Maschinelles Lernen
- Informatik -

Rapp, DHBW Lörrach

19.01.2024

Inhaltsübersicht

Prüfungsaufgabe 2

Empfehlung von Kunstwerken mit Matrix-Faktorisierung im Content- und Collaborative Filtering und Kundenprofilanreicherung

a) Übertragen Sie die Tracking Daten (Sample_Tracking_Data_for_Recommender_System_ABCD.csv) in das User-Item Interaction Matrix Template (User-Item_Interaction_-Matrix_Template_With_Randomized_Values.csv).
   - Tipp: Definieren Sie den zeitlichen Rahmen für die Verweildauer auf einem Kunstwerk so, dass diese als implizites Feedback interpretiert werden kann.
   - Tipp: Um bessere Ergebnisse zu erzielen, können z.B. weitere Tracking-Profile aus synthetischen Daten erzeugt werden.

b) Optimieren Sie relevante Parameter (z.B. von bm25_weight, AlternatingLeastSquares) in der Code-Vorlage so, dass beim Aufruf der Funktionen user_recommend() und artwork_recommend() sinnvolle Empfehlungen von Kunstwerken ausgesprochen werden. Begründen Sie jeweils Ihre Entscheidungen.

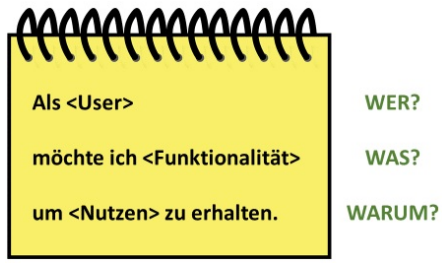
c) Implementieren Sie ein Python-Programm zur textbasierten Anreicherung der Kundenprofile (s. Spalte BESCHREIBUNG) in einem Customer Relationship Management System einschl. einfachem Flask Web-App User Interface.

Rahmen Beispiele für Bewertungskriterien

- Die Empfehlung von Kunstwerken ist sinnvoll und kann leicht interpretiert werden.
- Die Ergebnisqualität der Empfehlungen kann aus dem Programmcode nachvollzogen werden.
- Die Anreicherungen der Kundenprofile ist sinnvoll und ermöglicht eine zielgerichtete Kundenansprache.
- Aus dem Programmcode ist ersichtlich, dass die wesentlichen Konzepte aus der Vorlesung verstanden und auf die konkrete Problemstellung angewendet wurden.
- ...

User Story Beispiele einer Online-Plattform

User Story Beispiele aus Sicht von **Nutzer, Retailer** und **Plattform-Anbieter** für die Nutzung einer hypothetischen Online-Plattform für Marken-Klamotten:



User Story Template

1. Als **Nutzer** der Online-Plattform möchte ich über Marken-Klamotten informiert werden, die auf meine individuellen Bedürfnisse zugeschnitten sind.
2. Als **Retailer** möchte ich meinen Kunden zielgruppenorientiert Neuheiten aus meinem Produktsortiment anbieten, um meinen Umsatz zu steigern.
3. Als **Produktplattform-Anbieter** möchte ich Endkunden und Retailer möglichst effizient miteinander vernetzen, um meine Verkaufsprovision zu steigern.

Warum Empfehlungen?

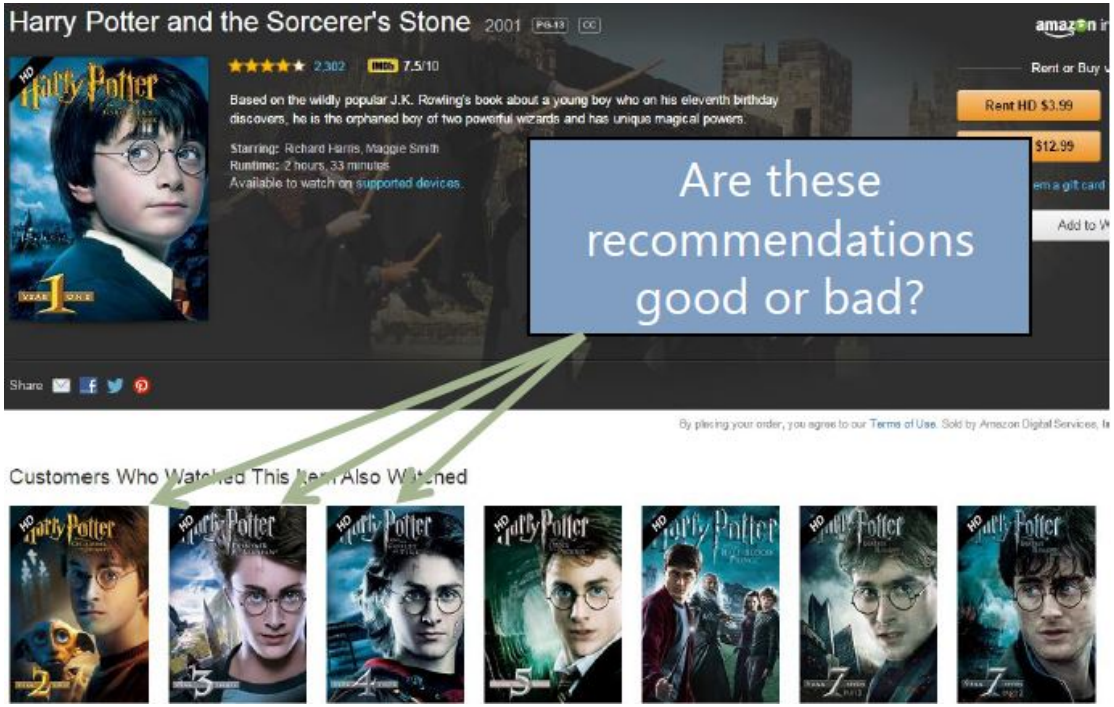
1. Bereitstellung von neuem Content

User-Story Beispiel

Als Nutzer einer Online-Plattform möchte ich Neuheiten entdecken, damit ich nichts mehr verpasse.

Warum Empfehlungen?

2. Bereitstellung von Content, den ich bereits gesucht habe

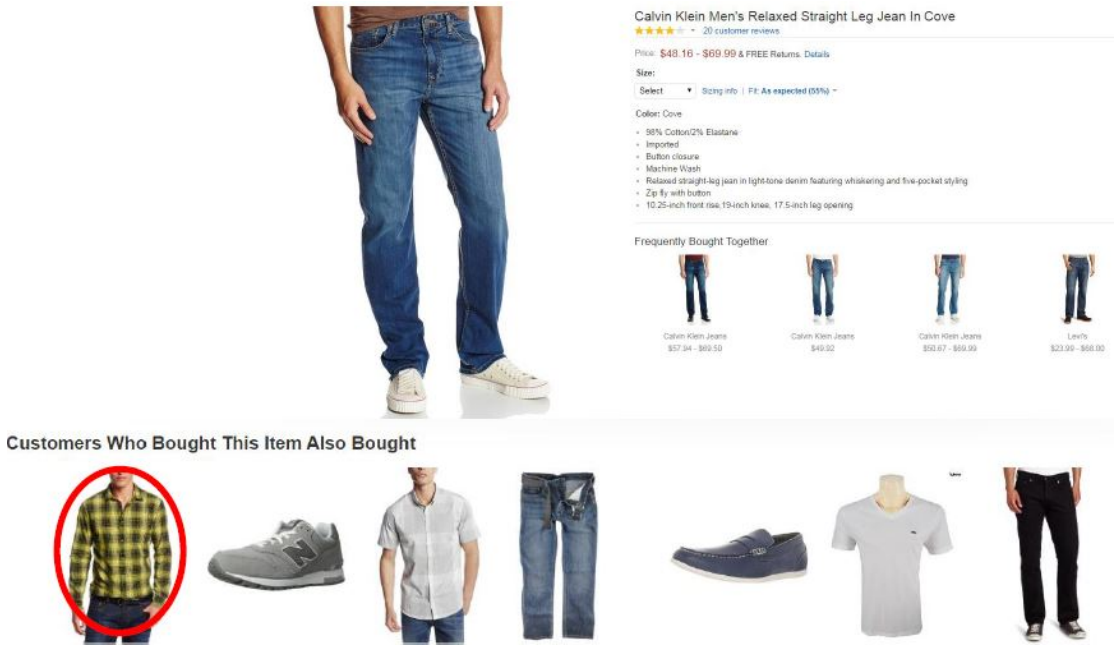


User-Story Beispiel

Als Nutzer einer Online-Plattform möchte ich passende Inhalte zu meinen bereits vorhandenen Suchkriterien finden, um eine effiziente Auswahl treffen zu können.

Warum Empfehlungen?

3. Bereitstellung von Content, der zusammenpasst

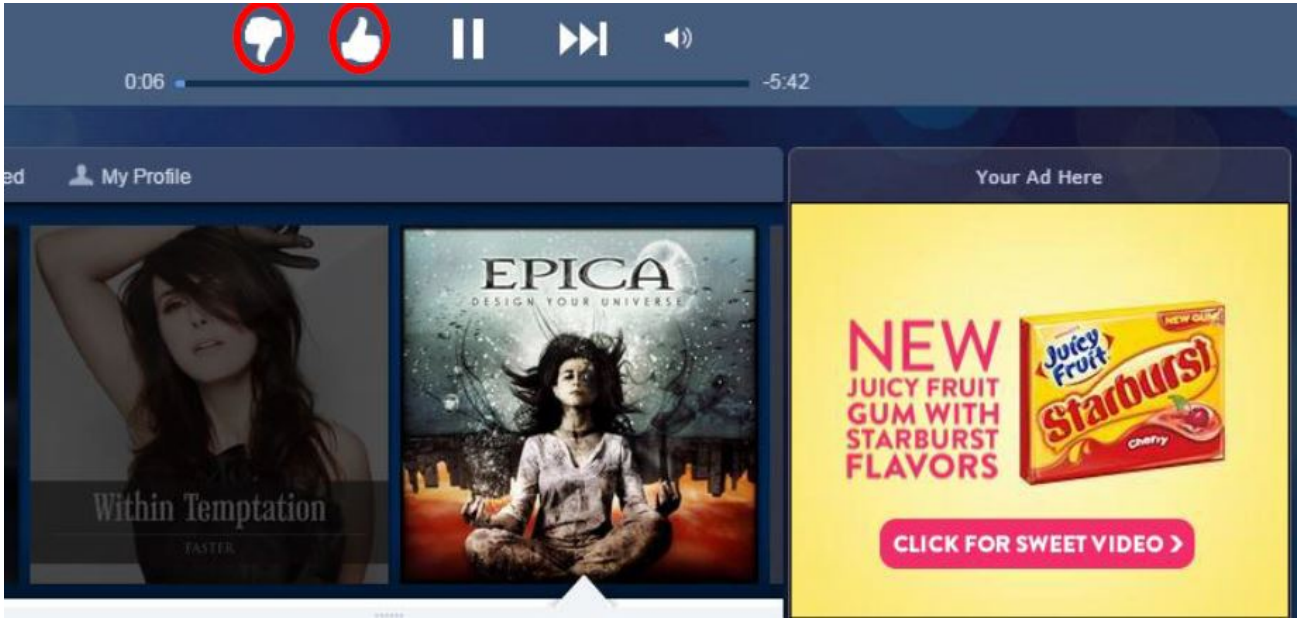


User-Story Beispiel

Als Nutzer einer Online-Plattform möchte ich zusätzlich zu meiner Auswahl weitere passende Angebote erhalten, die meine Auswahl vervollständigen.

Warum Empfehlungen?

4. Bereitstellung von personalisiertem Content

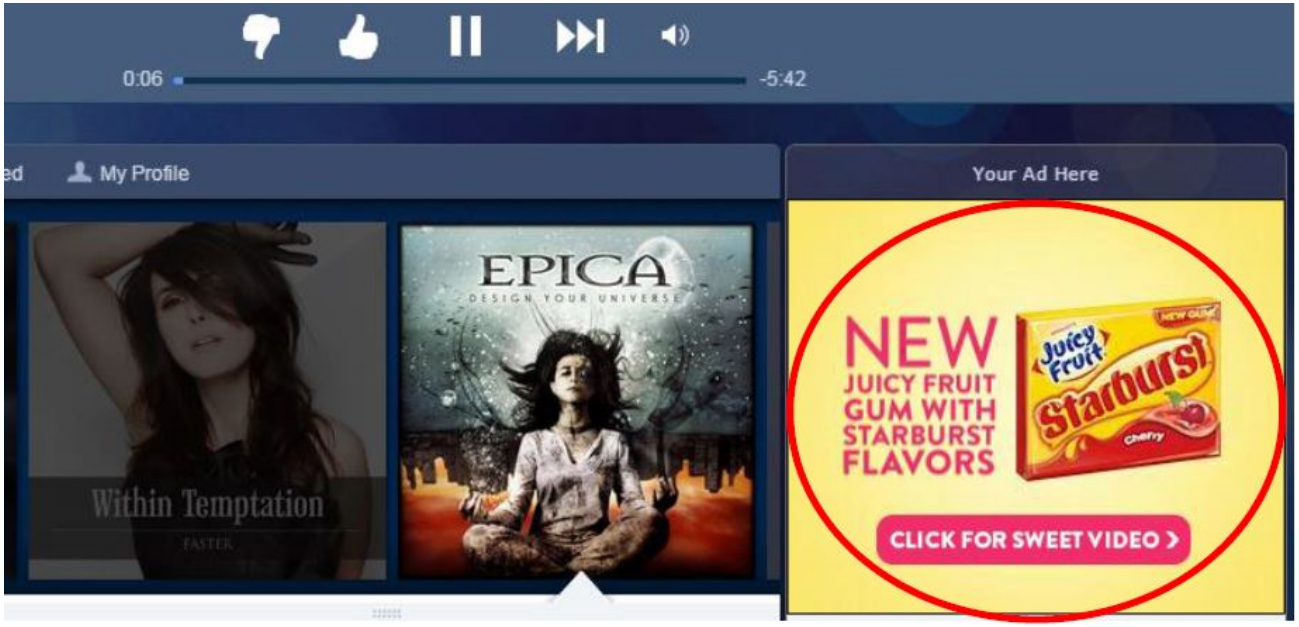


User-Story Beispiel

Als Nutzer einer Online-Plattform möchte ich, dass mein Feedback in zukünftige Angebote einfließt, um die Passgenauigkeit auf meine individuellen Bedürfnisse zu steigern.

Warum Empfehlungen?

5. Bereitstellung von relevantem Content



User-Story Beispiel

Als Nutzer einer Online-Plattform möchte ich Produkte angezeigt bekommen, die für meine Interessen relevant sind, um mein Nutzererlebnis noch intensiver zu gestalten.

Warum Empfehlungen?

6. Bereitstellung von Content, der mir gefällt

User-Story Beispiel

Als Nutzer einer Online-Plattform möchte ich Produkte angezeigt bekommen, die mir gefallen und Freude bereiten.

Zusammenfassung und Vorgehen

Zusammenfassung

Bereitstellung von

- neuem Content
- Content, den wir bereits gesucht haben
- Content, der zusammenpasst
- personalisiertem Content
- relevantem Content
- Content, der uns gefällt

Vorgehen

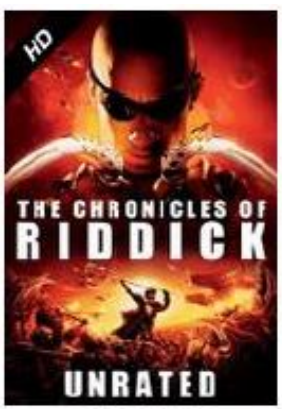
Wir modellieren **Nutzerinteressen, -meinungen** und **-verhalten.**

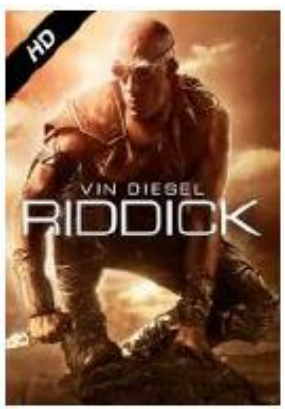
Aufgabenbeispiel Filmempfehlungen

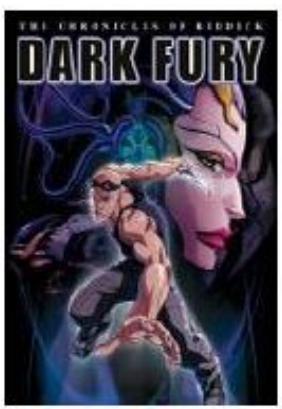
Nehmen Sie an, Sie wollen ein Recommender System für Filme entwickeln.

Zentrale Fragestellung

Welche der folgenden Filme würde ich am besten bewerten?







„ML Dreiklang“ für Netflix-Preis

Daten

Mit dem jeweiligen Rating gelabelte Daten wurden 2006 bereitgestellt:

User ID	Movie ID	Date	Rating
674439	2425	01.07.2005	3

Beispiel-Record aus Netflix Trainingsdatensatz mit ca. 500.000 Netflix-Nutzern

Ziel

Vorhersage der Bewertung *(„Rating“)*, die ein Nutzer einem Film geben würde.

Leistungsmaß/Metrik

$$RMSE(f) = \sqrt{\frac{1}{N} \sum_{u,i,t \in testset} (f(u,i,t) - r_{u,i,t})^2}$$, Root-Mean-Square-Error

$f(u,i,t)$: Vorhersage des Modells; $r_{u,i,t}$: Ground-Truth

Preisausschreibung

Das Team, das den RMSE zuerst um **10 Prozent** im Vergleich zur Netflix-Lösung reduziert, erhählt **USD 1.000.000 !!!**

Recommender System

Charakterisierung Recommender Problem

Ziel

Vohersage der Bewertung eines Items durch einen Nutzer.

Prediction Function

Finde p für die **Vorhersage des User-Nutzens,** sodass

$$p : u, i \mapsto \hat{r}_{ui}$$
$$\mathcal{U} \times \mathcal{I} \rightarrow \mathcal{S}$$

mit $u \in \mathcal{U} = \{u_1, ..., u_m\}$ User, $i \in \mathcal{I} = \{i_1, ..., i_n\}$ Item, $\hat{r}_{ui} \in \mathcal{S}$ fehlende Ratings aus der Menge $\{s_1, ..., s_q\}$ möglicher Ratings (z.B. $\mathcal{S} = [1, 5]$).

Output

- Top-k (z.B. 10) Item Liste als Empfehlungen für den Nutzer
- Sortierreihenfolge: absteigend entsprechend Prediction Function

Beispiele für Nutzenmessung

- Nachfolgeaktionen wie z.B. *bought, not interested, not show again*
- Implizites Nutzer-Feedback: Verweildauer auf Produktseiten

Herausforderungen in der Praxis

Herausforderungen

- Personalisierung setzt entsprechende Datenqualität voraus, die oftmals nicht vorhanden ist
- Unbekannte Bewertungen müssen aus den ggf. wenig vorhandenen bekannten extrapoliert werden
- Nutzer-Feedback oft nicht explizit, sondern nur implizit in Form von z.B. „Anzahl wiederkehrender Produktseitenaufrufe“ vorhanden

Arten von Recommender Systemen

1. **Inhaltsbasierter Empfehlungsdienst („Content-based filtering“)**
   - Empfehlung von Items, die ähnlich den Items sind, die der Benutzer bereits hoch bewertet hat.
   - Dazu ist es erforderlich, die Ähnlichkeit zwischen zwei Items durch z.B. Features bestimmen zu können.
2. **Kollaborativer Empfehlungsdienst („Collaborative filtering“)**
   - Empfehlung von Items, an denen Nutzer mit ähnlichem Bewertungsverhalten *(ähnliche Nutzer)* das größte Interesse haben.
   - Dazu müssen keine weiteren Kenntnisse über das Item selber vorhanden sein.
3. **Latent-Factor-basiert (Matrix Factorization)**
   - Erzielt eine Empfehlung durch die Projektion von Nutzern und Items in einen niedrig-dimensionalen latenten Raum *(Simon Funk, 2006)*.

Content-Based Recommender System

Nutzenmatrix

Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)	x_1 (romance)	x_2 (action)
Love at last	5	5	0	0	0.9	0
Romance forever	5	?	?	0	1.0	0.01
Cute puppies of love	?	4	0	?	0.99	0
Nonstop car chases	0	0	5	4	0.1	1.0
Swords vs. karate	0	0	5	?	0	0.9

Nutzenmatrix *für Vorhersage von* **Content-Based** *Film-Ratings von 0 bis 5 Sternen*

Gegeben ist der **Item Feature Vektor,** der sich aus den Eigenschaften des Items i zusammensetzt.
Gesucht ist der **User Feature Vektor,** der das Nutzerprofil darstellt und die Präferenzen und Vorlieben für z.B. Genres und Autoren beschreibt.

Feature-Vektoren

Item: $x^{(i)} = \begin{bmatrix} x_0^{(i)} \\ x_1^{(i)} \\ x_2^{(i)} \\ \dots \end{bmatrix} \overset{i=3}{=} \begin{bmatrix} 1 \\ 0.99 \\ 0 \end{bmatrix}$ **User:** $\theta^{(j)} = \begin{bmatrix} \theta_0^{(j)} \\ \theta_1^{(j)} \\ \theta_2^{(j)} \\ \dots \end{bmatrix} \overset{j=1,\text{Ann.: berechnet}}{=} \begin{bmatrix} 0 \\ 5 \\ 0 \end{bmatrix}$

mit $x_0^{(i)} = 1$ und $\theta_0^{(j)} = 0$ für die Anwendung linearer Regression.

Ziel und Rating-Vorhersage

Ziel

Lerne den **User Feature Vektor** $\theta^{(j)} \in \mathbb{R}^{n+1}$ *(im Beispiel: $n = 2$ Features)* für jeden User j.

Beispiel

Mit gelerntem $\theta^{(1)} = \begin{bmatrix} 0 \\ 5 \\ 0 \end{bmatrix}$ für *Alice* wäre die **Rating-Vorhersage** des Filmes *Cute puppies of love* mit $x^{(3)} = \begin{bmatrix} 1 \\ 0.99 \\ 0 \end{bmatrix}$ durch *Alice* durch **lineare Regression** gegeben:

$$\begin{aligned} \hat{r}_{ji} &= (\theta^{(j)})^T x^{(i)} \\ &= (\theta^{(1)})^T x^{(3)} = 4.95 \text{ Sterne.} \end{aligned}$$

Optimierungsalgorithmus: Gradient Descent

Lerne Parameter $\theta^{(j)}$ des Users j für alle n_u User:

$$\min_{\theta^{(1)},\dots,\theta^{(n_u)}} \frac{1}{2}\sum_{j=1}^{n_u}\sum_{i:r(i,j)=1}\left((\theta^{(j)})^T x^{(i)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2}\sum_{j=1}^{n_u}\sum_{k=1}^{n}(\theta_k^{(j)})^2$$

$$= \min_{\theta^{(1)},\dots,\theta^{(n_u)}} (\text{'Fehler'} + \text{'Länge'}) = \min_{\theta^{(1)},\dots,\theta^{(n_u)}} J(\theta^{(1)},\dots,\theta^{(n_u)}) \text{ , J: Zielfunktion}$$

Gradient Descent Update

$$\theta_k^{(j)} := \theta_k^{(j)} - \alpha\left(\sum_{i:r(i,j)=1}((\theta^{(j)})^T x^{(i)} - y^{(i,j)})x_k^{(i)} + \lambda\theta_k^{(j)}\right) \text{ (for } k \neq 0)$$

$$= \theta_k^{(j)} - \alpha\frac{\partial}{\partial\theta_k^{(j)}} J(\theta^{(1)},\dots,\theta^{(n_u)}) \quad , \quad \alpha\text{: Lernrate}$$

Regularisierung gegen Überanpassung

Regularisierung

Auswahl der gewünschten Komplexität des statistischen Modells, sodass die Vorhersagefähigkeit bzw. Generalisierbarkeit verbessert wird.

Ohne Regularisierung: Modell kann...

- **zu komplex** werden und die Daten überanpassen oder
- **zu einfach** werden und die Daten unteranpassen

In beiden Fällen hätte das Modell nur eine **geringe Generalisierbarkeit.**

Überanpassung (Overfitting)

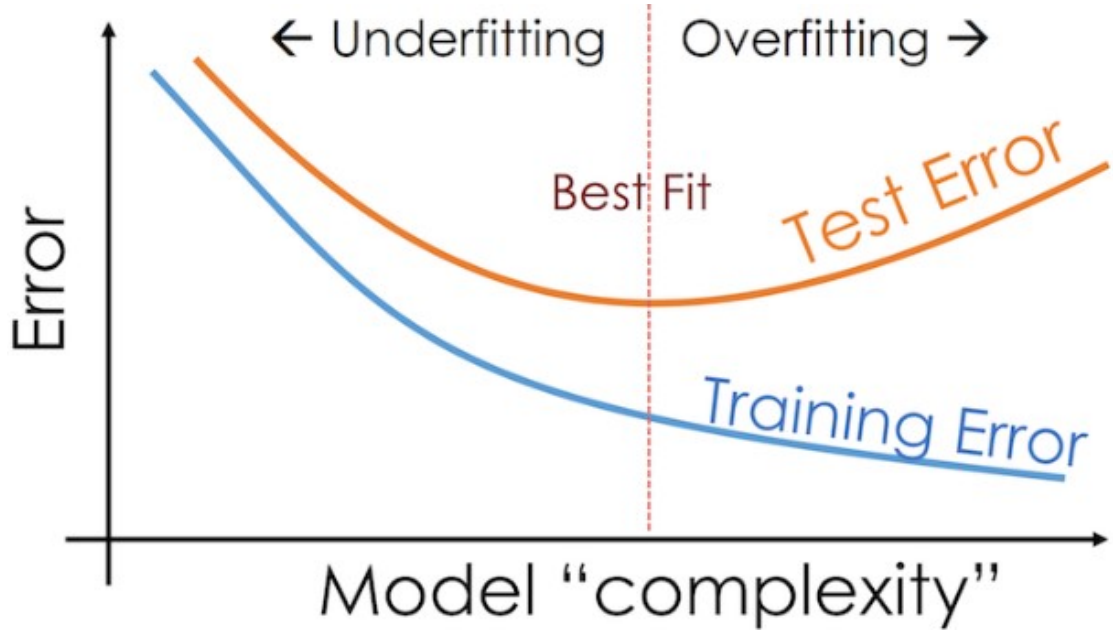
Komplexes Modell mit vielen Freiheitsgraden (freien Parametern) passt sich zu stark an die wenigen vorhandenen Trainingsdatensätze an und...

- ...beginnt Rauschen zu beschreiben
- ...kann nicht mehr auf Testdaten generalisieren

Beispiel: Modell wird mit der Methode der kleinsten Quadrate für einen relativ kleinen Datensatz angepasst $\Rightarrow$ Überanpassung.

Regularisierung versucht dieser Überanpassung entgegenzuwirken, indem eine hohe Komplexität des Modells penalisiert, das heißt bestraft wird.

Modellkomplexität



Auf der linken Seite ist das Modell zu einfach (⇒Underfitting) und auf der rechten zu komplex (⇒Overfitting)
Aus: Principles and Techniques of Data Science, samlau.me

Steuerung der Modellkomplexität durch Regularisierung

Z.B. mehr Rahmenbedinungen durch Regularisierungsterm
⇒ Geringere Modellkomplexität
⇒ verhindert Überanpassung

Collaborative Filtering

Beispiel: Amazon

e.g.:
Amazon

Calvin Klein Men's Relaxed Straight Leg Jean In Cove

20 customer reviews

Price: $48.16 - $69.99 & FREE Returns. Details

Size:

Select | Sizing info | Fit: As expected (55%)

Color: Cove

- 98% Cotton/2% Elastane
- Imported
- Button closure
- Machine Wash
- Relaxed straight-leg jean in light-tone denim featuring whiskering and five-pocket styling
- Zip fly with button
- 10.25-inch front rise,19-inch knee, 17.5-inch leg opening

Frequently Bought Together

Customers Who Viewed This Item Also Viewed

Customers Who Bought This Item Also Bought

Collaborative Filtering: Zentrale Fragestellungen

F: Wie können wir die **Ähnlichkeit** zwischen zwei **Usern** messen?
A: Durch die **Items**, die sie gekauft haben!

F: Wie können wir die Ähnlichkeit zwischen zwei **Items** messen?
A: Durch die **User**, die sie gekauft haben!

Collaborative Filtering = „People-To-People Correlation“

Aus: Schafer et al., Data Mining and Knowledge Discovery, 2001

Collaborative Filtering: Idee

Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)	x_1 (romance)	x_2 (action)
Love at last	5	5	0	0	0.9	0
Romance forever	5	?	?	0	1.0	0.01
Cute puppies of love	?	4	0	?	0.99	0
Nonstop car chases	0	0	5	4	0.1	1.0
Swords vs. karate	0	0	5	?	0	0.9

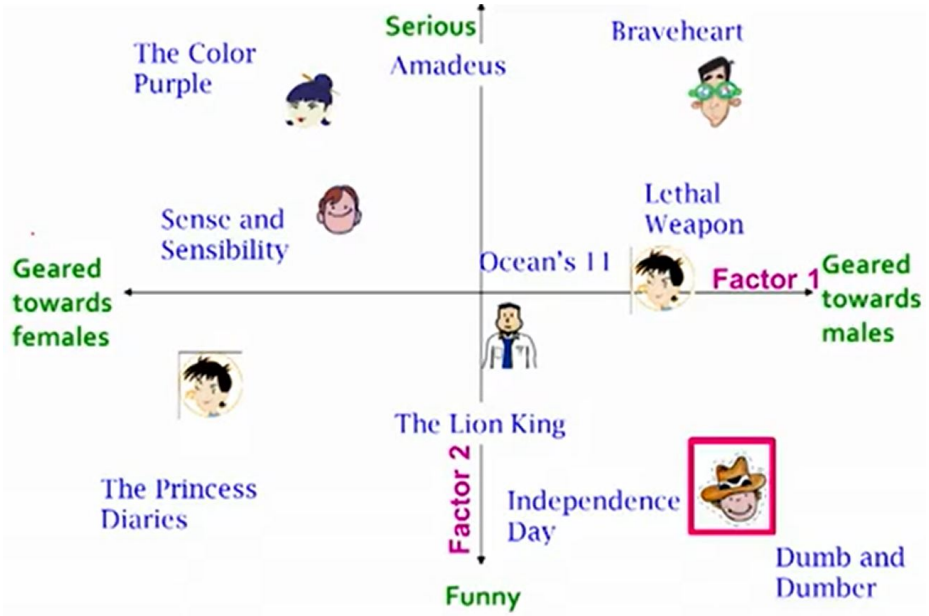
Latent Factor Model

- Matrix Faktorisierung durch z.B. Singulärwertzerlegung (engl. Singular Value Decomposition, „SVD“)

Idee

- Transformation der Items und Nutzer in denselben Raum latenter Faktoren
- Reduktion der Dimensionalität durch Lernen der relevanten Features
- User-Präferenzen und Item-Eigenschaften werden in niedrigdimensionalen Vektorräumen dargestellt.

Collaborative Filtering: Visualisierung



Gegeben: n Datenpunkte (Featurevektoren) $X = \{x_1, ..., x_n\}$ mit Dimensionalität d

Gesucht: Transformation der Datenpunkte in $(d - k)$-dimensionale Featurevektoren, so dass der dabei gemachte Fehler möglichst klein ist

Latente Faktoren: Variationsachsen *(hier: Factor 1 und 2)* z.B. bezogen auf Frauen/Männer und lustig/ernst

Example: Predict Movie Ratings

Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)	x_1 (romance)	x_2 (action)
Love at last	5	5	0	0	?	?
Romance forever	5	?	?	0	?	?
Cute puppies of love	?	4	0	?	?	?
Nonstop car chases	0	0	5	4	?	?
Swords vs. karate	0	0	5	?	?	?

Angenommen, die Feature Vektoren $\theta^{(1)}, ..., \theta^{(n_u)}$ und $x^{(1)}, ..., x^{(n_u)}$ wurden durch einen Algorithmus gelernt, dann lassen sich die Bewertungen beliebiger Items und User vorhersagen:

Beispiele

$$\theta^{(1)} = \begin{bmatrix} 0 \\ 5 \\ 0 \end{bmatrix}, \ \theta^{(2)} = \begin{bmatrix} 0 \\ 5 \\ 0 \end{bmatrix}, \ \theta^{(3)} = \begin{bmatrix} 0 \\ 0 \\ 5 \end{bmatrix}, \ \theta^{(4)} = \begin{bmatrix} 0 \\ 0 \\ 5 \end{bmatrix}$$

$$(\theta^{(1)})^T x^{(1)} = \begin{bmatrix} 0 & 5 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0.9 \\ 0 \end{bmatrix} \approx 5$$ → *Alice wird „Love at last“ lieben!*

$$(\theta^{(2)})^T x^{(1)} \approx 5, \ (\theta^{(3)})^T x^{(1)} \approx 0, \ (\theta^{(4)})^T x^{(1)} \approx 0$$

Alternating Least Squares *(ALS)* Algorithmus

Gegeben: User Feature Vektoren $\theta^{(1)}, ..., \theta^{(n_u)}$ und Filmbewertungen
Gesucht: Item Feature Vektoren $x^{(1)}, ..., x^{(n_m)}$

Optimiere

$$\min_{x^{(1)},...,x^{(n_m)}} \frac{1}{2} \sum_{i=1}^{n_m} \sum_{j:r(i,j)=1} ((\theta^{(j)})^T x^{(i)} - y^{(i,j)})^2 + \frac{\lambda}{2} \sum_{i=1}^{n_m} \sum_{k=1}^{n} (x_k^{(i)})^2$$

Gegeben: Item Feature Vektoren $x^{(1)}, ..., x^{(n_m)}$ und Filmbewertungen
Gesucht: User Feature Vektoren $\theta^{(1)}, ..., \theta^{(n_u)}$, *optimiere analog wie oben*

Ansatz

$$\theta \rightarrow x \rightarrow \theta \rightarrow x \rightarrow \theta \rightarrow x \rightarrow ...$$

$\Rightarrow$ User und Item Feature Vektoren alternierend:

Optimiere

$$\min_{x^{(1)},...,x^{(n_m)},\theta^{(1)},...,\theta^{(n_u)}} J(x^{(1)}, ..., x^{(n_m)}, \theta^{(1)}, ..., \theta^{(n_u)})$$

$$J = \frac{1}{2} \sum_{(i,j):r(i,j)=1} ((\theta^{(j)})^T x^{(i)} - y^{(i,j)})^2 + \frac{\lambda}{2} \sum_{i=1}^{n_m} \sum_{k=1}^{n} (x_k^{(i)})^2 + \frac{\lambda}{2} \sum_{j=1}^{n_u} \sum_{k=1}^{n} (\theta_k^{(j)})^2$$

Zusammenfassung ALS Vorgehen und Ratingvorhersage

1. Initialisiere $x^{(1)}, ..., x^{(n_m)}, \theta^{(1)}, ..., \theta^{(n_u)}$ mit kleinen zufälligen Werten
2. Minimiere $J(x^{(1)}, ..., x^{(n_m)}, \theta^{(1)}, ..., \theta^{(n_u)})$ mit z.B. Gradient Descent:

$$x_k^{(i)} := x_k^{(i)} - \alpha\Big(\sum_{j:r(i,j)=1} ((\theta^{(j)})^T x^{(i)} - y^{(i,j)})\theta_k^{(j)} + \lambda x_k^{(i)}\Big)$$

$$\theta_k^{(j)} := \theta_k^{(j)} - \alpha\Big(\sum_{i:r(i,j)=1} ((\theta^{(j)})^T x^{(i)} - y^{(i,j)})x_k^{(i)} + \lambda \theta_k^{(j)}\Big)$$

$$j = 1, ..., n_u, i = 1, ..., n_m$$

Vorhersagematrix des Sterne-Ratings R

$$R = \Theta^T X = \begin{pmatrix} (\theta^{(1)})^T \\ \vdots \\ (\theta^{(n_u)})^T \end{pmatrix} \begin{pmatrix} (x^{(1)}) & \dots & (x^{(n_m)}) \end{pmatrix}$$

$$= \begin{bmatrix} (\theta^{(1)})^T(x^{(1)}) & (\theta^{(2)})^T(x^{(1)}) & \dots & (\theta^{(n_u)})^T(x^{(1)}) \\ (\theta^{(1)})^T(x^{(2)}) & (\theta^{(2)})^T(x^{(2)}) & \dots & (\theta^{(n_u)})^T(x^{(2)}) \\ \vdots & \vdots & \vdots & \vdots \\ (\theta^{(1)})^T(x^{(n_m)}) & (\theta^{(2)})^T(x^{(n_m)}) & \dots & (\theta^{(n_u)})^T(x^{(n_m)}) \end{bmatrix}$$

$$= \begin{bmatrix} 5 & 5 & 0 & 0 \\ 5 & 0*1+5*0.01+0*0 & 0*1+0*0.01+5*0 & 0 \\ 0*1+5*0.99+0*0 & 4 & 0 & 0*1+0*0.99+5*0 \\ 0 & 0 & 5 & 4 \\ 0 & 0 & 5 & 0 \end{bmatrix} = \begin{bmatrix} 5 & 5 & 0 & 0 \\ 5 & 0 & 0 & 0 \\ 5 & 4 & 0 & 0 \\ 0 & 0 & 5 & 4 \\ 0 & 0 & 5 & 0 \end{bmatrix}$$

Liebe Alice, wir empfehlen dir als nächstes „Cute puppies of love“!

Content-Based Filtering mit latenten Faktoren

Distanzmaß für Filmempfehlungen

Distanzmaß für Content-basierte Filmempfehlungen im vorherigen Beispiel „Movie Rating Predictions“

Für jedes Item i (z.B. Film) lernen wir den **Feature Vektor** $x^{(i)} \in \mathbb{R}^n$.

- Beispiele: $x_1 =$ romance, $x_2 =$ action, $x_3 =$ comedy, $x_4 = \ldots$

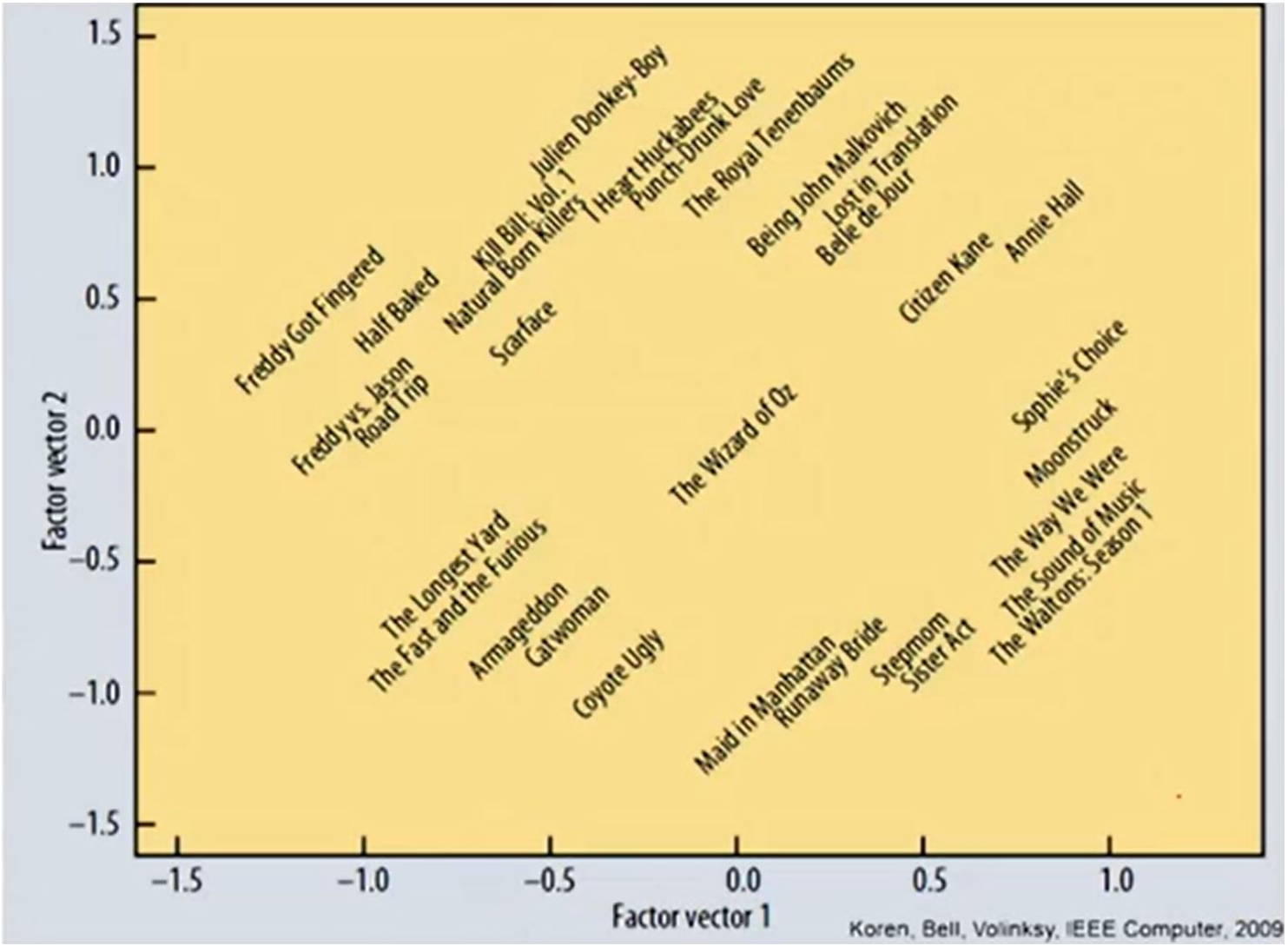
Frage: Wie finden wir Filme j, die *'ähnlich'* zu Film i sind?

- Wenn $||x^{(i)} - x^{(j)}||$ (Norm: z.B. Euklidische Distanz) klein ist, dann sind sich die Filme j und i ähnlich.

Frage: Wie finden wir die 5 Filme, die am ähnlichsten zum Film i sind?

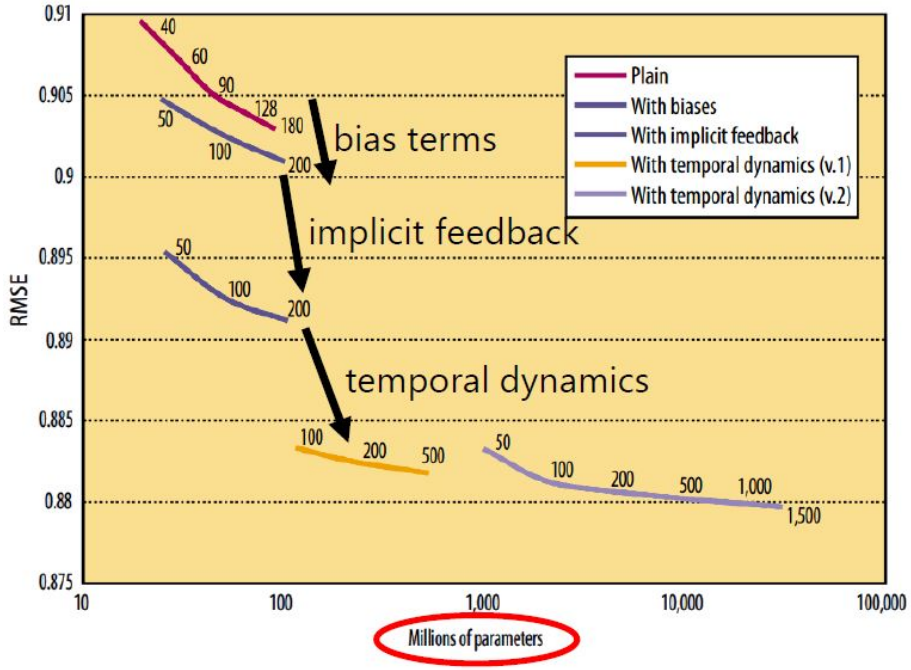
- Finde die 5 Filme j mit dem kleinsten $||x^{(i)} - x^{(j)}||$.

Beispiele latenter Faktoren der Netflix-Challenge



Modellvergleich der Netflix-Challenge

Fragestellung: Wie wirkt sich die Berücksichtigung weiterer Einflüsse auf den RSME aus?



Collaborative Filtering with Temporal Dynamics (KDD 2009)

Qualitatives Fazit: Zunahme der Komplexität hilft wenig und eine Anpassung des Modells hilft viel!