6	3	7
Karolina	7	4	4	6
Natalia	3	7	7	2
Asia	4	4	6	2
Julia	7	4	3	?

Given an active user (Julia) and an item (Pitbull) not yet seen by Julia

- **Find a set of items** which ared liked the same as Pitbull by all users and **which have been rated by Julia**
- **Predict a rating**, to check if Julia will like Pitbull
- Can be generalized to more items

[36] W przypadku predykcji na podstawie ocen innych produktów sposób postępowania jest nieco inny. Badamy podobieństwo ocen dla konkretnego obiektu z wszystkimi pozostałymi obiektami; wśród nich identyfikujemy k ocenianych najbardziej podobnie i agregujemy oceny dla tych obiektów wystawione przez użytkownika, dla którego dokonujemy predykcji. Przykładowo, dla Ingi najpierw badamy, które filmy ocenione przez Ingę są oceniane przez wszystkich użytkowników podobnie do Pitbulla. Oceny wystawione przez Ingę dla najbardziej podobnie ocenianych przez ogół filmów posłużą jako baza do wypracowania predykcji oceny Ingi dla Pitbulla. Ważne są te dwa warunki: obiekty te musiały być ocenione przez Ingę i być podobnie oceniane przez wszystkich do Pitbulla. Oczywiście analogiczny sposób postępowania można by powtórzyć dla innych filmów, których Inga nie widziała i potem zidentyfikować wśród nich te, dla których przewidywana ocena jest najwyższa.

Item-based Collab. Filter. (3) - Making Predictions

Making predictions

- Find a set of items liked similarly by all users and rated by Julia
- Use an aggregate of Julia's ratings on these items to predict if Julia will like Pitbull
- **Often k-NN part is skipped and all items rated by the user are accounted for**

	Wesele	Dom zły	K. mafii	Pitbull
Agata	7	6	3	7
Karolina	7	4	4	6
Natalia	3	7	7	2
Asia	4	4	6	2
Julia	**7**	**4**	3	?
Cosine	**0.98**	**0.84**	0.70	1.00

The average predicted rating

$$predction(r_{a,i}) = \frac{\sum_{p=1,\ldots,K} r_{a,p}}{K}$$

K=1 ⇨ $r_{J,P} = 7$

K=2 ⇨ $r_{J,P} = (7+4)/2 = 5.5$

Closer neighbors count more (most commonly used for item-based CF):

$$predction(r_{a,i}) = \frac{\sum_{p=1,\ldots,K} r_{a,p} \cdot sim(i,a)}{\sum_{p=1,\ldots,K} sim(i,a)}$$

Weighted average for K = 2:

$r_{J,P} = (7 \cdot 0.98 + 4 \cdot 0.84)/(0.98+0.84) = 5.61$

[37] Rozważmy przykład, w którym chcemy dokonać predykcji oceny Ingi dla Pitbulla przy wykorzystaniu algorytmu społecznego filtrowania bazującego na obiektach. W standardowym przypadku funkcja agregacji bazuje na ocenach Ingi dla k najbardziej podobniej ocenianych obiektów do Pitbulla. Najbardziej sensowną miarą, którą można wykorzystać do takiej oceny jest miara kosinusowa. Jej wyniki dla trzech pozostałych filmów w zestawieniu z Pitbullem zaprezentowano w tabeli. Jeśli wykorzystać by K=1, to pod uwagę weźmiemy tylko ocenę Ingi dla Wesela; jeśli K=2, to oceny Ingi dla Wesela i Domu złego. Najczęściej wykorzystywane funkcje agregacji to średnia oraz średnia ważona. Wyniki ich zastosowania zaprezentowano w prawej i dolnej części slajdu. Nie ma sensu stosować sposobu, w którym startowalibyśmy ze średniej ocen Ingi, bo w tym wypadku ocena jest przecież wypracowywana na podstawie jej ocen. W niektórych zastosowaniach nie ogranicza się obiektów, na podstawie których wypracowywana jest rekomendacja do k najbardziej podobnych. Zamiast tego bierze się pod uwagę wszystkie obiekty ocenione do tej pory przez użytkownika, dla którego dokonywana jest predykcja.

Collab. Filter. - Improving Prediction Functions

Not all neighbor ratings might be equally "valuable"

- Agreement on commonly liked items is not so informative as agreement on controversial items
- Possible solution: give more weight to items that have a higher variance

Value of number of co-rated items

- Use "significance weighting", by e.g., linearly reducing the weight when the number of co-rated items is low

Case amplification

- Intuition: give more weight to "very similar" neighbors, i.e., where the similarity value is close to 1

[38] W kontekście wartości wag, które wykorzystywane są w funkcji agregacji, powstał szereg dodatkowych rekomendacji, które można wykorzystywać w praktyce. Można tu zaobserwować trzy podstawowe pola działania. Po pierwsze, istotniejsze wydają się oceny dla obiektów, dla których nie ma pełnej zgody co do ich dobrej lub złej jakości. O różnorodności ocen obiektu świadczy wariancja i właśnie obiektom, dla których wariancja ocen jest wysoka, można by podwyższyć wagę. Po drugie, istotne jest nie tylko podobieństwo, ale też jego wiarygodność. W tym kontekście, im wyższa jest liczba użytkowników, na podstawie których wypracowano miarę podobieństwo, tym bardziej istotne jest to wskazanie. Po trzecie, jeśli podobieństwo dla jakiejś pary obiektów jest bardzo wysokie, to intuicyjnie chcielibyśmy, aby wkład takiej oceny w ostateczną predykcję był bardzo wysoki. Można to uzyskać poprzez dodatkowe podwyższanie wag tych ocen, dla których podobieństwo było maksymalne lub bliskie maksymalnemu.

Collaborative Filtering - Neighborhood

- Neighborhood size limited to a **specific size** or via **similarity threshold**
- Combine the two above options so that nearest-neighbors which are not really similar are neglected
- An analysis of the MovieLens dataset indicates that "*in most real-world situations, a neighborhood of 20 to 50 neighbors seems reasonable*"

Neighborhood size and similirity thresholds can be determined experimentally via tests referring to **error rate measures**

- N – number of tested items
- p_i – predicted rating
- r_i – actual rating

- **Mean Absolute Error** (*MAE*) computes the deviation between predicted ratings and actual ratings

$$MAE = \frac{\sum_{i=1,\ldots,N} |p_i - r_i|}{N}$$

- **Root Mean Square Error** (*RMSE*) is similar to *MAE*, but places more emphasis on larger deviation

$$RMSE = \sqrt{\frac{\sum_{i=1,\ldots,N} (p_i - r_i)^2}{N}}$$

[39] Ostatnie pytanie, na które musimy odpowiedzi dotyczy rozmiaru sąsiedztwo, tj. liczby najbliższych sąsiadów, którzy są brani pod uwagę przy wypracowaniu rekomendacji. Rozmiar ten może być zdeterminowany przez maksymalny rozmiar podany przez użytkownika, minimalny próg podobieństwa lub połączenie tych dwóch wymagań. Analiza przeprowadzono przez movielens wskazuje, że rozsądny rozmiar dla systemów z dużą bazą danych to kilkadziesiąt. Można jednak potraktować te wartości jako parametry modelu i wybrać najlepsze w sposób eksperymentalny. Do oceny jakości predykcji dla danej wartości parametru służą dwie miary. Pierwszą jest średni błąd bezwzględny (Mean Average Error), który oblicza odchylenie między przewidywaną a rzeczywistą wartością. Drugą jest błąd średniokwadratowy, który jest podobny do MAE, ale kładzie większy nacisk na duże odchylenia między wartością przewidywaną a rzeczywistą poprzez zastosowanie drugiej potęgi.

Slope One Predictor

Idea based on a *popularity differential* between items for users

Karolina rated *K. mafii* with 4 and Pitbull with 6. If Julia rated *K. mafii* with 3, we can make the assumption that the difference between both items will be the same as *for Kar*. With this in mind, *Julia* would rate *Pitbull* with: 3 + (6–4) =5

	K. mafii	Pitbull
Karolina	4	**6**
Julia	3	?

The difference between the ratings for other users determines the bias (6 - 4 = 2)

The rating for other item sets the starting point (3) for the prediction

Take the average of these differences of the co-ratings to make the prediction

	D	K	T
Natalia	2	3	**5**
Asia		4	**3**
Julia	5	2	?

Predict rating for T based on D: 5 + (5-2) = 5 + 3 = 8

Predict rating for T based on K: 2 + [(5-3) + (3-4)]/2 = = 2 + 0.5 = 2.5

Combine predictions using a weighted average with weights equal to the numbers of items used for deriving the prediction

$$r_{I,P} = \frac{8 \cdot 1 + 2.5 \cdot 2}{1 + 2} = 4.33$$

[40] Trzeci najpopularniejszy algorytm predykcji w tej dziedzinie zastosowań to slope one predictor. Jest on podobny do społecznego filtrowania na podstawie obiektów (item-based), ale bazuje na popularności różnicy ocen wystawionych przez użytkowników. Aby go wyjaśnić rozważmy łatwiejszy przykład w górnej części slajdu. Julia oceniła Kobiety mafii na 4, a Pitbulla na 6. Jeśli Inga oceniła Kobiety mafii na 3, to moglibyśmy założyć, że różnica jej ocen dla Pitbulla i Kobiet mafii będzie podobna jak dla Julii, czyli 2. Predykcja oceny Ingi dla Pitbulla byłaby więc równa jej ocenie dla Kobiet mafii, czyli 3, zmienionej o różnicę ocen zaobserwowaną dla Julii, czyli +2. Ostatecznie wyszłoby 5. Czy można ten pomysł uogólnić do większej liczby użytkowników i obiektów? Będziemy działać w trzech krokach. Po pierwsze, wypracujemy predykcję dla Pitbulla na podstawie każdego filmu, który oceniła Inga. W ramach tej predykcji będziemy startować z oceny wystawionej przez Ingę, a więc 5 dla D oraz 2 dla K. Po drugie, tę ocenę będziemy modyfikować o średnią różnicę w ocenach innych użytkowników, a więc 3 dla D oraz 0.5 dla K. Po trzecie, ostatecznej agregacji musimy dokonać, biorąc pod uwagę predykcje wypracowane na podstawie różnych filmów. Tu wykorzystywana jest średnia ważona, gdzie wagi odpowiadają liczbie ocen, na podstawie których obliczono, jak zmienić ocenę Ingi (czyli użytkowników, którzy ocenili obydwa filmy). W naszym przypadku, 1 dla D oraz 2 dla K. Ostatecznie przewidywana ocena jest równa 4.33, a więc jest bliższa estymacji oceny dokonanej na podstawie K niż D, bo K i T widziało razem więcej użytkowników niż D i T, a więc ta predykcja została uznana podczas agregacji za bardziej wiarygodną.

Bayesian Collaborative Filtering (1)

Basic idea (simplistic version)

- Given the user/item rating matrix
- Determine the probability that user will like an item up to a certain degree
- Based the recommendation on such probabilities

Calculation of rating probabilities

- How probable is rating value "1" for Pitbull (T) given Julia's previous ratings?
- Corresponds to conditional probability $P(T=1 \mid X)$, X = Julia's previous ratings = (W=5, D=4, K=3)

	W	D	K	T
Julia	5	4	3	?

Can be estimated based on Bayes Theorem for Y = (T=some rating)

$$P(Y|X) = \frac{P(X|Y) \cdot P(Y)}{P(X)} = \frac{\prod_{i=1,\ldots,N} P(X_i|Y) \cdot P(Y)}{P(X)}$$

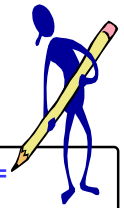
- Assumption: ratings are independent

[41] Ostatni algorytm predykcji jest rzadko wykorzystywany w tym kontekście i bardziej posłuży nam jako powtórzenie sposobu działania klasyfikatora Bayesa. Zadanie będzie więc polegało na obliczeniu prawdopodobieństwa, że użytkownik wystawi konkretną ocenę i wybraniu tej oceny, dla której prawdopodobieństwo jest najwyższe. Przykładowo, jeśli pytamy, jak prawdopodobne jest, że Inga wystawi 1 dla T, to musimy obliczyć prawdopodobieństwo warunkowe wystawienie takiej oceny pod warunkiem, że dany jest profil Ingi, a na profil ten składają się jej oceny, a ściślej rzecz biorąc koniunkcja jej ocen, dla innych filmów. Oczywiście klasyfikator Bayesa opiera się na założeniu o niezależności ocen, co w kontekście predykcji jest jeszcze bardziej kontrowersyjne niż w klasyfikacji.

Bayesian Collaborative Filtering (2)

	W	D	K	T
Agata	5	2	3	5
Karolina	3	4	4	1
Natalia	5	5	3	2
Asia	4	4	3	2
Julia	5	4	3	?

X = (W=5, D=4, K=3)



$P(T=1|X) \approx P(T=1) \cdot P(W=5|T=1) \cdot P(D=4|T=1) \cdot P(K=3|T=1) =$
$= 1/4 \cdot 0 \cdot 1/1 \cdot 0 = 0$
$P(T=2|X) \approx P(T=2) \cdot P(W=5|T=2) \cdot P(D=4|T=2) \cdot P(K=3|T=2) =$
$= 2/4 \cdot 1/2 \cdot 1/2 \cdot 2/2 = 1/8$
$P(T=3|X) \approx P(T=3) \cdot P(W=5|T=3) \cdot P(D=4|T=3) \cdot P(K=3|T=3) = 0$ ...

More to consider

- Zeros (smooting required)
- Like/dislike simplification possible

[42] Przykład, który jest analizowany ma zmienione oceny w stosunku do poprzednich slajdów tak, by lepiej zilustrować działanie Bayesowskiego społecznego filtrowania. Traktujemy każdą możliwą ocenę (np. 1 lub 2 lub 3 ?) jako potencjalną klasę i obliczamy dla niej prawdopodobieństwo warunkowe. W tym celu uwzględniamy prawdopodobieństwo wystąpienia tej ocen ydla tego konkretnego obiektu, analizując innych użytkowników. Przykładowo, prawdopodobieństwo wystąpienia oceny 1 dla T wynosi 1/4, bo 1 z 4 użytkowników, którzy ocenę dla T wystawili, dał 1. Takie prawdopodobieństwo mnożone jest przez iloczyn prawdopodobieństw warunkowych wystąpień ocen Ingi dla innych obiektów dla innych użytkowników pod warunkiem wystąpienia danej oceny dla obiektu, dla którego dokonujemy predykcji. Przykładowo, prawdopodobieństwo warunkowe wystąpienia oceny 5 dla W pod warunkiem, że dla T wystawiono 1 wynosi 0, bo 0 z 1 użytkowników, którzy wystawili 1 dla T ma 5 dla W. Analogicznie prawdopodobieństwo warunkowe wystąpienia oceny 5 dla W pod warunkiem, że dla T wystawiono 2 wynosi 1/2, bo 1 z 2 użytkowników, którzy wystawili 2 dla T ma 5 dla W. Ostatecznie najwyższe prawdopodobieństwo wyszło dla oceny 2 i to ona zostałaby zwrócona przez algorytm.

Collaborative Filtering - Summary

Pros

- ☺ Well-understood
- ☺ Easy to implement
- ☺ Context-independent
- ☺ More accurate compared to content-based techniques
- ☺ Proved to work well in some domains (where ratings are commonly used)

Cons

- ☹ Sparsity: the percentage of users who rate items is low
- ☹ Requires user community
- ☹ Scalability: cost of finding K neighbors
- ☹ Cold-start: new users will have little information to be compared with other users
- ☹ New item: lack of ratings to create a solid ranking
- ☹ Prune to manipulations (shilling, push attack, nuke, …)
- ☹ No integration of other knowledge sources

[43] Podsumowując zagadnienie społecznego filtrowania, wskażmy na jego podstawowe zalety i wady. Do tych pierwszych bez wątpienia zaliczyć trzeba łatwość zrozumienia, prostotę algorytmów, ich niezależność od kontekstu zastosowania, dokładność zwracanych wyników oraz potwierdzoną skuteczność w wielu rzeczywistych serwisach rekomendacyjnych. O wadach takich algorytmów powstają jednak osobne książki. W szczególności wskazuje się na potrzebę posiadania szerokiej bazy użytkowników, problemy ze skalowalnością, brak wiarygodności ocen dla nowych użytkowników i obiektów oraz brak integracji innych źródeł wiedzy niż same oceny. Najciekawsza i najszerzej studiowana z wad dotyczy jednak sposobów manipulacji takimi algorytmami, aby podwyższyć lub obniżyć predykcję dla konkretnego obiektu. Dobrze zdefiniowanych technik są tu dziesiątki i różnią się one nie tylko celem, na który są zorientowane, ale też fazą funkcjonowania systemy (czy jest to macierz rzadka czy gęsta; czy film jest nowy czy stary; czy użytkownik ocenił wiele czy mało filmów, itd.).

Summary (1)

Given the following similarities of Y with D1-D10 and classes A, B, and C:

	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10		**Y**
Class	A	C	B	C	B	A	A	C	A	B		**?**
Similarity with X	0.9	0.8	0.6	0.5	0.5	0.4	0.4	0.2	0.1	0.0		**1.0**

use 4-NN to provide a classification for Y :

I) … using either majority rule voting or weighted voting?

II) Would the results be different in case of using either k=3 or k=5?

III) What should be the minimal k so that A is recommended?

[44] I)
II)
III)

Summary (2)

Given the following collection of documents and assuming they are represented with the binary vector of terms, use the Naive Bayes classifier to classify D6:

	Doc	Content	Class
training	D1	T1 T2	A
training	D2	T3 T3	A
training	D3	T1 T3	A
training	D4	T1 T2 T2	B
training	D5	T1	B
test	**D6**	T1 T3 T3	?

I) What is the binary represenation of D5?

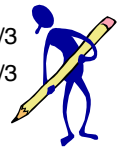
II) What is P(A)?

III) What is P(T3=1|A)? If 0, assume 0.1 as a simple smoothing technique.

IV) What class would be recommended for Y and why (compare P(A|D6) and P(B|D6))?

Doc	T1	T2	T3	Class
D1	1	1	0	A
D2	0	0	1	A
D3	1	0	1	A
D4	1	1	0	B
D5	?	?	?	B
D6	1	0	1	?

P(A) = ?
P(T1=1|A) = 2/3
P(T2=0|A) = 2/3
P(T3=1|A) = ?
P(A|D6) ≈ ?



P(B) = 2/5
P(T1=1|B) = 1
P(T2=0|B) = 1/2
P(T3=1|B) = 0 ≈ **0.1**
P(B|D6) ≈ 0.02

[45] I)
II)
III)
IV)

Summary (3)

Given the following collection of documents and assuming they are represented with the number of terms T1-T3, use the Naive Bayes classifier to classify D6:

	Doc	Content	Class
training	D1	T1 T2	A
training	D2	T3 T3	A
training	D3	T1 T3	A
training	D4	T1 T2 T2	B
training	D5	T1	B
test	**D6**	T1 T3 T3	?

I) What is the size of the dictionary?

II) What is P(B)?

III) What is P(T3|B) while using add-one smooting technique?

IV) What class would be recommended for Y and why (compare P(A|D6) and P(B|D6))?

P(A) = **3**/5

P(T1|A) = (2+1)/(**6**+3) = 3/9

P(T3|A) = (3+1)/(**6**+3) = 4/9

P(A|D6) $\approx$ 3/5·3/9·$(4/9)^2$ = 0.039

P(B) = **?**

P(T1|B) = (2+1)/(4+3) = 3/4

P(T3|B) = **?**

P(B|D6) $\approx$ **?**

[46] I)
II)
III)
IV)

Summary (4)

Given the following user-item rating matrix, predict rating of user U7 for item I4:

	I1	**I2**	I3	**I4**
U1	5	**4**	4	**4**
U2	5	**3**	7	**3**
U3	4	**3**	2	**3**
U4	6	**4**	5	**4**
U5	3	**4**	2	**4**
U6	4	**3**	5	**3**
U7	4	**3**	5	**?**

sim(U7,U·)
0.0
1.0
-0.5
0.5
-1.0
1.0

User-based CF with k=2 and simple average:

U7(I4) = (3+3)/2 = **?**

… or weighted average:

U7(I4) = (1·3 + 1·3)/(1+1) = **?**

I) Employ user-based CF with k=2 and either simple average or weighted average?

II) How the results change when using k=3? Does it make sense to account for k=4?

III) Employ user-based CF with k=2 and modify U7's average rating by the weighted modification of its nearest neigbors averages:

U7(I4) = 4 + (1·(3-5) + 1·(…))/(1+1) = **?**

IV) Which item should be analyzed to predict the rating when using item-based CF with k=1? What would be the predicted rating?

[47] I)
II)
III)