universität koblenz
weiter:denken

Sicherheit von Recommender Systems

Aaron Pasternak

Inhaltsverzeichnis

Einführung ins Thema / Motivation

Relevanz von Recommender Systemen

- Vielfalt an Informationen und Daten in einer schnelllebigen Welt nimmt immer weiter zu → Filterung wirklich relevanter Inhalte essenziell !

- Unternehmen nutzen Recommender Systeme, um ihren Kunden personalisierte Empfehlungen zu machen und ihnen so bei der Entscheidungsfindung zu helfen [1]

- Empfehlungen können auf Produkteigenschaften, Bewertungen, Kundenanforderungen oder demografischen Informationen basieren [1]

- Beispiele für Recommender Systeme aus der realen Welt [1] :
  - Filmempfehlungen bei Netflix
  - Produktempfehlungen bei Amazon
  - Musikempfehlungen bei last.fm
  - Personalisierte Werbung bei Google
  - Freundschaftsvorschläge bei Facebook

Folgen unsicherer Recommender Systeme

- Sicherheitslücken bei Recommender Systemen können eine Reihe an negativen Konsequenzen nach sich ziehen, sowohl für die Unternehmen als auch für die Nutzer:
  - Manipulierte Empfehlungen können zu verminderter Kundenzufriedenheit und zum Verlust von Kunden führen → auch die Umsätze der Unternehmen sinken [1]
  - Unfreiwillig veröffentlichte Daten können die Reputation von Anbietern verschlechtern und finanzielle Sanktionen zur Folge haben [2]
- → Absicherung von Recommender Systemen von enormer Bedeutung !

Wichtige Grundlagen rund um Recommender Systeme und mit ihnen verbundene Sicherheitsaspekte

Definition und Ziele von Recommender Systemen

- Recommender Systeme sind Systeme, die den Nutzern basierend auf bestimmten Daten und Algorithmen Items empfehlen. Dies können je nach Branche beispielsweise sein: Bücher, Filme oder Produkte [1]

- **Ziele** [1] **:**
  - Maximierung der Erträge durch gesteigerte Verkaufszahlen
  - Erhöhung der Kundenzufriedenheit
  - Steigerung der Kundenloyalität
  - Aus Unternehmenssicht: verbessertes Verständnis der Bedürfnisse der Kunden

Geschichte von Recommender Systemen

- Erstes Recommender System wurde im Jahr 1979 entwickelt: “Grundy“ empfehlt seinen Nutzern basierend auf expliziten in einer Befragung erhobenen Angaben und dem Nutzerverhalten für sie potenziell relevante Bücher. Durch Rückmeldungen der Nutzer wurde das Systems stets weiterentwickelt [3].

- Im Jahr 1992 entwickelten Goldberg et al. erstmals ein kollaboratives Recommender System, das auf Nutzeraktivitäten wie Bewertungen basiert [4].

- 1994 wurde von Resnick et al. ein auf der Korrelation von Nutzerbewertungen basierendes Recommender System entwickelt, um Nutzern personalisierte Empfehlungen von für sie potenziell interessanten Nachrichtenartikeln bieten zu können [5].

Geschichte von Recommender Systemen

- Seit 1998: Amazon Recommender System für individuelle Produktvorschläge, erstes auf der Ähnlichkeit von Items basierendes System. Mit der Zeit wurden auch implizite Daten wie Klickverhalten, Käufe und Suchverläufe als Grundlage für die Empfehlugen verwendet [6].

- Ab 2010er Jahre: Verwendung fortschrittlicher Ansätze:
  - Deep Neural Networks bei YouTube [7]
  - Reinforecement Learning bei Spotify [8] zur Analyse, um zu lernen, aus welchen Gründen Nutzer Songs überspringen und so künftig noch bessere Empfehlungen bieten zu können

Collaborative Filtering Recommender Systeme

- Muster in Bewertungen im System werden als Grundlage für Empfehlungen verwendet

- Dies basiert auf dem Prinzip, dass Personen mit ähnlichen Interessen oft auch ähnliche Produkte positiv bewerten

- Wenn zwei Nutzer bisher viele gleiche Filme/Produkte positiv bewertet haben, ist es wahrscheinlich, dass sie auch bei weiteren Empfehlungen ähnliche Vorlieben zeigen [1]

Collaborative Filtering Recommender Systeme

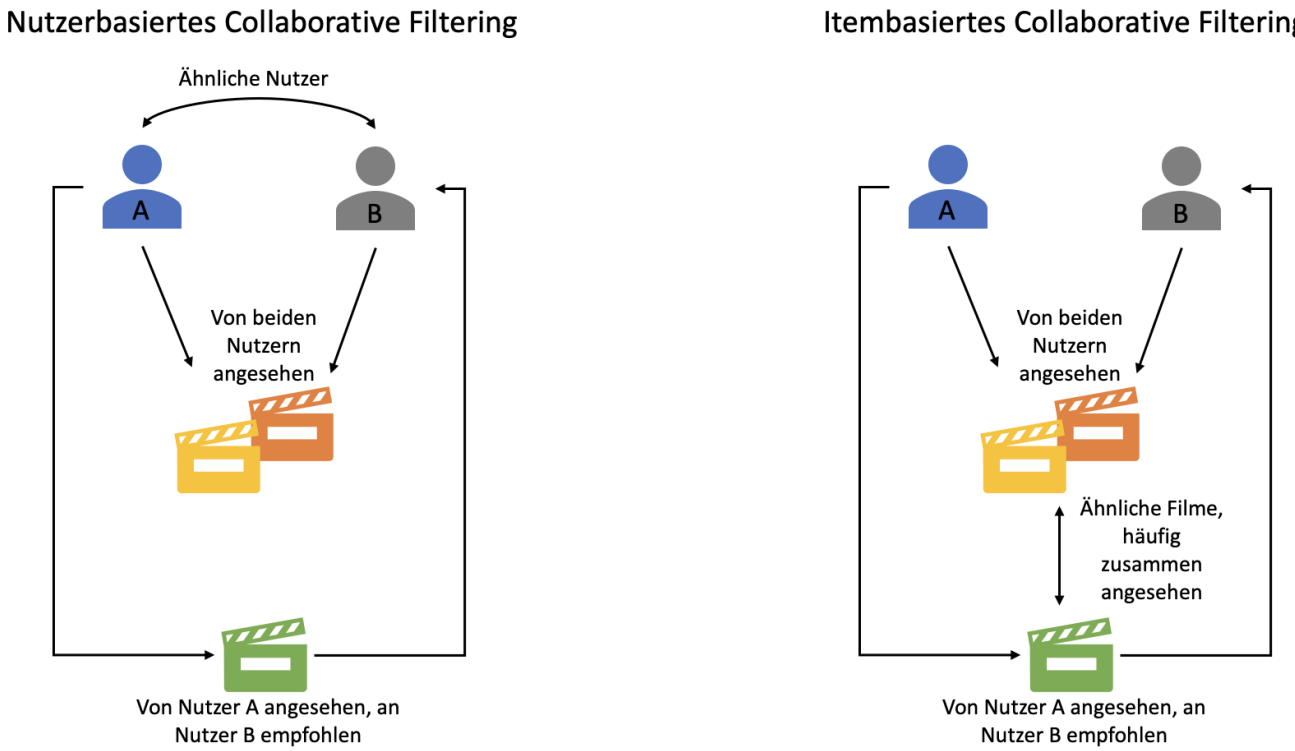
- Arten des Collaborative Filtering (CF) [1]:
  - **Memory-based CF (auch neighborhood-based genannt):**
    - Nutzerbasiertes (user-based) CF: Empfehlungen für eine Person weden auf Basis von Bewertungen anderer Nutzer erstellt, die ähnliche Präferenzen zeigen.
    - Itembasiertes (item-based) CF: Empfehlungen basieren auf der Ähnlichkeit von Items. Die Ähnlichkeit wird berechnet, indem analysiert wird, welche Items von denselben Nutzern ähnlich bewertet oder häufig zusammen konsumiert wurden.
  - **Model-based CF:**

  Diese Art basiert auf Machine Learning-Methoden, die datenbasiert Vorhersagen ermöglichen. Es wird ein Modell trainiert, das Muster in den vorhandenen Bewertungen erkennt und daraus Rückschlüsse auf unbekannte Bewertungen zieht. Beispiele sind Entscheidungsbäume oder Bayes-Modelle Faktorenmodelle.

  Wichtige Technik: Matrixfaktorisierung: Methode, um eine große Bewertungsmatrix in zwei kleinere Matrizen zu zerlegen. Dabei werden versteckte Muster erkannt, indem Nutzer- und Item-Eigenschaften in einem gemeinsamen Konzeptraum dargestellt werden. Singular Value Decomposition (SVD) und Alternating Least Squares (ALS) sind zwei in der Praxis häufig angewandte Algorithmen, die auf Matrixfaktorisierung basieren

Collaborative Filtering Recommender Systeme

- Veranschaulichung nutzerbasiertes vs. itembasiertes Collaborative Filtering:



Quelle: eigene Darstellung

Content-based Filtering Recommender Systeme

- Eigenschaften von Items werden verwendet, um darauf basierend Empfehlungen für Nutzer zu generieren. Das System analysiert die Eigenschaften der Items, die ein Nutzer positiv bewertet hat und schlägt ähnliche Inhalte vor [1]

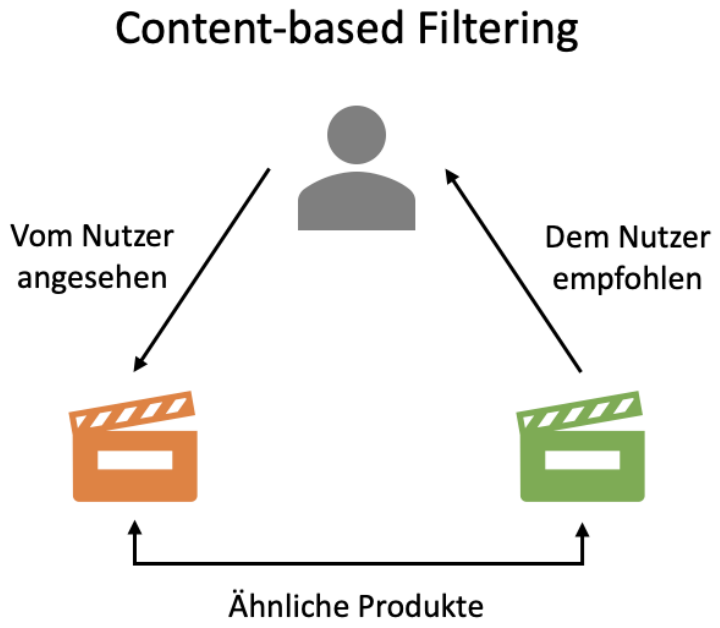
- Empfehlungsgrundlage: beispielsweise Genres, Schlüsselwörter oder Beschreibungen

- Content-based Filtering ist vor allem hilfreich, wenn es darum geht, Empfehlungen für neue Items zu generieren, zu denen bisher keine Bewertungen vorliegen. Stattdessen können Eigenschaften der Items genutzt werden, u basierend auf ähnlichen Items diese Items zu empfehlen.

- Nachteil dieser Methode: Nutzer bekommen oft nur sehr ähnliche Inhalte vorgeschlagen → Einschränkung der Vielfalt der Empfehlungen

Content-based Filtering Recommender Systeme

- Veranschaulichung Content-based Filtering:



Quelle: eigene Darstellung

Weitere Arten von Recommender Systemen

- **Knowledge-based Filtering** [1]**:**

Nutzer werden direkt nach ihren Anforderungen und Präferenzen gefragt, um maßgeschneiderte Vorschläge zu erstellen.
Beispielhafte Kontexte, in denen diese Art Anwendung findet: seltene und individuell konfigurierbare Produkte wie Immobilien oder Autos.

- **Demographic Filtering** [9]**:**

Demografische Daten der Nutzer wie das Alter oder das Geschlecht werden verwendet, um Nutzer zu gruppieren und ihnen individuelle Empfehlungen machen zu können. Diese Art von Recommender Systemen hat auch bei neuen Nutzern keine Schwierigkeiten, Empfehlungen zu generieren.

- **Hybride Ansätze** [1,9]**:**

Kombination mehrerer Ansätze, um so die Stärken verschiedener Ansätze zu nutzen und die Schwächen kompensieren zu können → noch robustere Systeme

Maße zur Bewertung der Empfehlungsqualität

- **Mean Absolute Error (MAE):** berechnet die durchschnittliche Abweichung zwischen realer und vorhergesagter Werte eines Recommender Systems. Dafür wird ein Datensatz in Trainings- und Testdaten eingeteilt und die Vorhersagen basierend auf den Testdaten werden genutzt, um die Qualität des Recommender Systems zu evaluieren [1].

$$e_{uj} = \hat{r}_{uj} - r_{uj}$$

$$MAE = \frac{\sum_{(u,j)\in E} |e_{uj}|}{|E|}$$

$\hat{r}_{uj}$ = vorhergesagte Bewertung des Nutzer u für das Item j

r_{uj} = tatsächliche Bewertung des Nutzer u für das Item j

E = Menge aller betrachteten Nutzer-Item Paare

$|E|$ = Anzahl aller betrachteten Nutzer-Item Paare

- **Root Mean Square Error (RMSE):** ähnlich wie MAE, allerdings hier stärkere Gewichtung größerer Abweichungen [1]

$$RMSE = \sqrt{\frac{\sum_{(u,j)\in E} e_{uj}^2}{|E|}}$$

Maße zur Bewertung der Empfehlungsqualität

- **HitRate@K:** Anteil der Nutzer, der in seinen Top k Empfehlungen mindestens ein relevantes (vom Nutzer positiv bewertetes) Item hat [10]

$$HitRate@K = \frac{1}{N} \sum_{u=1}^{N} \mathbb{1} \left[\exists i \in T_u^K \mid i \in R_u \right]$$

N = Nutzeranzahl; T_u^K = K dem Nutzer u empfohlene Items, R_u = relevante Items
$\mathbb{1}$ = 1, wenn mindestens ein Item i aus R_u in T_u^K enthalten, sonst $\mathbb{1}$ = 0

- **Prediction Shift:** durchschnittliche Veränderung vorhergesagter Bewertungen als Folge eines Angriffs [11]

$\Delta_{u,i} = p'_{u,i} - p_{u,i}$ $\Delta_{u,i}$ = Differenz zwischen neuen $p'_{u,i}$ und alten $p_{u,i}$ Vorhersagen

$\Delta_i = \sum_{u \in U} \Delta_{u,i} / |U|$ Δ_i = Veränderung der Vorhersagen für Item i. Summe über alle Nutzer u aus Menge U; $|U|$ = Anzahl der Nutzer

$\overline{\Delta} = \sum_{i \in I} \Delta_i / |I|$ $\overline{\Delta}$ = durchschnittlicher Prediction Shift über alle Items i

Datensätze zur Forschung

- Es gibt einige frei zugängliche Datensätze für Recommender Systeme. Diese Datensätze unterscheiden sich in verschiedenen Aspekten wie ihrer Größe oder der Branche, aus der sie stammen.

- Zu den bekanntesten und meist genutzten Datensätze zählen die Datensätze von MovieLens, die Bewertungen zu Filmen enthalten und in verschiedenen Größen wie mit 100.000, 1 Millionen oder 10 Millionen Bewertungen verfügbar sind.

- Durch ihre expliziten Bewertungen sind die meisten Datensätze für Forschungszwecke mit Collaborative Filtering-Algorithmen anwendbar.

- Viele Datensätze enthalten auch Eigenschaften wie das Genre, was eine Anwendung von Content-based Filtering-Algorithmen ebenfalls ermöglicht [12].

Datenbezogene Schwachstellen

- **Cold-Start Problem:** Schwierigkeit eines Collaborative Filtering Recommender Systems, Empfehlungen für neue Nutzer/Items zu machen, wenn keine Bewertungen vorliegen [13].
- **Spärlichkeit:** Schwierigkeit, Ähnlichkeiten zu erkennen, wenn die meisten Nutzer nur wenige Items bewertet haben und für viele Kombinationen aus Nutzer und Item keine Daten vorliegen [14].
- **Synonymie:** Ähnlichkeit zweier Items kann vom System nicht erkannt werden, wenn diese unterschiedliche Namen haben [15].
- **Begrenzte Inhaltsanalyse:** Zwei Items sind schwer zu unterscheiden, wenn sie im System mit denselben Eigenschaften repräsentiert sind [16].

Datenbezogene Schwachstellen

- **Big Data und Skalierbarkeit:** Recommender Systeme arbeiten häufig mit einer großen Menge an Daten. Durch neue Nutzer und Items können die Rechenanforderungen immer weiter ansteigen [15]. Exponentiell ansteigende Berechnungen können zu ungenauen Ergebnissen führen [13].

- **Latenzzeit:** In Recommender Systemen kann es vorkommen, dass neu hinzugekommene Items aufgrund fehlender zugehöriger Bewertungen nicht bzw. deutlich seltener im Vergleich zu bestehenden Items empfohlen werden [17].

Algorithmische Schwachstellen

- **Überspezialisierung bei Content-based Filtering:** Es kann passieren, dass Content-based Recommender Systeme den Nutzern nur ähnliche Items zu bisher angesehenen/gekauften empfehlen. Durch fehlende Variation lernen die Nutzer keine neuen Inhalte kennen. Dies kann zu sinkender Nutzerzufriedenheit führen [16].

- **Schwächen von nachbarschaftsbasierten Methoden:** Nachbarschaftsbasierte Methoden können für einen hohen Rechenaufwand sorgen. Durch Spärlichkeit in solchen Systemen wird die Berechnung von Ähnlichkeiten zwischen Nutzern eingeschränkt. Das erschwert präzise Empfehlungen [1].

- **Feedback Loops und Fairness:** Empfehlungen in Recommender Systemen basieren auf historischen Daten. Dies kann zu verzerrten Empfehlungen führen, wenn das System durch eine überrepräsentierte Nutzergruppe falsche Annahmen über individuelle Präferenzen trifft. Außerdem kann es dafür sorgen, dass populäre Items häufiger empfohlen werden als unpopuläre Items [18].

Algorithmische Schwachstellen

- **Diversität:** Wenn alle Empfehlungen in der Empfehlungsliste eines Nutzers sich nicht unterscheiden oder aus dem selben Genre stammen kann das dazu führen, dass einem Nutzer keine seiner Empfehlungen zusagt [1].

- **Neuartigkeit:** Empfehlungen eines Recommender Systems sollten auch neue Inhalte enthalten, die der Nutzer noch nicht kennt. So können Nutzer neue Inhalte kennenlernen und neue Präferenzen entdecken. Zugleich kann das System seine Informationen über die Präferenzen der Nutzer erweitern [1].

- **Serendipität:** Empfehlungen sollten zumindest teilweise überraschend sein und nicht nur den Nutzern bereits bekannte oder ähnliche Items beinhalten. Solche Überraschungen können langfristig die Zufriedenheit der Nutzer mit einem Recommender System steigern [1].

Sicherheits- und Manipulationsprobleme

- **Shilling-Angriffe:** Angriffe mit dem Ziel, Bewertungen in einem Recommender System zu manipulieren. Angegriffene Items sollen vermehrt/vermindert empfohlen werden. Zudem wird die Performance der Empfehlungsqualität angegriffen [19]. Angreifer nutzen Verschleierungstechniken um unbemerkt zu bleiben [20].

- **Robustheit und Stabilität:** Systeme müssen robust gegen Angriffe wie Shilling-Attacken sein. Implementierte Mechanismen sollten eine Manipulation kurz- und langfristig verhindern [1].

- **Betrug:** Anbieter könnten durch Fake-Bewertungen versuchen, die Empfehlungwahrscheinlichkeit eigener Items zu stärken oder die der Konkurrenz zu schwächen, um sich einen Vorteil zu verschaffen [21].

Nutzerzentrierte Herausforderungen

- **Gray Sheep:** In kleinen oder mittelgroßen auf Collaborative Filtering basierenden Recommender Systemen kann es vorkommen, dass einzelne Nutzer sich in ihren Präferenzen von den meisten anderen Nutzern unterscheiden. So ist es schwer, diese Nutzer zu gruppieren und ihnen präzise, zutreffende Empfehlungen zu machen [22].

- **Vertrauen und Nutzerakzeptanz:** Empfehlungen sollten nachvollziehbar sein, um Unsicherheit auf Seiten der Nutzer zu reduzieren. Umsetzung u.a. durch Vergleiche von empfohlenen Produkten, z.B. mit anderen gekauften/angesehenen Produkten. Vertrauenswürdige Empfehlungen in einem Online-Shop können zu mehr Käufen der Nutzer führen [2].

Relevanz von Privatsphäre und Datenschutz

- Verarbeitung vieler verschiedener Daten in Recommender Systemen [23]:
  - Bewertungen, Kommentare, Favoriten
  - Kauf-/View-Historie
  - Nutzerdaten: demografische Merkmale, Verbindungen zu anderen Personen, Mitgliedschaften in Gruppen
- Daten werden von Anbietern (teilweise) plattformübergreifend geteilt [24]
- Daten können private Informationen wie politische Ansichten oder sexuelle Orientierung enthalten [1]
- Offenlegung von Daten → Reputationsverlust und weniger Gewinne bei Anbieter; Nutzer verlieren Vertrauen, teilen keine Inhalte mehr [2,25]
- → Maßnahmen zum Schutz der Daten & zur Einhaltung der DSGVO [26] notwendig !

Techniken zur Sicherstellung von Privatsphäre und Datenschutz

- **Stochastische Methoden: Zufallsrauschen:**
  - Randomized Perturbation als Ansatz, bei dem jeder Bewertung ein zufälliger Wert hinzugefügt wird → tatsächliche Bewertungen nicht mehr ersichtlich, aggregierte Informationen können weiterhin zur Generierung der Empfehlungen verwendet werden [27]
  - Differential Privacy: Rauschen zu Daten hinzufügen, um Rückschlüsse auf individuelle Nutzerdaten zu erschweren. Stärke des Rauschens abhängig von Sensitivität der Abfrage. Ziel: Wahrscheinlichkeiten sollen sich nicht signifikant ändern, unabhängig davon, ob eine bestimmte Person in den Daten enthalten ist oder nicht [28]

- **Anonymisierungsverfahren:**
  - k-Anonymität: spezifische Daten durch allgemeine ersetzen und generalisieren, damit Daten nicht mehr eindeutig einzelnen Personen zuordenbar sind → Jede Kombination von Quasi-Identifikatoren (z.B. Geschlecht, Geburtsdatum) wird mit mindestens k Personen geteilt [29]
  - l-Diversität: Daten sind genau dann l-divers, wenn es in jeder Gruppe mindestens l verschiedene gut verteilte Wert des sensiblen Attributs gibt [30]
  - t-Closeness: Ziel: Verteilung eines sensiblen Attributs innerhalb jeder Äquivalenzklasse der Gesamtverteilung des Attributs im Datensatz möglichst ähnlich halten, um Informationsgewinn über einzelne Personen einzuschränken [31]

Techniken zur Sicherstellung von Privatsphäre und Datenschutz

- **Kryptographische Methoden:**
  - Homomorphe Verschlüsselung: Verschlüsselungsverfahren, das Berechnungen direkt auf verschlüsselten Daten ermöglicht, so dass entschlüsseltes Ergebnis mit dem einer Berechnung auf den unverschlüsselten Daten übereinstimmt → So können in Recommender Systemen Nutzerdaten verschlüsselt bleiben und trotzdem als Grundlage für die Generierung der Empfehlungen genutzt werden → Keine Gefährdung der Vertraulichkeit [32]

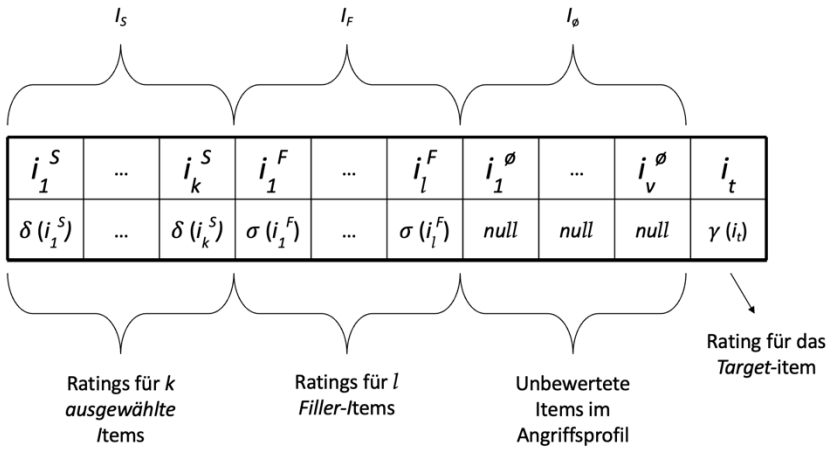
- **Dezentrale und agentbasierte Lösungen:**
  - Agentenbasierte Ansätze: Personalisierte Empfehlung, ohne dass es dafür einer dauerhaften Speicherung oder Weitergabe privater Informationen bedarf → Nutzerprofile in einem Recommender System werden nur temporär verarbeitet. Durch ein Kommunikationsprotokoll und vertrauenswürdige Agenten wird die Offenlegung sensibler Daten verhindert. → Datenschutzanforderungen können eingehalten werden [33]
  - Blockchain: Verkettete Transaktionsblöcke, die durch Hashes abgesichert sind. Jede Transaktion wird mittels Peer-to-Peer-Netzwerk validiert und in eine unveränderbare Kette aufgenommen → Manipulation wird erschwert, da immense Rechenleistung benötigt würde. → So könnten in Recommender Systemen die Nutzerdaten vor Manipulation geschützt werden [34]

Shilling-Angriffe

- Angriffe auf Collaborative Filtering Recommender Systeme durch Einfügen von Fake-Bewertungen mit dem Ziel, die Empfehlungen zu verzerren [1]

- Empfehlungswahrscheinlichkeit eines Items soll erhöht (**Push-Angriffe**) oder vermindert (**Nuke-Angriffe**) werden [35]

- **Angriffsprofil:** Bewertungen des Nutzerprofils eines Angreifers, besteht aus [36]:
  - Bewertung für das Ziel- bzw. Target-Item
  - Bewertungen für ausgewählte Items
  - Bewertungen für zufällig ausgesuchte Filler-Items zur Verschleierung der Attacke



Quelle: eigene Darstellung (nach [36])

Shilling-Angriffe: Schlüsselbegriffe

- **Angriffsgröße:** Anzahl an Angriffsprofilen, die im Rahmen eines Angriffs Bewertungen in das System einfügen [37]
- **Filler-Größe:** Anzahl an Filler-Items, die als Bestandteil eines jeden Angriffsprofils im Rahmen einer Attacke bewertet werden [38]
- **Profil-Größe:** Anzahl an Ratings, die ein Angriffsprofil enthält [38]

Arten von Shilling-Angriffen

- **Bandwagon:** Target-Item wird maximal bewertet, populäre Items werden ebenfalls maximal bewertet, um die Angriffsprofile ähnlicher zu denen normaler Nutzer erscheinen zu lassen, Bewertungen der Filler-Items folgen einer Normalverteilung mit dem Mittelwert und der Standardabweichung des Systems [39]

- **Random**: Größtenteils identisch zu Bandwagon-Angriffen mit dem Unterschied, dass hier keine ausgewählten (populären) Items bewertet werden [39]

- **Average:** Ähnlich wie Random-Angriffe mit dem Unterschied, dass hier die Bewertung der Filler-Items nicht auf dem Gesamt-Mittelwert des Systems sondern auf den individuellen Mittelwerten der bewerteten Items basiert [40]

Arten von Shilling-Angriffen

- **Segment:** Ziel: ein Item innerhalb eines bestimmten Segments zu pushen bzw. die Ähnlichkeit des Target-Items zu anderen Items desselben Segments zu maximieren. Target-Item und Items aus dem selben Segment (z.B. Genre) werden maximal bewertet. Für besondere Effektivität dieser Angriffsart werden Filler-Items minimal bewertet [41]

- **Reverse Bandwagon:** Gegenteil von Bandwagon. Nuke-Angriffsart. Target-Item und unbeliebte Items werden minimal bewertet, Bewertungen der Filler-Items folgen einer Normalverteilung mit dem Mittelwert und der Standardabweichung des Systems [42]

- **Weitere Arten:** z.B. Probe-Angriffe, Love/Hate-Angriffe, Power-User-Angriffe, Power-Item-Angriffe [37,42]

Features zur Erkennung von Shilling-Angriffen

Generische Features:

- **Rating Deviation from Mean Agreeement (RDMA):** misst die Abweichung der Bewertungen eines Nutzers zu den durchschnittlichen Item-Bewertungen und untersucht, inwiefern die Bewertungen von anderen Nutzern beeinflusst wurden [43]

$$RDMA_j = \frac{\sum_{i=0}^{N_j} \frac{|r_{i,j} - Avg_i|}{NR_i}}{N_j}$$

j=Nutzer, i=item, NR_i=Anzahl Ratings für i im System, N_j= Anzahl Ratings von j, r=rating Avg_i=durchschnittliche Bewertung von i

- **Weighted Deviation from Mean Agreeement (WDMA):** ähnlich wie RDMA, Bewertungsabweichungen selten bewerteter Items werden stärker gewichtet [36]

$$WDMA_u = \frac{\sum_{i=0}^{n_u} \frac{|r_{u,i} - \bar{r_i}|}{l_i^2}}{n_u}$$

u=Nutzer, i=Item, l_i= Anzahl Ratings für i im System, n_u=Anzahl Ratings von u, r=rating, $\bar{r}_i$= durchschnittliche Bewertung von i

Features zur Erkennung von Shilling-Angriffen

Generische Features:

- **Length Variance (LengthVar):** analysiert, wie stark die Profillänge eines Nutzerprofils von der durchschnittlichen Profillänge aller Nutzer abweicht [41]

$$LengthVar_u = \frac{|n_u - \overline{n_u}|}{\sum_{u \in U} (n_u - \overline{n_u})^2}$$

n_u= Anzahl an Bewertungen von einem Nutzer u

$\overline{n_u}$ = durchschnittliche Anzahl an Bewertungen von allen Nutzern

- **Degree of Similarity (DegSim):** misst die durchschnittliche Ähnlichkeit (Pearson-Korrelation) eines Nutzerprofils zu seinen k nächsten Nachbarn [44]

$$DegSim = \frac{\sum_{u=1}^{k} W_{uv}}{k}$$

W_{uv}= Ähnlichkeit zwischen Nutzer v und seinem Nachbarn u. Summe läuft über k nächste Nachbarn von v

Features zur Erkennung von Shilling-Angriffen

Modellspezifische Features: Einteilung in Target-Items (Items mit maximaler Bewertung) und Filler-Items (Rest) [45]:

- **Mean Variance (MeanVar):** durchschnittliche Varianz zwischen Bewertungen der Filler-Items und deren Mittelwerten. Kleine Differenz → vermutlich Average-Angriff [45]

$$MeanVar_u = \frac{\sum_{i \in P_{u,F}} (r_{u,i} - \overline{r_i})^2}{|P_{u,F}|}$$

$r_{u,i}$=Bewertung des Nutzers u für Filler-Item i
$\bar{r}_i$= durchschnittliche Bewertung von i
$P_{u,F}$= Menge von u bewerteter Filler-Items

- **Filler Mean Target Difference (FMTD):** Abweichung zwischen durchschnittlicher Bewertung der Filler- und Target-Items. Hohe Abweichung → vermutlich Segment-Angriff [41]

$$FMTD_u = \left| \left(\frac{\sum_{i \in P_{u,T}} r_{u,i}}{|P_{u,T}|} \right) - \left(\frac{\sum_{k \in P_{u,F}} r_{u,k}}{|P_{u,F}|} \right) \right|$$

$r_{u,k}$ = Bewertung des Nutzers u für Target-Item k
$P_{u,T}$ =Menge von u bewerteter Target-Items

Features zur Erkennung von Shilling-Angriffen

Modellspezifische Features: Einteilung in Target-Item (Items mit maximaler Bewertung) und Filler-Items (Rest) [45]:

- **Filler Average Correlation (FAC):** misst die durchschnittliche Differenz zwischen der Filler-Bewertungen eines Nutzers und der durchschnittlichen Bewertungen der Filler-Items im System. Geringe Korrelation → vermutlich Random-Angriff [45]

$$FAC = \frac{\sum_{i \in I_u} (r_{u,i} - \bar{r}_i)}{\sqrt{\sum_{i \in I_u} (r_{u,i} - \bar{r}_i)^2}}$$

$r_{u,i}$=Bewertung des Nutzers u für Filler-Item i

$\bar{r}_i$= durchschnittliche Bewertung von i

- **Filler Mean Difference (FMD):** Abweichung zwischen Bewertung der Filler-Items eines Nutzers und der durchschnittlichen Bewertung aller Items im System. Geringe Abweichung → vermutlich Random-Angriff [45]

$$FMD = \frac{1}{U_u} \sum_{i=1}^{|U|} |r_{u,i} - \bar{r}_i|$$

U_u=Anzahl der Filler-Bewertungen des Nutzers u

Machine Learning

- Bereich der Informatik, der verschiedene Algorithmen und Techniken zur automatisierten Lösung komplexer Probleme beinhaltet [46]
- Algorithmen lernen Strukturen, Regeln und Zusammenhänge aus Daten
- Seit Mitte der 2000er Jahre große Fortschritte dank großer frei verfügbarere Datenmengen, hoher möglicher Rechenkapazität und optimierter Algorithmen
- Probleme, in denen Machine Learning angewendet wird: Klassifikation, Clustering, Vorhersagen

Teilbereiche des Machine Learning

- **Überwachtes Lernen:** Verwendung von gelabelten Daten als Grundlage für Klassifikationsalgorithmen. Diese lernen Eigenschaften aus Daten und versuchen für neue Daten die richtigen Antworten vorherzusagen. Klassifikation (Vorhersage bestimmer Klasse) und Regression (Vorhersage kontinuierlicher Variablen) [46]

- **Nicht überwachtes Lernen:** nicht gelabelte Daten als Eingabe für Machine Learning Algorithmen. Ziel: Trends und Zusammenhänge aus Daten ableiten und trennbare Gruppen bilden (Clustering), in denen sich die Daten jeweils ähneln [46]

- **Semi-überwachtes Lernen:** Kombination von Aspekten des überwachten und des nicht überwachten Lernens. Nur Teil der Daten gelabeled. Clustering mit Rest der Daten und Zuweisung der am besten zu den Daten passenden Labels [46]

- **Deep Learning:** Verwendung künstlicher neuronaler Netze, um aus Daten zu lernen. Anwendungsbereiche: Spracherkennung, NLP, Computer Vision [42]

Machine Learning Algorithmen

- **Naive Bayes:** Basiert auf dem Bayes Theorem, nutzt die bedingte Wahrscheinlichkeit und beruht auf der Annahme, dass das Vorkommen eines bestimmten Features in einer Klasse nicht mit dem Vorkommen eines anderen zusammenhängt [47]

- **K-Nearest Neighbors:** Basiert auf der auf der Ähnlichkeit von Daten. Daten müssen erst normalisiert werden. Für jeden Datenpunkt werden die k nächsten Nachbarn basierend auf der euklidischen Distanz bestimmt → Klassifikation [48]

- **Decision Tree:** Baumstruktur, Algorithmus trifft durch rekursive Aufteilung der Daten anhand von Entscheidungsregelen eine Vorhersage. Schrittweise Verteilung der Daten anhand Merkmale auf Knoten, dann finale Entscheidung [49]

- **Random Forest:** basiert auf mehreren Entscheidungsbäumen, die auf zufällig ausgewählten Datenmerkmalen beruhen, um Korrelation zu reduzieren. Kombination der Bäume → präzisere Vorhersagen, robust gegen Overfitting [50]

Machine Learning Algorithmen

- **Support Vector Machines:** Daten werden basierend auf Hyperebene optimal voneinander getrennt. Gewichtung aller Attribute, Berechnung des höchsten Spielraums pro Attribut um Abstand zur Hyperebene zu maximieren [38]

- **Logistische Regression:** Zusammenhang zwischen Output- und Vorhersage-Variablen wird durch geschätzte Wahrscheinlichkeiten basierend auf logistischer Funktion gemessen, um optimale Vorhersagen zu treffen → binäre Klassifikation [51]

- **XGBoost:** Leistungsstarker Boosting-Algorithmus. Basiert auf regularisierter Zielfunktion und Gradient-Tree-Boosting. Parallele Datenverarbeitung und optimierte Speicherzugriffe → für große Datensätze hervorragend geeignet [52]

- **Neural Network:** Nachahmung der Funktionsweise des menschl. Gehirns, um Zusammenhänge in Daten zu erkennen und aus diesen zu lernen. Schichten für Dateneingabe und –verarbeitung. Können sich an veränderte Eingaben anpassen [47]

Stacking-/Ensemble-Ansätze:

- **Ensemble:** Kombination der Vorhersagen mehrerer Modelle → verbesserte Modell-Performance [47]

- **Stacking:** konkreter Ensemble-Ansatz, zweischichtig [53]:
  - Mehrere Algorithmen als Basis-Algorithmen, aus denen das Modell lernt
  - Verwendung eines weiterem Algorithmus für die finale Vorhersage

Herausforderungen & Schwachstellen von Machine Learning Algorithmen

- **Overfitting:** Algorithmen sind zu angepasst an spezifische Daten, können schlecht auf neue Daten generalisieren [54]
- **Underfitting:** Modelle sind zu unangepasst auf die Daten [54]
- **Unvollständige Daten / Rauschen:** Vorverarbeitung der Daten nötig, sonst deutliche Schwächung der Modellperformance möglich [55,56]
- **Größe von Datensätzen:** Manche Algorithmen (z.B. Support Vector Machines, Neural Network) besser für große, andere eher für kleinere Datensätze geeignet [57]
- **Art von Daten:** Manche Algorithmen besser für kontinuierliche Features, andere funktionieren eher mit kategorischen Daten [57]
- **Hyperparameter Tuning:** Finden optimaler Hyperparameter (rechenaufwändig) → diese können sich für verschiedene Datensätze unterscheiden [54]

Herausforderungen & Schwachstellen von Machine Learning Algorithmen

- **Hoher Rechenaufwand & lange Trainingszeit:** Einige Algorithmen (z.B. Support Vector Machines, K-Nearest Neighbor) benötigen bei großen Datensätzen hohen Rechenaufwand (steigt mit größeren Datensätzen exponentiell an) und lange Trainingszeit [54,56]

- **Unbalancierte Klassen:** kann Performance der Algorithmen schwächen [58]

- **Concept Drift:** Neue Daten und veränderte Strukturen/Zusammenhänge können ein erneutes Training eines Machine Learning Modells nötig machen [58]

- **Weitere Herausforderungen:** z.B. Adversarial Attacks oder Privatsphäre & Datenschutz [59]

Maße zur Auswertung einer Klassifikation

TP: korrekt als positiv klassifizierte Daten
TN: korrekt als negativ klassifiziert
FP: falsch als positiv klassifiziert
FN: falsch als negativ klassifiziert

- **Precision:** wie viele der als Angriffsprofile klassifizierten wirklich solche sind [60]

$$\frac{TP}{TP + FP}$$

- **Recall:** wie viele der Angriffsprofile auch korrekt als solche erkannt werden [60]

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

- **F1-Score:** harmonisches Mittel aus Precision und Recall → Gesamtqualität [61]

$$\frac{TP}{TP + 0,5 \times (FP + FN)}$$

- **PR-AUC:** Fläche unter Precision-Recall-Kurve. Berücksichtigt, wie Precision und Recall bei verschiedenen Schwellenwerten im Verhältnis zueinander stehen. Fokus auf positive Klasse → vor allem bei unausgewogenen Datensätzen geeignet [62]

Tools und Technologien zur Simulation von Shilling Angriffen und Angriffserkennung

- **Python** [63]**:**
  - Leistungsfähige interpretierte Programmiersprache
  - Zeichnet sich durch optimierte Datenstrukturen, intuitive objektorientierte Programmierung und gut lesbare Syntax aus

- **Jupyter Notebooks** [64]**:**
  - Interaktive Umgebung, in der Code in versch. Programmiersprachen (z.B. Python) geschrieben und direkt im Browser ausgeführt werden kann.
  - Code in Zellen organisiert → einzelne Ausführung und Testung möglich
  - Abbildung von Grafiken, Tabellen und mathematischen Formeln → optimal für Machine Learning geeignet

- **Google Colab** [65]**:**
  - Cloudbasierte Umgebung für Jupyter Notebooks, die direkt im Browser genutzt werden kann
  - Stellt Rechenressourcen wie Graphics Processing Units (GPUs) und Tensor Processing Units (TPUs) bereit → besonders für Machine Learning geeignet

Aktueller Forschungsstand und Forschungslücken

Aktueller Forschungsstand

- Bereits einige Publikationen zur Sicherheit von Recommender Systemen in der Fachliteratur vorhanden. Analyse sowohl, wie sich Angriffe auf Empfehlungsqualität auswirken als auch wie gut Angriffe verschiedener Arten und Größen mit unterschiedlichen Ansätzen erkannt werden können

- Verschiedene Ansätze angewandt: überwachtes Lernen, nicht überwachtes Lernen, Deep Learning → Vor allem Ensemble-Methoden mit guter Performance (Werte von über 90% bei Precision, Recall und F1-Score)

- Bisher fast immer sehr dichte, große Datensätze (z.B. MovieLens, Netflix) genutzt

- In den meisten Studien festgelegte Filler-Größen (z.B. 5%, 10% oder 50%) verwendet → je nach Datensatz deutlich über- oder unterdurchschnittliche Profillänge der Angreifer im Vergleich zu normalen Nutzerprofilen → kann Klassifikationsqualität entscheidened beeinflussen

Forschungslücken

Forschungslücken:

1. Wie ist die Qualität der Angriffserkennung bei kleineren, weniger dichten Datensätzen?

2. Wie wirkt es sich auf die Performance der Angriffserkennung aus, wenn die Filler-Anzahl in den Angriffsprofilen dynamisch an die durchschnittliche Anzahl an Bewertungen normaler Nutzer angepasst wird, so dass es hinsichtlich der Profillänge keine signifikanten Unterschiede zwischen normalen Nutzern und Angreifern gibt und sich diese nur in ihrem Bewertungsmuster unterscheiden?

Methodik: Simulation von Shilling-Angriffen und Klassifikation mit Machine Learning

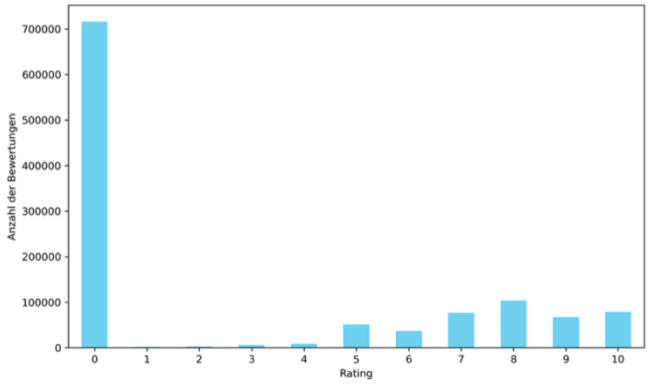
Methodik: Überblick

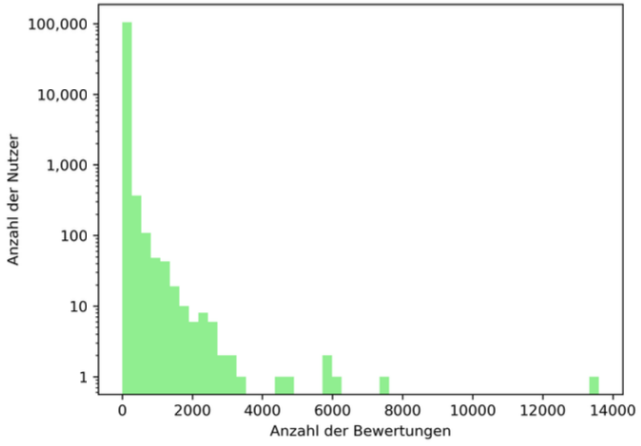
- Simulation von Shilling Angriffen verschiedener Arten und Größen
- Analyse der Auswirkungen der Angriffe
- Angriffserkennung mit Machine Learning
- Nutzung von drei Datensätzen, die sich in Branche, Dichte und Größe unterscheiden:

Datensatz	Branche	Nutzer	Items	Bewertungen	Dichte
BookCrossing	Bücher	105.283	340.556	1.149.780	0,0032%
Yelp	Unternehmen	1.987.929	150.346	6.990.280	0,0023%
MovieLens 32M	Filme	200.948	84.432	32.000.204	0,1886%

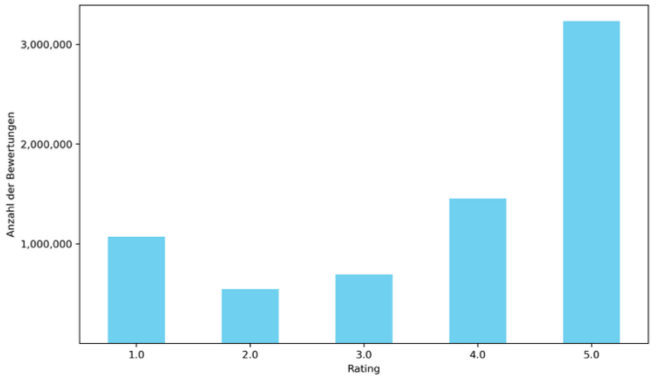
Datensätze: Verteilung der Bewertungen & Bewertungen pro Nutzer

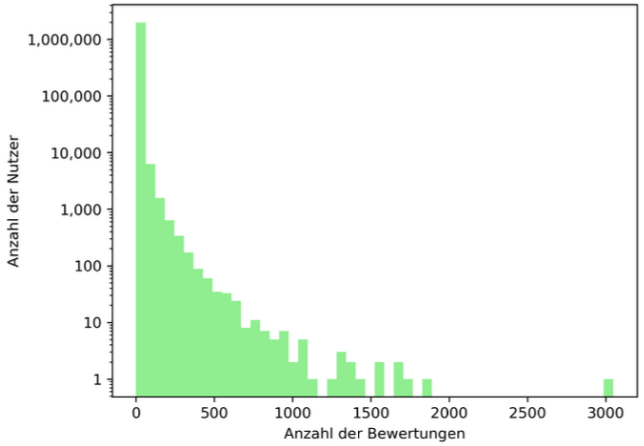
BookCrossing [66]:



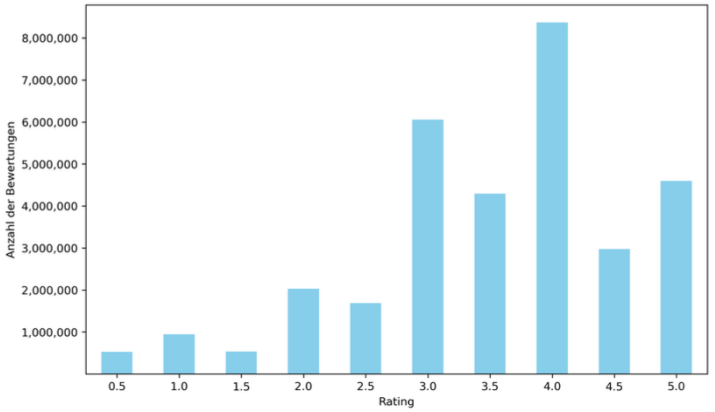


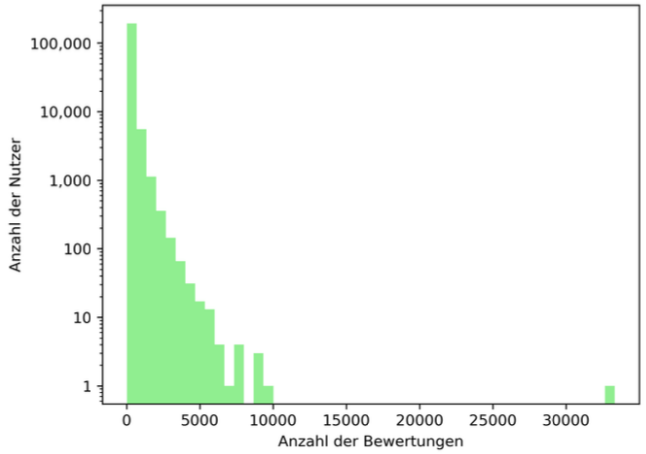
Yelp [67]:





MovieLens 32M [68]:





Datenvorverarbeitung & Durchführung

- Datenvorverarbeitung:
  - Entfernung von Duplikaten
  - Berechnung von Attributen zur Angriffserkennung (RDMA, WDMA, LengthVar, DegSim, MeanVar, FMTD, FAC, FMD)
  - Stratified K-fold-Cross Validation [69] mit K = 5
  - Hyperparameter Tuning für jeden Algorithmus mit GridSearchCV und RandomizedSearchCV

- Untersuchung:
  - Wie wirken sich die Angriffe jeweils auf die Empfehlungsqualität von auf dem Alternating Least Squares (ALS) Algorithmus basierenden Recommender Systemen aus? Wie verändern sich MAE, RMSE und HitRate@K nach Angriffen? Zudem Messung des Prediction Shift
  - Wie gut lassen sich die Angriffsprofile mit verschiedenen Machine Learning-Modellen (Naive Bayes, K-Nearest Neighbors, Decision Tree, Random Forest, Support Vector Machine, Logistic Regression, XGBoost, Neural Network, Stacking) erkennen? Auswertung basierend auf Precision, Recall, F1-Score und PR-AUC

- Vergleich der Ergebnisse mit der Literatur

Analyse der Auswirkungen von Shilling-Angriffen und deren Erkennung

Auswertung der Empfehlungsqualität beim BookCrossing Datensatz

Angriffstyp	/	Bandwagon				Random				Average				Segment				Reverse Bandwagon				Ø
Angriffsgröße	/	1%	5%	10%	20%	1%	5%	10%	20%	1%	5%	10%	20%	1%	5%	10%	20%	1%	5%	10%	20%	
RMSE	2,01	2,06	2,24	2,42	2,69	2,07	2,33	2,53	**2,86**	2,06	2,24	2,42	2,70	1,99	1,95	1,92	1,84	2,06	2,26	2,46	2,77	2,29
MAE	1,19	1,21	1,31	1,41	1,57	1,22	1,36	1,48	**1,72**	1,21	1,31	1,41	1,59	1,17	1,12	1,01	1,00	1,21	1,30	1,40	1,59	1,33
HitRate@10	0,09%	0,08%	0,05%	0,07%	0,02%	0,05%	0,05%	0,06%	0,02%	0,04%	0,06%	0,04%	0,02%	0,05%	0,04%	**0,10%**	0,09%	0,05%	0,04%	0,04%	0,03%	0,05%
HitRate@20	0,14%	0,11%	0,10%	0,11%	0,05%	0,09%	0,07%	0,10%	0,04%	0,07%	0,12%	0,07%	0,05%	0,09%	0,11%	0,14%	**0,15%**	0,10%	0,09%	0,06%	0,06%	0,09%
Prediction Shift	/	0,03	0,03	**0,18**	**0,18**	0,03	0,03	0,04	0,04	0,03	0,03	0,04	**0,18**	0,03	0,03	**0,18**	**0,18**	0,03	0,03	0,04	0,16	0,08

- Fehlermaße **MAE** und **RMSE** steigen vor allem bei groß angelegten Angriffen deutlich an. Stärkste Auswirkungen bei 20%-Random-Angriffen. Gegenteilige Tendenzen bei Segment-Angriffen aufgrund spezieller Bewertungssystematik bei dieser Angriffsart

- **HitRate@10/HitRate@20:** Jeweils leichte Verschlechterung mit zunehmender Angriffsgröße. Steigende Werte nur bei Segment-Angriffen der Größen 10%/20%

- **Prediction Shift:** Große Auswirkungen von Bandwagon-/Segment-Angriffen der Größen 10/20% sowie von 20%-Average-Angriffen. Sonst moderate Veränderung

Auswertung der Empfehlungsqualität beim Yelp Datensatz

universität koblenz
weiter:denken

Angriffstyp	/	Bandwagon				Random				Average				Segment				Reverse Bandwagon				Ø
Angriffsgröße	/	1%	5%	10%	20%	1%	5%	10%	20%	1%	5%	10%	20%	1%	5%	10%	20%	1%	5%	10%	20%	
RMSE	1,58	1,58	1,57	1,56	1,54	1,58	1,63	1,66	1,71	1,59	1,62	1,66	**1,72**	1,58	1,57	1,56	1,54	1,58	1,57	1,55	1,52	1,60
MAE	1,26	1,25	1,23	1,21	1,18	1,26	1,29	1,32	**1,37**	1,26	1,29	1,32	**1,37**	1,25	1,23	1,20	1,15	1,25	1,23	1,20	1,15	1,25
HitRate@10	0,10%	0,11%	0,10%	0,10%	0,09%	0,11%	0,12%	0,13%	0,18%	0,11%	0,11%	0,11%	0,13%	1,33%	10,11%	18,12%	**30,47%**	0,12%	0,10%	0,10%	0,08%	3,09%
HitRate@20	0,22%	0,22%	0,21%	0,20%	0,20%	0,22%	0,24%	0,27%	0,35%	0,23%	0,22%	0,24%	0,26%	2,21%	10,22%	18,21%	**30,56%**	0,23%	0,21%	0,18%	0,16%	3,24%
Prediction Shift	/	**0,04**	**0,04**	**0,04**	**0,04**	**0,04**	**0,04**	**0,04**	**0,04**	**0,04**	**0,04**	**0,04**	**0,04**	**0,04**	**0,04**	**0,04**	**0,04**	**0,04**	**0,04**	**0,04**	**0,04**	0,04

- **RMSE/MAE:** Sinkende Werte bei Arten Bandwagon, Segment und Reverse Bandwagon. Gründe: Geringe Dichte des Datensatzes und ausgewählte Items in Angriffsprofilen. Max. MAE-Wert wie bei BookCrossing bei 20%-Random-Angriffen

- **HitRate@10/HitRate@20:** Zu erwartendes Verhalten zu beobachten: steigende Werte bei Push-Angriffen, sinkende bei der Nuke-Angriffsart Reverse Bandwagon. Höchste Werte bei Segment-Angriffen, wie beim BookCrossing Datensatz

- **Prediction Shift:** keine Unterschiede zwischen verschiedenen Angriffsarten und Angriffsgrößen → keine starke Beeinflussung der vorhergesagten Werte

Auswertung der Empfehlungsqualität beim MovieLens 32M Datensatz

universität koblenz
weiter:denken

Angriffstyp	/	Bandwagon				Random				Average				Segment				Reverse Bandwagon				Ø
Angriffsgröße	/	1%	5%	10%	20%	1%	5%	10%	20%	1%	5%	10%	20%	1%	5%	10%	20%	1%	5%	10%	20%	
RMSE	0,76	0,76	0,77	0,79	0,81	0,77	0,78	0,80	**0,82**	0,76	0,78	0,79	0,81	0,76	0,74	0,73	0,70	0,76	0,77	0,79	0,81	0,77
MAE	0,58	0,58	0,59	0,60	0,62	0,58	0,60	0,61	**0,63**	0,58	0,59	0,60	0,62	0,58	0,56	0,53	0,49	0,58	0,59	0,60	0,61	0,59
HitRate@10	5,90%	14,31%	24,06%	28,41%	**34,35%**	9,35%	12,19%	13,73%	14,72%	3,36%	2,30%	1,14%	0,66%	11,66%	25,00%	20,74%	20,79%	7,67%	9,75%	10,06%	9,88%	13,71%
HitRate@20	12,58%	24,10%	35,45%	40,12%	**46,67%**	17,14%	19,67%	20,80%	21,35%	6,11%	3,46%	2,55%	1,23%	20,16%	36,96%	27,49%	24,79%	14,74%	16,60%	16,36%	15,46%	20,56%
Prediction Shift	/	0,08	0,08	0,08	**0,09**	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,08

- **RMSE/MAE:** Wie beim BookCrossing Datensatz: Steigende Werte mit zunehmender Angriffsgröße bei allen Arten außer Segment. Auch hier 20%-Random-Angriffe mit den höchsten Werten bei den beiden Fehlermaßen

- **HitRate@10/HitRate@20:** Wie beim Yelp Datensatz: Größtenteils steigende Werte nach Angriffen. Ziel wird erreicht: Angriffe verändern die Empfehlungen signifikant. Höchste Werte bei Bandwagon- und Segment-Angriffen

- **Prediction Shift:** keine Unterschiede zwischen verschiedenen Angriffsarten und Angriffsgrößen → moderate Beeinflussung der vorhergesagten Werte

Auswertung der Angriffserkennung beim BookCrossing Datensatz (F1-Score)

universität koblenz
weiter:denken

Angriffstyp	Bandwagon				Random				Average				Segment				Reverse Bandwagon				Ø
Angriffsgröße	1%	5%	10%	20%	1%	5%	10%	20%	1%	5%	10%	20%	1%	5%	10%	20%	1%	5%	10%	20%	
Naive Bayes	14,03%	34,52%	48,21%	61,97%	21,17%	44,18%	57,75%	70,99%	19,02%	43,31%	57,78%	70,92%	14,58%	38,36%	52,29%	65,96%	25,77%	50,87%	62,15%	72,32%	46,31%
K-Nearest-Neighbor	82,92%	89,64%	91,79%	93,97%	84,70%	91,86%	93,01%	94,38%	79,22%	88,41%	90,96%	93,06%	91,43%	96,81%	98,04%	98,85%	73,74%	87,97%	90,35%	92,67%	90,19%
Decision Tree	78,54%	88,06%	90,99%	93,90%	81,10%	89,85%	92,04%	93,61%	73,02%	86,81%	89,62%	92,58%	92,55%	98,10%	98,97%	99,31%	75,84%	88,09%	90,41%	93,29%	89,33%
Random Forest	84,19%	91,61%	94,00%	95,82%	84,70%	92,50%	94,17%	95,48%	80,32%	90,37%	92,47%	94,50%	94,97%	98,77%	99,32%	99,61%	79,86%	90,46%	92,56%	94,74%	92,02%
Support Vector Machines	85,82%	91,25%	92,98%	94,93%	87,70%	92,28%	93,64%	94,93%	81,05%	88,67%	91,20%	93,54%	94,09%	98,34%	99,15%	99,51%	77,28%	89,91%	90,89%	93,97%	91,56%
Logistic Regression	48,95%	67,21%	72,20%	76,10%	74,65%	87,94%	90,03%	91,81%	66,81%	80,49%	85,36%	89,12%	82,02%	89,90%	91,57%	95,81%	47,56%	71,89%	76,80%	81,07%	78,37%
XGBoost	85,84%	92,64%	94,75%	96,38%	86,05%	93,05%	94,46%	95,65%	82,01%	90,16%	92,97%	94,85%	95,24%	98,77%	99,40%	99,66%	81,95%	90,83%	93,10%	94,92%	**92,63%**
Stacking (Meta-Model)	86,11%	92,27%	94,55%	96,15%	85,96%	92,96%	94,47%	95,68%	82,17%	90,70%	92,96%	94,77%	94,95%	98,92%	99,43%	99,67%	81,02%	90,67%	92,82%	94,86%	92,55%
Neural Network	86,14%	91,75%	94,18%	96,01%	87,78%	92,52%	94,13%	95,46%	80,40%	89,51%	92,00%	94,23%	93,50%	98,52%	99,13%	99,45%	76,52%	90,64%	92,85%	94,65%	91,97%
Ø	72,50%	82,10%	85,96%	89,47%	77,09%	86,35%	89,30%	92,00%	71,56%	83,16%	87,26%	90,84%	83,70%	90,72%	93,03%	**95,31%**	68,84%	83,48%	86,88%	90,28%	

- XGBoost schneidet am besten ab mit im Schnitt 92,63% gefolgt von Stacking, Random Forest, Neural Network und Support Vector Machines. Logistische Regression und Naive Bayes fallen ab
- Segment-Angriffe am besten erkannte Angriffsart vor Random-Angriffen
- Verbesserte Ergebnisse mit steigender Angriffsgröße, größte Verbesserung zwischen Angriffsgrößen 1% und 5%

Auswertung der Angriffserkennung beim Yelp Datensatz (F1-Score)

universität koblenz
weiter:denken

Angriffstyp	Bandwagon				Random				Average				Segment				Reverse Bandwagon				Ø
Angriffsgröße	1%	5%	10%	20%	1%	5%	10%	20%	1%	5%	10%	20%	1%	5%	10%	20%	1%	5%	10%	20%	
Naive Bayes	11,50%	32,55%	47,73%	45,13%	18,62%	41,38%	51,66%	61,62%	17,26%	33,05%	45,16%	59,84%	13,25%	27,72%	35,00%	64,20%	31,84%	40,68%	49,53%	56,31%	39,20%
K-Nearest-Neighbor	81,63%	88,77%	91,56%	94,12%	49,48%	73,87%	82,37%	88,46%	46,95%	71,75%	80,22%	87,12%	82,34%	92,26%	95,44%	97,57%	86,62%	93,77%	95,73%	97,31%	83,87%
Decision Tree	74,57%	83,11%	87,09%	91,07%	43,91%	69,01%	77,80%	84,17%	35,15%	64,19%	74,82%	82,30%	74,89%	88,68%	93,49%	96,38%	77,06%	89,05%	93,04%	95,45%	78,76%
Random Forest	83,03%	90,97%	93,77%	95,78%	54,15%	76,97%	84,33%	89,83%	48,46%	74,36%	82,10%	88,61%	85,42%	95,09%	96,76%	98,04%	90,40%	96,37%	97,59%	98,52%	86,03%
Support Vector Machines	71,13%	82,12%	86,38%	91,56%	30,00%	66,87%	76,52%	83,93%	23,24%	62,38%	72,62%	82,29%	72,88%	90,16%	94,38%	97,86%	75,70%	91,96%	93,99%	95,60%	77,08%
Logistic Regression	0,00%	6,64%	23,34%	37,36%	0,44%	22,74%	43,38%	57,79%	0,00%	10,43%	38,25%	55,44%	0,00%	16,08%	34,46%	44,59%	3,85%	70,79%	75,02%	80,54%	31,06%
XGBoost	82,15%	90,74%	93,37%	95,47%	41,83%	75,74%	83,24%	88,74%	36,39%	72,09%	81,05%	87,54%	82,31%	94,49%	96,67%	98,11%	92,63%	96,76%	97,89%	98,71%	84,30%
Stacking (Meta-Model)	83,68%	91,12%	93,76%	95,77%	57,17%	76,59%	83,92%	89,59%	54,23%	74,16%	81,71%	88,32%	85,47%	94,94%	97,33%	98,40%	92,47%	96,81%	97,89%	98,75%	**86,60%**
Neural Network	80,35%	90,72%	93,75%	95,94%	39,68%	77,67%	85,69%	90,89%	35,54%	74,51%	83,58%	89,60%	87,39%	95,47%	97,58%	98,86%	87,87%	95,99%	97,91%	98,74%	84,89%
Ø	63,12%	72,97%	78,97%	82,47%	37,25%	64,54%	74,32%	81,67%	33,02%	59,66%	71,06%	80,12%	64,88%	77,21%	82,35%	88,22%	70,94%	85,80%	88,73%	**91,10%**	

- Stacking-Ansatz liefert die besten Ergebnisse, knapp vor Random Forest. Ebenfalls gute Werte von im Schnitt über 80% bei Neural Network, XGBoost und K-Nearest Neighbor. Erneut fallen Logistische Regression und Naive Bayes ab

- Bei Nuke-Angriffsart Reverse Bandwagon werden die besten Ergebnisse erzielt → vermutlich auf die Bewertungsverteilung im Yelp-Datensatz zurückzuführen: Hier sind Profile mit mehrheitlich negativen Bewertungen seltener

- Auch hier verbesserte Ergebnisse mit steigender Angriffsgröße

Auswertung der Angriffserkennung beim MovieLens 32M Datensatz (F1-Score)

universität koblenz
weiter:denken

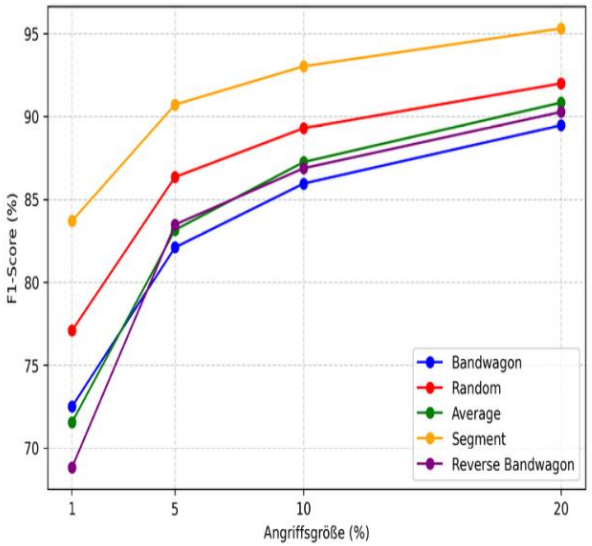
Angriffstyp	Bandwagon				Random				Average				Segment				Reverse Bandwagon				Ø
Angriffsgröße	1%	5%	10%	20%	1%	5%	10%	20%	1%	5%	10%	20%	1%	5%	10%	20%	1%	5%	10%	20%	
Naive Bayes	86,51%	94,33%	95,80%	96,41%	91,13%	97,57%	98,33%	98,92%	89,26%	97,11%	98,27%	98,79%	84,13%	95,21%	97,11%	97,85%	81,71%	92,70%	94,81%	96,03%	94,10%
K-Nearest-Neighbor	97,90%	99,11%	99,35%	99,53%	98,67%	99,52%	99,64%	99,79%	98,45%	99,52%	99,69%	99,81%	99,55%	99,93%	99,98%	99,98%	97,80%	98,96%	99,38%	99,57%	99,31%
Decision Tree	97,99%	99,10%	99,29%	99,53%	98,48%	99,49%	99,66%	99,78%	98,47%	99,54%	99,67%	99,79%	99,78%	99,95%	99,98%	99,97%	97,99%	98,99%	99,37%	99,55%	99,32%
Random Forest	98,55%	99,39%	99,58%	99,77%	99,28%	99,83%	99,93%	99,95%	99,13%	99,88%	99,93%	99,95%	99,85%	99,96%	99,99%	99,99%	98,69%	99,35%	99,61%	99,77%	99,62%
Support Vector Machines	98,48%	99,48%	99,70%	99,80%	98,60%	99,60%	99,78%	99,86%	98,65%	99,55%	99,79%	99,86%	99,43%	99,96%	99,98%	99,98%	98,21%	99,49%	99,71%	99,81%	99,49%
Logistic Regression	93,69%	98,17%	98,75%	99,16%	98,45%	98,89%	98,92%	99,18%	98,68%	99,08%	99,17%	99,32%	99,38%	99,92%	99,95%	99,98%	96,08%	97,78%	98,54%	98,95%	98,60%
XGBoost	98,53%	99,48%	99,68%	99,83%	99,23%	99,82%	99,92%	99,95%	99,50%	99,88%	99,95%	99,96%	99,73%	99,94%	99,98%	99,99%	98,60%	99,58%	99,74%	99,86%	99,66%
Stacking (Meta-Model)	98,65%	99,51%	99,66%	99,83%	99,45%	99,88%	99,93%	99,96%	99,48%	99,91%	99,95%	99,97%	99,83%	99,97%	99,99%	100,00%	98,90%	99,56%	99,72%	99,86%	**99,70%**
Neural Network	98,63%	99,47%	99,68%	99,81%	98,93%	99,70%	99,90%	99,93%	99,13%	99,81%	99,91%	99,95%	99,68%	99,91%	99,97%	99,99%	98,36%	99,54%	99,75%	99,84%	99,59%
Ø	96,55%	98,67%	99,05%	99,30%	98,02%	99,37%	99,56%	99,70%	97,86%	99,36%	99,59%	99,71%	97,93%	99,42%	99,66%	**99,75%**	96,26%	98,44%	98,96%	99,25%	

- Fast durchgehend bei allen Kombinationen von Angriffsarten und Angriffsgrößen F1-Score Werte von über 99%. Naive Bayes und Logistische Regression fallen auch hier leicht ab.

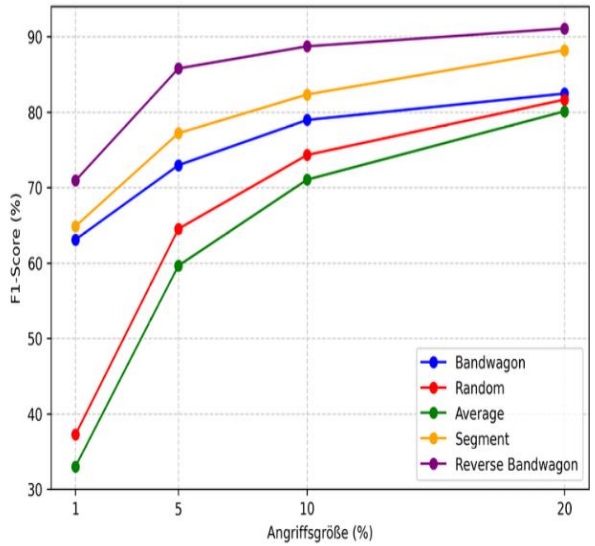
- Herausragende Ergebnisse der Angriffsdetektion vermutlich auf Dichte und Größe des MovieLens 32M Datensatzes zurückzuführen → Algorithmen gelingt es die Daten anhand der berechneten Attribute optimal zwischen den beiden Klassen (0 = normale Nutzer und 1 = Angreifer) zu trennen

Bedeutung von Angriffsgrößen & Angriffsarten

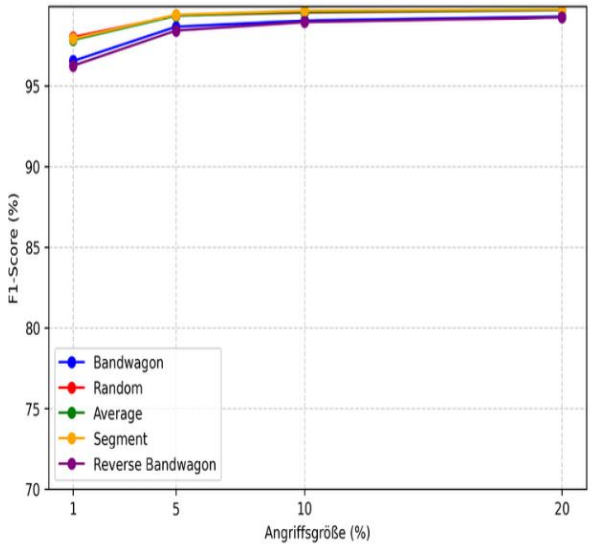
BookCrossing:



Yelp:



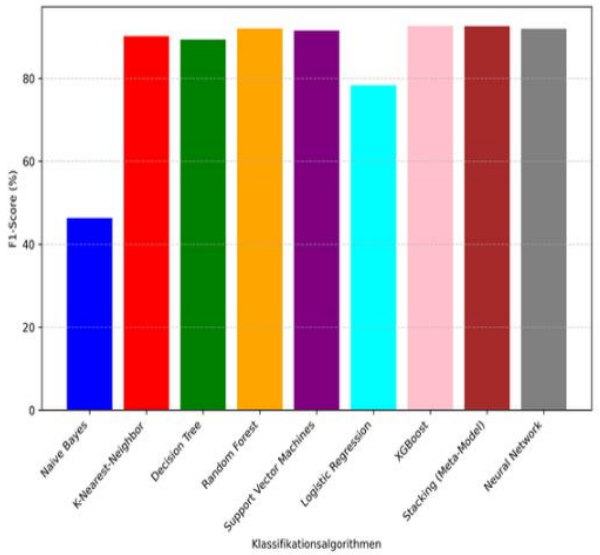
MovieLens 32M:



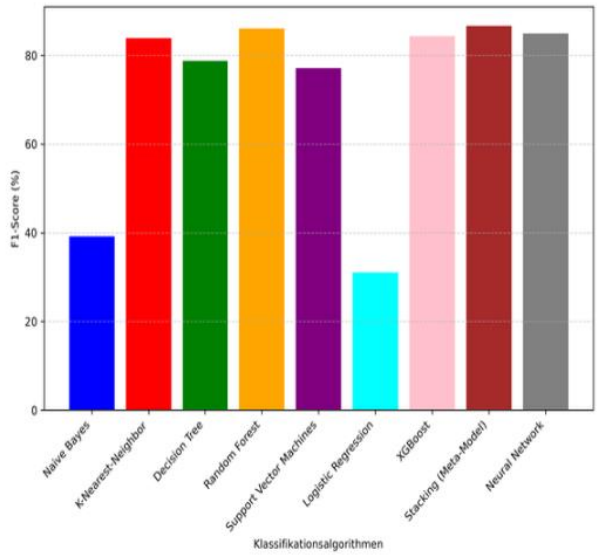
- Bei allen drei Datensätzen steigende F1-Score Werte mit zunehmender Angriffsgröße, Verbesserung nimmt mit zunehmenden Angriffsgrößen ab.
- Datensatz-übergreifend sehr gute Erkennung von Segment-Angriffen. Keine Art fällt signifikant ab

Bedeutung von Klassifikationsalgorithmen

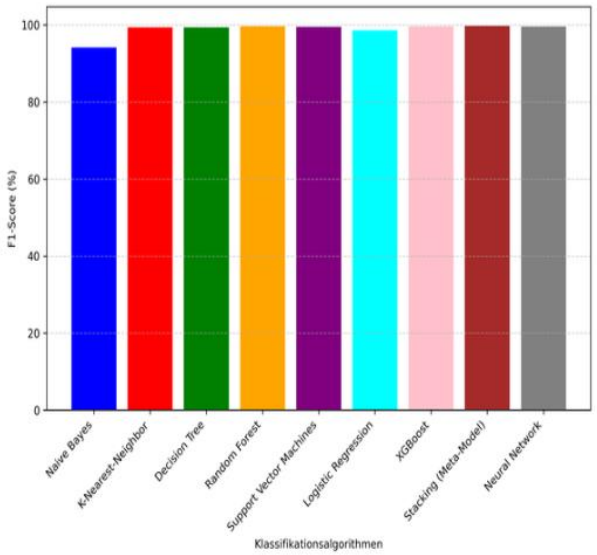
BookCrossing:



Yelp:

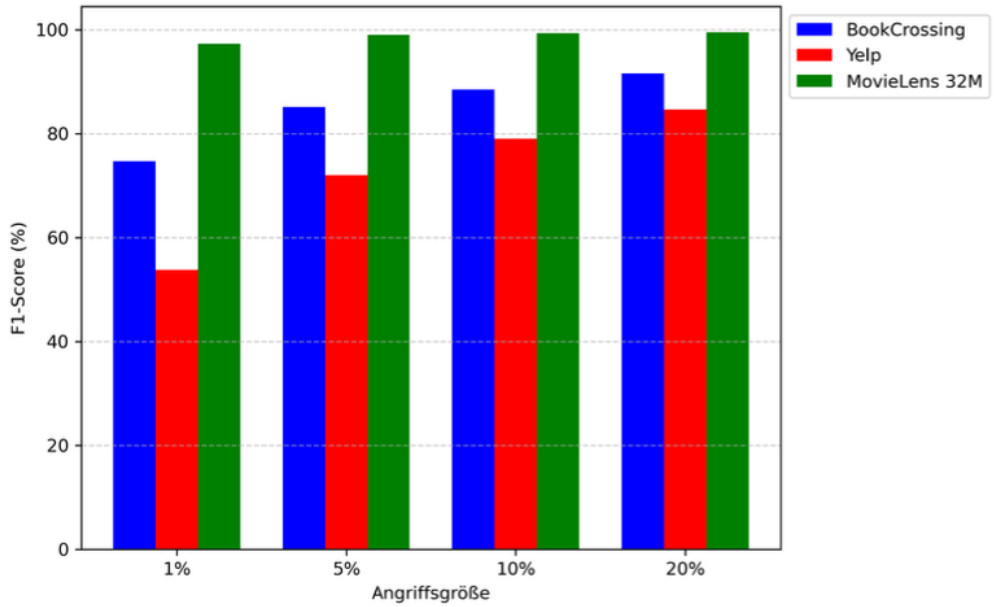


MovieLens 32M:



- Detektionsergebnisse vieler Algorithmen auf ähnlichem Niveau. Beste Ergebnisse mit Random Forest, XGBoost und Neural Network. Stacking-Ansatz verbessert diese noch leicht
- Naive Bayes und Logistische Regression fallen in den Ergebnissen ab, vor allem bei den Datensätzen von Yelp und BookCrossing

Bedeutung angewandter Datensätze



- Beste Resultate bei allen Angriffsgrößen beim MovieLens 32M Datensatz, gefolgt vom BookCrossing Datensatz. Werte beim Yelp Datensatz am schwächsten
- Es zeigt sich, dass mit dem größten und dichtesten Datensatz die besten Ergebnisse erzielt werden, der Datensatz Yelp mit der geringsten Dichte liefert die schwächsten Ergebnisse.

Vergleich der Ergebnisse mit verwandten Arbeiten bezüglich Angriffsgrößen (F1-Score in %)

universität koblenz
weiter:denken

S	BC	Y	ML	[70]	[71]	[40]	[72]	[73]	[74]	[75]	[76]
1%	95,24	92,63	99,85		24	96,80				97,25	96,73
5%	98,92	96,81	99,97	98,6	76	95,99	94		100	97,49	89,43
10%	99,43	97,91	99,99	98,5	94	96,94		99,32	100		
20%	99,67	98,86	100	98,6				100			

S = Angriffsgröße
BC = BookCrossing
Y = Yelp
ML = MovieLens 32M

- In den meisten anderen Studien [71,73, 75] ebenfalls verbesserte Werte mit steigender Angrifffsgröße und deutlichere relative Verbesserung zwischen Angriffgrößen 1% und 5% als zwischen 5% und 10% festzustellen
- Im Vergleich schwache Werte bei [71]: wahrscheinlich wegen Verwendung eines nicht überwachten Ansatzes
- Bestwerte bei [70] und [40] ebenfalls wie in dieser Arbeit bei höchster Angriffsgröße

Vergleich der Ergebnisse mit verwandten Arbeiten bezüglich Angriffsarten (F1-Score in %)

universität koblenz
weiter:denken

T	BC	Y	ML	[73]	[71]	[77]	[72]	[74]	[70]	[78]	[75]
B	96,38	95,95	99,83	99,11	96	97,14	94	100	98,6		
R	95,68	90,89	99,96	99,09	96	97,30	91	100		98,5	97,36
A	94,85	89,60	99,97	99,14	93	97,46	91	100	98,6	98,5	97,49
S	99,67	98,86	100								
RB	94,92	98,75	99,86					100			

T = Angriffstyp
(R)B = (Reverse) Bandwagon
R = Random
A = Average
S = Segment

- In verwandten Studien ähnlich wie in dieser Arbeit keine signifikanten Unterschiede zwischen ArtenBandwagon, Random und Average
- Wie beim ML32M auch in [73], [75] und [77] Bestwerte bei Average-Angrifen erzielt
- Etwas schwächere Werte als in dieser Arbeit bei [72]: vemutlich weil Training/Test-Verhältnis in [72] 70/30 statt wie sonst meistens 80/20
- Bestwerte von 100% in [74] vermutlich wegen Ansatz basierend auf komplexer Verhaltensanalyse und Verwendung großer, dichter Datensätze (Netflix, MovieLens 1M)

Vergleich der Ergebnisse mit verwandten Arbeiten bezüglich Algorithmen (F1-Score in %)

- Werte in ähnlicher Größenordnung bei dieser Arbeit und verwandten Studien → fast immer F1-Score > 95%
- Bessere Ergebnisse mit Stacking-Ansatz als mit einzelnen Algorithmen bei [40] und [79] → genau wie in dieser Arbeit
- In [76] und [80] Werte von stets über 98% mit XGBoost → wie in dieser Arbeit
- Werte von über 98% mit NB in [70] und [78] wie in dieser Arbeit mit ML32M → in [70] und [78] ebenfalls dichte Datensätze von MovieLens genutzt

C	BC	Y	ML	[70]	[78]	[79]	[40]	[76]	[80]
NB	72,32	64,20	98,92	98,6	98		96,59		64
KNN	98,85	97,57	99,99				92,59		
DT	99,31	96,38	99,98					98,02	
RF	99,61	98,52	100	98,6	98		94,59		
SVM	99,51	97,86	99,99			98,2		95,43	63
LR	95,81	80,54	99,98			97,8			64
XGB	99,66	98,71	99,99				94,41	98,79	98
STA	99,67	98,75	100	98,6	98	99	96,97		
NN	99,45	98,86	99,99			98,2			

C = Classifier, NB = Naive Bayes, KNN = K-Nearest Neighbor, DT = Decision Tree, RF = Random Forest, SVM = Support Vector Machines, LR = Logistische Regression, XGB = XGBoost, STA = Stacking, NN = Neural Network

Vergleich mit weiteren Studien aus der Literatur

- Zhang und Zhou, 2013 [81]:
  - mit auf mehreren Back Propagation Neural Networks basierendem Ansatz Werte von bis knapp unter 100\% bei der Precision erreicht
  - Vergleichbar mit dieser Arbeit konnten die Forscher ebenfalls jeweils ähnliche Ergebnisse für die drei Angriffsarten Random, Bandwagon und Average erzielen.
  - Weitere Gemeinsamkeit: auch in [81] zu beobachten, dass die Precision-Werte sich mit steigender Angriffsgröße verbessern, wobei die Unterschiede zwischen den Angriffsgrößen 1\% und 5\% größer als zwischen den Angriffsgrößen 5\% und 10\% sind.
  - Keine deutlichen Unterschiede zwischen Push- und Nuke-Bandwagon-Angriffen festzustellen, wie in dieser Arbeit ebenfalls zu beobachten
- Zhou et al., 2016 [82]:
  - erreichten mit ihrem auf SVM & Gruppen-Eigenschaften in Angriffsprofilen basierenden Ansatz Precision von je 99,5\% bei Random-, Bandwagon, Reverse Bandwagon-Angriffen. Recall-Werte bei allen Angriffsarten etwas schwächer, zwischen 90\% und 95\%.
  - Da in der Studie [82] der MovieLens 100K Datensatz verwendet → Vergleich mit ML32M aus dieser Arbeit
  - Gemeinsamkeit: auch hier Precision leicht über Recall und keine signifikanten Unterschiede zwischen verschiedenen Angriffsarten. Bei den Angriffsgrößen ähnlich wie in dieser Arbeit auch in [82] Werte mit zunehmender Angriffsgröße verbessert, bei Recall stärker als Precision

Vergleich mit weiteren Studien aus der Literatur

- Singh et al., 2021 [83]:
  - Nutzten verschiedene Algorithmen des überwachten Lernens, um basierend auf dem MovieLens 100K Datensatz simulierte Shilling-Angriffe einer Angriffsgröße von 25\% erfolgreich zu erkennen.
  - Erreichten Bestwerte bei der Precision von 99,7\% bei Random- und Bandwagon-Angriffen, 100\% bei Average-Angriffen und 99,8\% bei Segment-Angriffen → wie in dieser Arbeit Segment-Angriffe sehr zuverlässig erkannt
  - Keine wesentlichen Unterschiede zwischen den untersuchten Angriffsarten, so wie es auch in dieser Arbeit der Fall ist
- Yang, 2015 [84]:
  - Stellte einen Ansatz vor, in dem analysiert wurde, wie gut es möglich ist, Angriffe auf Recommender Systeme zu erkennen, wenn die Target-Items leicht über- bzw. unterdurchschnittlich statt maximal bzw. minimal bewertet werden.
  - Unter anderem ein Datensatz von BookCrossing verwendet, so wie es in dieser Arbeit ebenso gemacht wurde
  - Studie [84] hat mit dieser Arbeit gemein, dass es keine signigfikanten Unterschiede zwischen der Erkennung von Bandwagon- und Reverse Bandwagon-Angriffen gibt
  - Sowohl bei [84] als auch in dieser Arbeit zu beobachten, dass Segment-Angriffe gute Detektionsergebnisse liefern.

Vergleich mit weiteren Studien aus der Literatur

- Zhou et al., 2018 [85]:
  - stellten einen Ansatz vor, um basierend auf verdächtigen Zeitfenstern, einer Analyse von Target-Items und der Glaubwürdigkeit von Nutzergruppen Shilling-Angriffe zu erkennen
  - Forscher nutzten dafür, wie es in dieser Arbeit ebenso gemacht wurde, mehrere Datensätze
  - Gemeinsamkeit mit den Klassifikationsergebnissen dieser Arbeit: bei [85] stets bessere Performance bei größeren und dichteren Datensätze als bei kleineren, weniger dichten Datensätzen → Bessere Performance bei MovieLens 10M als beim Datensatz MovieLens1M und bei diesem wiederum eine bessere als beim MovieLens 100K beobachtet
  - Weitere Gemeinsamkeit mit den Ergebnissen dieser Arbeit: auch in [85] beobachtet, dass Ergebnisse mit steigender Angriffsgröße verbessert, mit größeren Verbesserungen bei kleinen Angriffsgrößen

Lösungsansätze für Schwachstellen bei der Empfehlungsqualität

Nr.	Schwachstelle/Risiko	Beschreibung	Lösungsvorschlag
1	Doppelte Bewertungen	Nutzer kann dasselbe Item mehrmals bewerten	Beschränkung, dass jeder Nutzer jedes Item nur einmal bewerten kann, implementieren; Umsetzung durch Hashing der Nutzer-Item-Kombinationen
2	Extrem hohe Bewertungsanzahl	Nutzer kann beliebig viele Bewertungen abgeben	Bewertungsanzahl begrenzen, z.B. 3 am Tag oder 10 im Monat
3	Bewertungsmöglichkeit direkt nach Beitritt	Nutzer kann direkt nach Erstellung seines Accounts Bewertungen abgeben	Kriterien für Bewertungsmöglichkeit: Account ist min. 30 Tage alt oder Nutzer hat mindestens 5 Käufe
4	Bewertungsmöglichkeit für nicht gekaufte Produkte	Nutzer kann Produkte bewerten, ohne diese gekauft zu haben	Implementation einer Datenbank-basierten Überprüfung der Kaufhistorie: Nur Käufer können bewerten
5	Bewertungsmöglichkeit für vor Erhalt stornierte Bestellungen	Nutzer kann Produkte bewerten, obwohl er die Bestellungen vor Erhalt der Produkte storniert hat	Funktion implementieren, die prüft, ob ein Nutzer eine Bestellung storniert hat, bevor er eine Bewertung zu einem in der Bestellung enthaltenen Produkt abgeben kann
6	Extrem einseitige Bewertungen	Nutzer bewertet wenig positiv, dafür aber ganz viel negativ	Automatischer Hinweis an Admins wenn mindestens 80\% der Bewertungen eines Nutzers negativ sind
7	Manipulierte Bewertungen durch Angriffe	Angreifer beeinflussen das Empfehlungssystem durch Massenbewertungen aus verschiedenen Accounts	Implementation ML-basierter automatisierter Anomalieerkennung. Kontinuierliches Monitoring von Maßen wie MAE, RMSE und HitRate@K zur Beobachtung von Veränderungen bei der Empfehlungsqualität
8	Overfitting	Empfehlungen des Systems basieren zu sehr auf bereits vorhandenen Daten und sind für neue Nutzer ungenau	Regelmäßiges Training des Recommender Modells basierend auf den gesamten vorhandenen Bewertungsdaten; Cross-Validation beim Trainieren des Modells

Lösungsansätze für Schwachstellen bei der Empfehlungsqualität

Nr.	Schwachstelle/Risiko	Beschreibung	Lösungsvorschlag
9	Für Nutzer nicht nachvollziehbare Empfehlungen	Nutzer können Empfehlungen nicht verstehen, da sie sie als unpassend erachten	Einsatz von Explainable AI (XAI) zur Erklärung und Visualisierung von Gründen für die Empfehlungen
10	Cold-Start	Für neue Nutzer oder neue Items können aufgrund fehlender Daten keine Empfehlungen generiert werden	Einsatz von Content-based Filtering sowie Hybrid-Methoden; Abfrage von Genre-/Kategorie-Präferenzen bei neuen Nutzern
11	Nichterkennung von Angreifern	Angriffe bzw. Angriffsprofile werden nicht erkannt	Nicht überwachtes Lernen zur Erkennung von untypischen Bewertungsmustern
12	Nutzerfeedback	Nutzer sollten durch Feedback aktiv die Möglichkeit zur Verbesserung der Empfehlungsqualität des Recommender Systems haben, um falsche oder irrelevante Empfehlungen zu minimieren	Implementierung einer „Kein Interesse“-Funktion zur Verbesserung des Systems
13	Shilling-Angriffe	Angreifer versuchen möglichst unauffällig durch Fake-Bewertungen die Empfehlungen des Recommender Systems zu manipulieren	Verwendung von Attributen wie RDMA, WDMA zur Detektion auffälliger Bewertungsschemata; Manuelle Überprüfung potenzieller Angriffsprofile durch Admins
14	Sybil-Angriffe	Angreifer versuchen, von mehreren Accounts aus durch gezielt verfälschende Bewertungen die Empfehlungsqualität des Systems negativ zu beeinflussen	Nutzung von Graph-basierten Algorithmen, um Gruppen-Angriffe zu erkennen
15	Bot-Bewertungen	Bots bewerten Items und beeinflussen so die Empfehlungsqualität des Systems	Implementation gezielter Maßnahmen wie 2-Faktor-Authentifizierung oder Captchas gegen Bots
16	Fehlende Gewichtung von vertrauenswürdigen Nutzern	Bewertungen werden gleich behandelt, unabhängig davon, ob ein Nutzer vertrauenswürdig ist oder nicht	Einführung eines nutzerbasierten Vertrauenssystems: Nutzer können anderen Nutzern Vertrauen aussprechen, basierend auf Bewertungen, Verifizierung oder Community-Feedback; Bewertungen von vertrauenswürdigen Nutzern erhalten ein höheres Gewicht im Empfehlungsalgorithmus

Lösungsansätze für technische Schwachstellen

Nr.	Schwachstelle/Risiko	Beschreibung	Lösungsvorschlag
1	Störungen des Systems	Das System könnte durch verschiedene Faktoren gestört werden wie bspw. eine hohe Anzahl an Anfragen, eine Vielzahl neuer Nutzer oder Bewertungen auf einmal	Es sollte ein Monitoring-System implementiert werden, das dauerhaft die Aktivität (z.B. Nutzerauslastung, neue Bewertungen in der letzten Stunde, etc.) aufzeichnet und protokolliert und so ermöglicht, Unregelmäßigkeiten automatisch und frühzeitig zu erkennen
2	Mehrere Rezensionen aus dem selben Netzwerk (IP)	Von der selben IP bzw. aus dem selben Netzwerk können mehrere Accounts am Tag erstellt und so Bewertungen abgegeben werden	Funktion implementieren, die überprüft, ob aus dem selben Netzwerk bereits ein Account im System aktiv ist; maximal 5 Accounts aus dem selben Netzwerk zulassen, ab dann Erstellung weiterer Accounts verhindern
3	Skalierbarkeit des Systems	Systeme können an ihre technischen Grenzen stoßen und ungenauer werden, wenn die Nutzer- und Bewertungsanzahl exorbitant ansteigt	Einsatz von Cloud-Technologien als bestmögliche Hardwareunterstützung für hohe Skalierbarkeit; regelmäßiges Training des Modells auf aktuellen Daten
4	Rechenanforderungen	Recommender Algorithmen (z.B. ALS, SVD) benötigen viel Rechenleistung, um schnell Empfehlungen für alle Nutzer zur Verfügung stellen zu können	Einsatz von GPUs oder TPUs, um die benötigte hohe Rechenlast bereitzustellen
5	DDoS-Angriffe	Angreifer überlasten das System mit einer sehr hohen Anzahl an Anfragen und schränken so die Verfügbarkeit ein	Möglichst hoch skalierbare Infrastruktur, z.B. durch den Einsatz von Cloud-Technologie; Spezielle DDoS-Abwehr Mechanismen implementieren

Lösungsansätze für Schwachstellen bezüglich Datenschutz

Nr.	Schwachstelle/Risiko	Beschreibung	Lösungsvorschlag
1	Rückschluss auf persönliche Daten	Daten aus dem Recommender System lassen Rückschlüsse auf persönliche Daten des Nutzers zu	Transparent darstellen, welche Daten der Nutzer verwendet werden und wofür. Differential Privacy und Verschlüsselung von Nutzerdaten. Nutzern die Anonymisierung ihrer Daten ermöglichen.
2	Kommerzielle Verwendung von Nutzerdaten	Daten aus dem Recommender System werden ohne Zustimmung der Nutzer kommerziell verwendet	Daten dürfen nicht kommerziell verwendet werden
3	Social Engineering	Angreifer täuschen Mitarbeiter, um Zugriff zum System und auf interne Daten zu erhalten	Zwei-Faktor-Authentifizierung auch für Mitarbeiter. Regelmäßig verpflichtende Sicherheits-Trainings und spezielle Angriffs-Schulungen für Mitarbeiter

Zusammenfassung und zukünftige Forschungsansätze

Zusammenfassung

- Recommender Systeme fester Bestandteil des Alltags vieler Menschen → Bewusstsein für Schwachstellen und Sicherheitsherausforderungen sowie Privatsphäre und Datenschutz essenziell

- Shilling-Angriffe können Empfehlungen verzerren und zu sinkenden Erträgen und geringerer Kundenzufriedenheit führen → Erkennung nötig, um Recommender Systeme zu schützen und deren Resilienz zu erhöhen

- Vor allem Segment-Angriffe einer Größe von 20% verzerren die Empfehlungen, werden allerdings auch am zuverlässigsten erkannt. Besonders mit Algorithmen Random Forest, XGBoost und Neural Network robuste Erkennung → mit Stacking-Ansatz noch leicht verbesserte Performance

- Zuverlässige Erkennung auch bei Datensätzen mit geringerer Dichte und dynamisch angepasster Filler-Anzahl möglich

- Vorgestellter Ansatz erreicht state-of-the-art Ergebnisse und übertrifft diese teilweise

Limitationen

- Die Recommender Systeme basieren auf dem ALS-Algorithmus → die Performance der Angriffserkennung könnte bei auf anderen Algorithmen basierenden Systemen anders ausfallen

- Angreifer können neue Angriffsarten mit bisher unbekannten Bewertungsmustern verwenden → unklar, ob mit dem vorgestellten Ansatz auch solche zuverlässig erkannt werden

- Die Beurteilung der Performance der Recommender Systeme in dieser Arbeit basiert auf historischen Daten → eine Online-Auswertung mit realen Nutzerinteraktionen wäre sinnvoll, um die Qualität der Empfehlungen vollumfänglich beurteilen zu können

- Weitere Angriffe neben Shilling-Angriffen können Recommender Systemen schaden → auch solche müssen erkannt werden

Zukünftige Forschungsansätze & Code der Arbeit

- Auf dem vorgestellten Ansatz dieser Arbeit kann aufgebaut werden:
  - Angriffssimulation auf Recommender Systeme, deren Empfehlungen auf anderen Algorithmen als dem ALS-Algorithmus basieren
  - Online-Auswertung mit realen Nutzerinteraktionen, um die Qualität der Empfehlungen noch fundierter beurteilen zu können
  - Entwicklung neuer Features zur Angriffserkennung, um Performance weiter zu verbessern
  - Erforschung weiterer, neuer Arten von Shilling-Angriffen
  - Anwendung von Deep Learning Ansätzen wie tiefer neuronaler Netze, um die Performance der Angriffsdetektion möglicherweise noch weiter zu verbessern
  - Implementation von Schutzmaßnahmen, um die Resilienz von Recommender Systemen gegen andere Angriffe als Shilling-Angriffe sicherzustellen

- Zu Forschungs- und Anwendungszwecken wird der im Rahmen der Masterarbeit implementierte Code hier zur Verfügung gestellt: https://github.com/aaronpasternak/Sicherheit-von-Recommender-Systems/tree/main/Code

Danke für die Aufmerksamkeit!

Fragen?

Literatur

[1] C. C. Aggarwal, Recommender Systems: The Textbook. Cham, Schweiz: Springer, 2016.
[2] D. Jannach, M. Zanker, A. Felfernig, and G. Friedrich, Recommender Systems: An Introduction, 1st ed. USA: Cambridge University Press, 2010.
[3] E. Rich, "User modeling via stereotypes," Cognitive Science, vol. 3, no. 4, pp. 329–354, 1979. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0364021379800129
[4] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry, "Using collaborative filtering to weave an information tapestry," Communications of the ACM, vol. 35, no. 12, pp. 61–70, Dec. 1992. [Online]. Available: https://doi.org/10.1145/138859.138867
[5] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "Grouplens: an open architecture for collaborative filtering of netnews," in Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work, ser. CSCW '94. New York, NY, USA: Association for Computing Machinery, 1994, pp. 175–186. [Online]. Available: https://doi.org/10.1145/192844.192905
[6] B. Smith and G. Linden, "Two decades of recommender systems at amazon.com," IEEE Internet Computing, vol. 21, no. 3, pp. 12–18, 2017.
[7] P. Covington, J. Adams, and E. Sargin, "Deep neural networks for youtube recommendations," in Proceedings of the 10th ACM Conference on Recommender Systems, ser. RecSys '16. New York, NY, USA: Association for Computing Machinery, 2016, pp. 191–198. [Online]. Available: https://doi.org/10.1145/2959100.2959190
[8] F. Meggetto, C. Revie, J. Levine, and Y. Moshfeghi, "Why people skip music? on predicting music skips using deep reinforcement learning," in Proceedings of the 2023 Conference on Human Information Interaction and Retrieval, ser. CHIIR '23. New York, NY, USA: Association for Computing Machinery, Mar. 2023, pp. 95–106. [Online]. Available: https://doi.org/10.1145/3576840.3578312

Literatur

[9] E. Cano and M. Morisio, “Hybrid recommender systems: A systematic literature review,” Intelligent Data Analysis, vol. 21, no. 6, pp. 1487–1524, Nov. 2017. [Online]. Available: http://dx.doi.org/10.3233/IDA-163209
[10] V. Anand and A. K. Maurya, “A survey on recommender systems using graph neural network,” ACM Transactions on Information Systems, vol. 43, no. 1, pp. 9–9:49, Nov. 2024. [Online]. Available: https://doi.org/10.1145/3694784
[11] R. Burke, B. Mobasher, and R. Bhaumik, “Limited knowledge shilling attacks in collaborative filtering systems,” in Proceedings of 3rd International Workshop on Intelligent Techniques for Web Personalization (ITWP 2005), Edinburgh, Schottland, Jan. 2005, pp. 1–8.
[12] E. Cano and M. Morisio, “Characterization of public datasets for recommender systems,” in 2015 IEEE 1st International Forum on Research and Technologies for Society and Industry Leveraging a better tomorrow (RTSI). IEEE, 2015, pp. 249–257.
[13] L. Sharma and A. Gera, “A survey of recommendation system: Research challenges,” International Journal of Engineering Trends and Technology (IJETT), vol. 4, no. 5, pp. 1989–1992, May 2013. [Online]. Available: http://www.ijettjournal.org
[14] P. Melville, R. J. Mooney, and R. Nagarajan, “Content-boosted collaborative filtering for improved recommendations,” in Eighteenth National Conference on Artificial Intelligence. USA: American Association for Artificial Intelligence, 2002, pp. 187–192.
[15] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, “Application of dimensionality reduction in recommender system – a case study,” University of Minnesota Department of Computer Science and Engineering, Boston, Massachusetts, USA, Tech. Rep. ADA439541, Aug. 2000.
[16] S. S. Lakshmi and T. A. Lakshmi, “Recommendation systems: Issues and challenges,” International Journal of Computer Science and Information Technologies, vol. 5, no. 4, pp. 5771–5772, 2014.

Literatur

[17] S. Khusro, Z. Ali, and I. Ullah, “Recommender systems: Issues, challenges, and research opportunities,” in Information Science and Applications (ICISA) 2016, ser. Lecture Notes in Electrical Engineering, K. J. Kim and N. Joukov, Eds., vol. 376. Singapore: Springer Singapore, 2016, pp. 1179–1189.
[18] M. Yang, J. Wang, and J.-F. Ton, “Rectifying unfairness in recommendation feedback loop,” in Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, ser. SIGIR ’23. New York, NY, USA: Association for Computing Machinery, 2023, pp. 28–37. [Online]. Available: https://doi.org/10.1145/3539618.3591754
[19] S. K. Lam and J. Riedl, “Shilling recommender systems for fun and profit,” in Proceedings of the 13th International Conference on World Wide Web, ser. WWW ’04. New York, NY, USA: Association for Computing Machinery, May 2004, pp. 393–402. [Online]. Available: https://doi.org/10.1145/988672.988726
[20] J. J. Sandvig, B. Mobasher, and R. Burke, “Robustness of collaborative recommendation based on association rule mining,” in Proceedings of the 2007 ACM Conference on Recommender Systems, ser. RecSys ’07. New York, NY, USA: Association for Computing Machinery, 2007, pp. 105–111. [Online]. Available: https://doi.org/10.1145/1297231.1297249
[21] P. Melville and V. Sindhwani, Recommender Systems. Berlin: Springer, 2010, pp. 829–838.
[22] M. Claypool, A. Gokhale, T. Miranda, P. Murnikov, D. Netes, and M. Sartin, “Combining content-based and collaborative filters in an online newspaper,” in Proceedings of ACM SIGIR Workshop on Recommender Systems, Berkeley CA, USA, Aug. 1999.
[23] A. J. P. Jeckmans, M. Beye, Z. Erkin, P. Hartel, R. L. Lagendijk, and Q. Tang, Privacy in Recommender Systems, ser. Computer Communications and Networks. Springer, London, 2013.
[24] Z. Sun, Z. Wang, and Y. Xu, “Privacy protection in cross-platform recommender systems: techniques and challenges,” Wireless Networks, vol. 30, no. 8, pp. 6721–6730, Sep. 2023. [Online]. Available: https://doi.org/10.1007/s11276-023-03509-z

Literatur

[25] Y. Ge, S. Liu, Z. Fu, J. Tan, Z. Li, S. Xu, Y. Li, Y. Xian, and Y. Zhang, “A survey on trustworthy recommender systems,” ACM Transactions on Recommender Systems, vol. 3, no. 2, pp. 13–13:68, Nov. 2024. [Online]. Available: https://doi.org/10.1145/3652891
[26] European Union, “Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data (General Data Protection Regulation),” Available online at https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32016R0679, 2016, letzter Zugriff am 28.01.2025.
[27] H. Polat and W. Du, “Privacy-preserving collaborative filtering,” International Journal of Electronic Commerce, vol. 9, no. 4, pp. 9–35, 2005. [Online]. Available: http://www.jstor.org/stable/27751163
[28] C. Dwork, “Differential privacy,” in Proceedings of the 33rd International Colloquium on Automata, Languages and Programming (ICALP), ser. Lecture Notes in Computer Science, M. Bugliesi, B. Preneel, V. Sassone, and I. Wegener, Eds. Berlin, Heidelberg: Springer, 2006, pp. 1–12.
[29] P. Samarati and L. Sweeney, “Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression,” SRI International, Computer Science Laboratory, Technical Report SRI-CSL-98-04, 1998.
[30] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam, “L-diversity: Privacy beyond k-anonymity,” ACM Transactions on Knowledge Discovery from Data, vol. 1, no. 1, pp. 1–52, Mar. 2007. [Online]. Available: https://doi.org/10.1145/1217299.1217302
[31] N. Li, T. Li, and S. Venkatasubramanian, “t-closeness: Privacy beyond k-anonymity and l-diversity,” in 2007 IEEE 23rd International Conference on Data Engineering. IEEE, 2007, pp. 106–115.
[32] X. Yi, R. Paulet, and E. Bertino, Homomorphic Encryption and Applications, ser. SpringerBriefs in Computer Science. Cham, Heidelberg, New York, Dordrecht, London: Springer, 2014.

Literatur

[33] R. Cissée and S. Albayrak, “An agent-based approach for privacy-preserving recommender systems,” in Proceedings of the 6th International Joint Conference on Autonomous Agents and Multiagent Systems, ser. AAMAS ’07. New York, NY, USA: Association for Computing Machinery, May 2007, pp. 319–326. [Online]. Available: https://doi.org/10.1145/1329125.1329345
[34] S. Nakamoto, “Bitcoin: A peer-to-peer electronic cash system,” Available online at http://www.bitcoin.org/bitcoin.pdf, May 2008, letzter Zugriff am 02.02.2025.
[35] M. O’Mahony, N. Hurley, N. Kushmerick, and G. Silvestre, “Collaborative recommendation: A robustness analysis,” ACM Transactions on Internet Technology, vol. 4, no. 4, p. 344–377, Nov. 2004. [Online]. Available: https://doi.org/10.1145/1031114.1031116
[36] C. Williams, B. Mobasher, R. Burke, J. Sandvig, and R. Bhaumik, “Detection of obfuscated attacks in collaborative recommender systems,” in Proceedings of the ECAI’06 Workshop on Recommender Systems, Held at the 17th European Conference on Artificial Intelligence (ECAI’06), Riva del Garda, Italy, Aug. 2006. [Online]. Available: http://proserver3-iwas.uni-klu.ac.at/ECAI06-Recommender-Workshop/workshop proceedings.pdf
[37] A. P. Sundar, F. Li, X. Zou, T. Gao, and E. D. Russomanno, “Understanding shilling attacks and their detection traits: A comprehensive survey,” IEEE Access, vol. 8, pp. 171 703–171 715, 2020.
[38] C. Williams, B. Mobasher, and R. Burke, “Defending recommender systems: Detection of profile injection attacks,” Service Oriented Computing and Applications, vol. 1, no. 3, pp. 157–170, Oct. 2007.
[39] P. Kaur and S. Goel, “Shilling attack models in recommender system,” in 2016 International Conference on Inventive Computation Technologies (ICICT), vol. 2. IEEE, 2016, pp. 1–5.
[40] N. Goutham, K. Singh, L. Banda, P. Sharma, C. Verma, and S. B. Goyal, “Shad-sef: An efficient model for shilling attack detection using stacking ensemble framework in recommender systems,” International Journal of Performability Engineering, vol. 19, no. 5, pp. 291–302, 2023. [Online]. Available: https://www.ijpe-online.com/EN/abstract/article 4773.shtml

Literatur

[41] R. Burke, B. Mobasher, C. Williams, and R. Bhaumik, “Classification features for attack detection in collaborative recommender systems,” in Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ser. KDD ’06. New York, NY, USA: Association for Computing Machinery, 2006, pp. 542–547. [Online]. Available: https://doi.org/10.1145/1150402.1150465
[42] R. A. Zayed, L. F. Ibrahim, H. A. Hefny, H. A. Salman, and A. AlMohimeed, “Experimental and theoretical study for the popular shilling attacks detection methods in collaborative recommender system,” IEEE Access, vol. 11, pp. 79 358–79 369, 2023.
[43] P.-A. Chirita, W. Nejdl, and C. Zamfir, “Preventing shilling attacks in online recommender systems,” in Proceedings of the 7th Annual ACM International Workshop on Web Information and Data Management, ser. WIDM ’05. New York, NY, USA: Association for Computing Machinery, 2005, pp. 67–74. [Online]. Available: https://doi.org/10.1145/1097047.1097061
[44] W. Zhou, J. Wen, Y. S. Koh, Q. Xiong, M. Gao, G. Dobbie, and S. Alam, “Shilling attacks detection in recommender systems based on target item analysis,” PLOS ONE, vol. 10, no. 7, pp. 1–26, Jul. 2015. [Online]. Available: https://doi.org/10.1371/journal.pone.0130968
[45] R. Burke, B. Mobasher, C. Williams, and R. Bhaumik, “Detecting profile injection attacks in collaborative recommender systems,” in The 8th IEEE International Conference on E-Commerce Technology and The 3rd IEEE International Conference on Enterprise Computing, E-Commerce, and E-Services (CEC/EEE’06). IEEE, 2006, pp. 23–30.
[46] G. Rebala, A. Ravi, and S. Churiwala, An Introduction to Machine Learning, 1st ed. Cham: Springer International Publishing AG, 2019.
[47] B. Mahesh, “Machine learning algorithms-a review,” International Journal of Science and Research (IJSR), vol. 9, no. 1, pp. 381–386, 2020.
[48] A. Pandey and A. Jain, “Comparative analysis of knn algorithm using various normalization techniques,” International Journal of Computer Network and Information Security, vol. 9, no. 11, pp. 36–42, Nov. 2017.

Literatur

[49] S. Bishnoi and B. K. Hooda, “Decision tree algorithms and their applicability in agriculture for classification,” Journal of Experimental Agriculture International, vol. 44, no. 7, pp. 20–27, May 2022. [Online]. Available: https://journaljeai.com/index.php/JEAI/article/view/1967
[50] L. Breiman, “Random forests,” Machine Learning, vol. 45, no. 1, pp. 5–32, Oct. 2001. [Online]. Available: https://doi.org/10.1023/A:1010933404324
[51] A. A. Imran, M. N. Amin, and F. T. Johora, “Classification of chronic kidney disease using logistic regression, feedforward neural network and wide deep learning,” in 2018 International Conference on Innovation in Engineering and Technology (ICIET), 2018, pp. 1–6.
[52] T. Chen and C. Guestrin, “Xgboost: A scalable tree boosting system,” in Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ser. KDD ’16. New York, NY, USA: Association for Computing Machinery, 2016, pp. 785–794. [Online]. Available: https://doi.org/10.1145/2939672.2939785
[53] A. G., C. H. R., and M. Rafi, “Ensemble learning techniques for improvised image recognition accuracy,” International Journal of Progressive Research in Engineering Management and Science (IJPREMS), vol. 04, no. 01, pp. 458–467, Jan. 2024. [Online]. Available: www.ijprems.com
[54] S. Tufail, H. Riggs, M. Tariq, and A. I. Sarwat, “Advancements and challenges in machine learning: A comprehensive review of models, libraries, applications, and algorithms,” Electronics, vol. 12, no. 8, pp. 1–43, 2023. [Online]. Available: https://www.mdpi.com/2079-9292/12/8/1789
[55] S. Zhang, C. Zhang, and Q. Yang, “Data preparation for data mining,” Applied Artificial Intelligence, vol. 17, no. 5-6, pp. 375–381, 2003.
[56] D. C. Asogwa, S. O. Anigbogu, M. O. Onyesolu, and C. I. Chukwuneke, “Review of some machine learning techniques for big data, their uses, strengths and weaknesses,” International Journal of Trend in Research and Development (IJTRD), vol. 6, no. 5, pp. 215–219, Oct. 2019. [Online]. Available: http://www.ijtrd.com/papers/IJTRD20761.pdf

Literatur

[57] F. Y. Osisanwo, J. E. T. Akinsola, O. Awodele, J. O. Hinmikaiye, O. Olakanmi, and J. Akinjobi, "Supervised machine learning algorithms: Classification and comparison," International Journal of Computer Trends and Technology (IJCTT), vol. 48, no. 3, pp. 128–138, Jun. 2017. [Online]. Available: https://www.ijcttjournal.org/Volume48/number-3/IJCTT-V48P126.pdf
[58] A. L'Heureux, K. Grolinger, H. F. Elyamany, and M. A. M. Capretz, "Machine learning with big data: Challenges and approaches," IEEE Access, vol. 5, pp. 7776–7797, 2017.
[59] L. Munoz-González and E. Lupu, The Security of Machine Learning Systems. Springer Cham, May 2018, pp. 47–79.
[60] B. Mobasher, R. Burke, R. Bhaumik, and C. Williams, "Toward trustworthy recommender systems: An analysis of attack models and algorithm robustness," ACM Transactions on Internet Technology, vol. 7, no. 4, pp. 23–23:38, Oct. 2007. [Online]. Available: https://doi.org/10.1145/1278366.1278372
[61] G. Schröder, M. Thiele, and W. Lehner, "Setting goals and choosing metrics for recommender system evaluations," in Proceedings of the RecSys 2011 Workshop on Human Decision Making in Recommender Systems (Decisions@RecSys'11) and User-Centric Evaluation of Recommender Systems and Their Interfaces - 2 (UCERSTI 2). ACM, Oct. 2011, pp. 78–85. [Online]. Available: https://www.researchgate.net/publication/268381252 Setting Goals and Choosing Metrics for Recommender System Evaluations
[62] G. Rivera, R. Florencia, V. García, A. Ruiz, and J. P. Sánchez-Solís, "News classification for identifying tra"c incident points in a spanish-speaking country: A real-world case study of class imbalance learning," Applied Sciences, vol. 10, no. 18, pp. 1–23, 2020. [Online]. Available: https://www.mdpi.com/2076-3417/10/18/6253
[63] G. Van Rossum and F. L. Drake, Python 3 Reference Manual. 2009.
[64] T. Kluyver, B. Ragan-Kelley, F. P´ erez, B. Granger, M. Bussonnier, J. Frederic, K. Kelley, J. Hamrick, J. Grout, S. Corlay, P. Ivanov, D. Avila, S. Abdalla, C. Willing, and Jupyter Development Team, "Jupyter notebooks—a publishing format for reproducible computational workflows," in Positioning and Power in Academic Publishing: Players, Agents and Agendas, F. Loizides and B. Schmidt, Eds. IOS Press, Jan. 2016, pp. 87–90. [Online]. Available: https://doi.org/10.3233/978-1-61499-649-1-87

Literatur

[65] G. Research, “Google colaboratory,” Available online at https://colab.google/, 2025, letzter Zugriff am 23.01.2025.
[66] C.-N. Ziegler, S. M. McNee, J. A. Konstan, Konstan, and G. Lausen, “Book-crossing dataset,” Available online at http://www2.informatik.uni-freiburg.de/→cziegler/BX/, 2004, letzter Zugriff am 14.11.2024.
[67] Yelp Inc., “Yelp dataset,” Available online at https://www.yelp.com/dataset, 2015, letzter Zugriff am 18.11.2024.
[68] F. M. Harper and J. A. Konstan, “The MovieLens Datasets: History and Context,” ACM Transactions on Interactive Intelligent Systems, vol. 5, no. 4, pp. 19–19:19, Dec. 2015. [Online]. Available: https://doi.org/10.1145/2827872
[69] D. Anguita, L. Ghelardoni, A. Ghio, L. Oneto, and S. Ridella, “The ‘k’ in k-fold cross validation,” in Proceedings of the 20th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN), Bruges, Belgium, April 2012, pp. 441–446. [Online]. Available: https://www.esann.org/sites/default/files/proceedings/legacy/es2012-62.pdf
[70] P. Kaur and S. Goel, “Shilling attack detection in recommender systems using classification techniques,” International Journal of Engineering Applied Sciences and Technology, vol. 1, no. 7, pp. 147–152, May–June 2016. [Online]. Available: http://www.ijeast.com
[71] F. Zhang, Z.-J. Deng, Z.-M. He, X.-C. Lin, and L.-L. Sun, “Detection of shilling attack in collaborative filtering recommender system by pca and data complexity,” in 2018 International Conference on Machine Learning and Cybernetics (ICMLC), vol. 2, 2018, pp. 673–678.
[72] V. Grozdanić, K. Vladimir, G. Delač, and M.Šilić, “Detection of shilling attacks on collaborative filtering recommender systems by combining multiple random forest models,” in 2023 46th MIPRO ICT and Electronics Convention (MIPRO), 2023, pp. 959–963.

Literatur

[73] M. Ebrahimian and R. Kashef, “E”cient detection of shilling’s attacks in collaborative filtering recommendation systems using deep learning models,” in 2020 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM), 2020, pp. 460–464.
[74] H. Cai and F. Zhang, “Detecting shilling attacks in recommender systems based on analysis of user rating behavior,” Knowledge-Based Systems, vol. 177, pp. 22–43, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0950705119301601
[75] H. Ayaz and Z. Kamıslı Öztürk, “Shilling attack detection with one class support vector machines,” Necmettin Erbakan Üniversitesi Fen ve Mühendislik Bilimleri Dergisi, vol. 5, no. 2,pp. 246–256, 2023.
[76] X. Chen, X. Deng, C. Huang, and H. Shin, “Detection of trust shilling attacks in recommender systems,” IEICE Transactions on Information and Systems, vol. E105.D, pp. 1239–1242, Jun. 2022.
[77] F. Yang, M. Gao, J. Yu, Y. Song, and X. Wang, “Detection of shilling attack based on bayesian model and user embedding,” in 2018 IEEE 30th International Conference on Tools with Artificial Intelligence (ICTAI), 2018, pp. 639–646.
[78] R. A. Zayed, L. F. Ibrahim, H. A. Hefny, H. A. Salman, and A. AlMohimeed, “Using ensemble method to detect attacks in the recommender system,” IEEE Access, vol. 11, pp. 111 315–111 323, 2023.
[79] M. Kaur and S. Rani, “Recommender system: Towards identification of shilling attacks in rating system using machine learning algorithms,” International Journal of Performability Engineering, vol. 19, no. 7, pp. 443–451, 2023. [Online]. Available: https://www.ijpe-online.com/EN/abstract/article 4789.shtml
[80] A. Sihombing and A. Fong, “Fake review detection on yelp dataset using classification techniques in machine learning,” in 2019 International Conference on contemporary Computing and Informatics (IC3I), 2019, pp. 64–68.

Literatur

[81] F. Zhang and Q. Zhou, “Ensemble detection model for profile injection attacks in collaborative recommender systems based on bp neural network,” IET Information Security, vol. 9, no. 1, pp. 24–31, 2013. [Online]. Available: https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/iet-ifs.2013.0145
[82] W. Zhou, J. Wen, Q. Xiong, M. Gao, and J. Zeng, “Svm-tia a shilling attack detection method based on svm and target item analysis in recommender systems,” Neurocomputing, vol. 210, pp. 197–205, 2016, sI:Behavior Analysis In SN. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231216306038
[83] P. K. Singh, P. K. D. Pramanik, M. Sardar, A. Nayyar, M. Masud, and P. Choudhury, “Generating a new shilling attack for recommendation systems,” Computers, Materials & Continua, vol. 71, no. 2, pp. 2827–2846, 2022. [Online]. Available: http://www.techscience.com/cmc/v71n2/45779
[84] Z. Yang, “Defending grey attacks by exploiting wavelet analysis in collaborative filtering recommender systems,” Available online at https://arxiv.org/abs/1506.05247, 2015, letzter Zugriff am 10.02.2025.
[85] W. Zhou, J. Wen, Q. Qu, J. Zeng, and T. Cheng, “Shilling attack detection for recommender systems based on credibility of group users and rating time series,” PLOS ONE, vol. 13, no. 5, pp. 1–17, May 2018. [Online]. Available: https://doi.org/10.1371/journal.pone.0196533