Attacchi Avversari di Avvelenamento dei Dati contro Sistemi di Recommendation Next-Item

Tesi di Laurea Magistrale in Ingegneria Informatica

Giorgio Tocco

Relatore: Prof. Marco Morana

Correlatore: Ing. Andrea Giammanco

UNIVERSITÀ DEGLI STUDI DI PALERMO
DIPARTIMENTO DI INGEGNERIA

CORSO DI LAUREA MAGISTRALE IN INGEGNERIA INFORMATICA

ATTACCHI AVVERSARI DI AVVELENAMENTO DEI DATI CONTRO SISTEMI DI RECOMMENDATION NEXT-ITEM

Tesi di Laurea di
Dott. Giorgio Tocco

Relatore:
Prof. Marco Morana

Controrelatore:

Correlatore:
Ing. Andrea Giammanco

Sommario

I moderni sistemi di raccomandazione sono vulnerabili ad attacchi di avvelenamento dei dati, dove dei campioni dannosi vengono deliberatamente inseriti nei dati di addestramento per manipolare la raccomandazione dei prodotti. Gli attacchi esistenti sono basati su regole euristiche definite manualmente o progettati per sfruttare vulnerabilità di particolari sistemi di raccomandazione. I primi soffrono di scarsi risultati in quanto tendono a sfruttare regole superficiali che non riescono a catturare la complessità dei comportamenti di acquisto degli utenti. Il secondo tipo di attacchi richiede invece un'estensiva conoscenza del particolare sistema di raccomandazione utilizzato dalla piattaforma da colpire, un'informazione che raramente è diffusa al pubblico. Il metodo proposto definisce un modello di attacco di avvelenamento dati contro sistemi di raccomandazione blackbox, basandosi sulle moderne tecniche di adversarial machine learning. Esso consiste nell'utilizzo di un agente di attacco capace di generare campioni di interazioni utente per l'avvelenamento dei dati, con l'obiettivo di manipolare i risultati di raccomandazione per un sottoinsieme di prodotti target. La frequenza di interazioni che è possibile avere con un sistema di raccomandazione è ristretta. Per ovviare a questo problema il metodo propone di far interagire l'agente di attacco con un simulatore locale di sistemi di raccomandazione, sfruttando la trasferibilità dei campioni generati per avvelenare il sistema target. Il costo per il riaddestramento dei sistemi di raccomandazione è spesso molto alto, per questo il metodo proposto sfrutta una funzione di influenza capace di stimare l'effetto dei campioni avversari senza riaddestrare il sistema. I risultati sperimentali mostrano un miglioramento nel

tasso di raccomandazione dei prodotti target, analizzando l'effetto di diverse strategie per la generazione dei campioni avversari.

Indice

Introduzione

L'enorme mole di prodotti offerti dai moderni siti di e-commerce ha reso impossibile per i consumatori vagliare l'intero catalogo nel limitato tempo di utilizzo di queste piattaforme. Basti pensare che Amazon, il sito che vale quasi la metà dell'intero mercato e-commerce, conta più di 600 milioni di prodotti disponibili alla vendita [1]. Al fine di migliorare l'esperienza utente molte aziende di e-commerce si affidano a sistemi di raccomandazione automatici. Essi sono capaci di mostrare agli utenti i prodotti più pertinenti basandosi sugli acquisti e le interazioni passate degli utenti stessi. L'estensivo utilizzo di tali sistemi ha fatto nascere incentivi per lo sviluppo di attacchi volti a manipolare i risultati di raccomandazione. In particolare essi hanno l'obiettivo di spingere la raccomandazione di specifici prodotti o sfavorire la raccomandazione di altri, con il risultato di aumentare le vendite. Recenti studi [2, 3, 4] mostrano che gli attacchi di avvelenamento dei dati riescono con successo a manipolare i risultati di diversi tipi di sistemi di raccomandazione. Essi sono attacchi nati dalla branca di ricerca nota come adversarial machine learning e consistono nell'introduzione di campioni dannosi all'interno dei dati di addestramento usati per imparare il comportamento e le preferenze degli utenti. Questa tipologia di attacchi verrà descritta più nel dettaglio nel Cap.1 e nel particolare caso d'uso dei sistemi di raccomandazione nel Cap.3.

Il metodo proposto consiste nella definizione di un framework per attacchi di avvelenamento dati contro sistemi di raccomandazione next item. I sistemi di raccomandazione next-item hanno l'obiettivo di predirre il prossimo prodotto con cui l'utente ha più probabilità di interagire, basandosi sulle precedenti interazioni dell'utente con i prodotti della piattaforma. Il framework non necessita alcuna informazione sul sistema di raccomandazione da attaccare e prevede un ridotto numero di interazioni con esso.

Il funzionamento è basato sull'interazione di un agente di attacco con un simulatore locale del sistema di raccomandazione target. La costruzione del simulatore locale si basa sulle tecniche di estrazione del modello [5] dell'adversarial machine learning, dove viene creato un modello

surrogato a partire da alcune interazioni con un modello target. L'utilizzo di un'implementazione locale del sistema da attaccare svolge una duplice funzione:

- Ridurre il numero di interazioni con il sistema target. Questa esigenza nasce dalla ridotta frequenza con la quale è possibile interagire con i sistemi in produzione.
- Avere totale accesso all'implementazione del sistema. Questo permette l'utilizzo di tecniche come la funzione di influenza volte a migliorare l'efficienza dell'attacco.

I moderni sistemi di raccomandazione si basano su tecniche di machine learning [6]. In particolare, il metodo proposto va a sfruttare la proprietà di trasferibilità [7] di tali sistemi, permettendo ai campioni dannosi generati a partire dal simulatore locale di avere effetto sul sistema target di cui non si conosce l'implementazione. Tale proprietà fa si che che due modelli definiti sullo stesso dominio (es. medesimo dataset) costruiscano confini decisionali simili, e siano quindi vulnerabili agli stessi campioni antagonisti. Al fine di massimizzare la trasferibilità dei campioni, il simulatore locale è formato da un'aggregazione di diversi sistemi, il cui contributo viene pesato seguendo diverse strategie. Questa scelta nasce dall'intuizione che ciascun modello che forma il simulatore locale può carpire una parte diversa del funzionamento del sistema di raccomandazione target.

La principale sfida per l'agente di attacco è quella di stabilire quanto un campione generato abbia impatto sul punteggio dei prodotti target (i prodotti di cui si vuole aumentare la raccomandazione). La soluzione più immediata sarebbe quella di introdurre i campioni da testare nei dati di addestramento e osservare come cambia la raccomandazione nel modello riaddestrato. Tuttavia riaddestrare un sistema di raccomandazione risulta molto oneroso sia temporalmente che economicamente. Per diminuire il costo dell'attacco il metodo propone l'utilizzo di una funzione di influenza [8]. Essa è una funzione, calcolabile a partire dai gradienti interni del modello, che restituisce l'impatto di un campione di addestramento nel punteggio di un prodotto. Il metodo proposto sfrutta questa funzione di influenza per stabilire quanto, un campione avversario introdotto nell'insieme di addestramento, migliori o peggiori il tasso di raccomandazione di un prodotto target. Il principale vantaggio dell'impiego di una funzione di influenza è quello di evitare il riaddestramento del modello per ciascun campione avversario.

Gli esperimenti sono stati svolti sui dataset MovieLens-100k [9], Amazon Musical Instruments [10], ModCloth [11] e Yelp 2018 [12]. Per testare la trasferibilità dell'attacco sono stati scelti 4 sistemi di raccomandazione next-item (BPRMF [13], CDAE [14], LightGCN [15],

MultiDAE [16]) in modalità leave-one-out. La modalità leave-one-out consiste nel trattare uno dei quattro sistemi come sistema target, e generare i campioni avversari a partire dai restanti tre. Questo viene effettuato con l'obiettivo di simulare l'interazione con il sistema target black-box.

I principali contributi di questo lavoro possono essere riassunti come segue:

- Viene proposto un framework per attacchi di avvelenamento dati contro sistemi di raccomandazione next-item blackbox.
- Proponiamo l'utilizzo di un simulatore locale per ridurre il numero di interazioni con il sistema target e sfruttare l'efficienza della funzione di influenza.
- Il metodo viene ampiamente valutato sperimentalmente, analizzando anche l'effetto di diverse strategie d'attacco.

Il resto di questa tesi è organizzato come segue. Il Capitolo 1 è dedicato ad una panoramica delle tecniche di adversarial machine learning. Il Capitolo 2 e il Capitolo 3 sono dedicati rispettivamente allo stato dell'arte dei sistemi di raccomandazione e agli attacchi di avvelenamento dati contro di essi. Nel Capitolo 4 viene descritto nel dettaglio il sistema proposto. I risultati sono presentati e discussi nel Capitolo 5. Il Capitolo 6 è dedicato alle conclusioni finali.

Capitolo 1

Adversarial Machine Learning

Sin dalla nascita dell'Intelligenza Artificiale si sono costruiti sistemi capaci di generalizzare e prendere decisioni in maniera autonoma. Nella costruzione di tali sistemi assumiamo che l'ambiente e i dati utilizzati siano corretti e incontaminati. Tuttavia è necessario porsi la domanda di cosa potrebbe succedere nei casi in cui non ci si possa fidare dei dati raccolti [17]. La branca di ricerca di Adversarial Machine Learning si occupa proprio di studiare i potenziali attacchi di un soggetto antagonista interessato a compromettere il funzionamento del sistema.

A seconda del compito svolto dall'algoritmo di apprendimento, l'attaccante può impiegare diverse tecniche per manipolare i risultati di predizione. Prendiamo ad esempio il caso di un algoritmo di apprendimento impiegato per svolgere il compito di rilevamento dello spam, dove viene deciso se una mail è considerabile spam o meno a partire da alcune caratteristiche estratte da essa. Il soggetto antagonista potrebbe, in questo scenario applicativo, alterare la mail di spam inviata al fine di portare il sistema a registrare un falso negativo [18]. Un metodo potrebbe essere quello di mascherare le parole comunemente associate a email fraudolente alterando la loro ortografia (es. utilizzare cifre numeriche al posto di alcuni caratteri alfabetici). Altri scenari applicativi simili sono il rilevamento di phishing, intrusioni nella rete [19], malware e altri comportamenti fraudolenti.

Possiamo generalizzare l'esempio descritto come l'interazione tra un classificatore (modello di machine learning impiegato), che raccoglie dati in ingresso e prende decisioni a partire da essi, e un soggetto antagonista, che desidera manipolare i risultati del classificatore adattando costantemente il proprio comportamento in base alla conoscenza di esso [20].

L'obiettivo potrebbe anche essere quello di peggiorare la performance complessiva del

sistema, e nel caso del sistema di rilevamento spam, portare l'utente a disattivarlo permettendo all'antagonista di attaccare senza alcun vincolo [20].

Tuttavia la vulnerabilità dei sistemi di machine learning alla manipolazione antagonistica non si limita a questi casi. Un altro scenario di attacco è l'esfiltrazione di informazioni sensibili contenute nel modello, riguardanti i dati di addestramento o altri segreti industriali. Queste informazioni potrebbero essere divulgate o sfruttate per compiere attacchi più complessi.

È stato inoltre dimostrato che ogni modello addestrato potrebbe avere difficoltà nel compiere predizioni corrette su specifici campioni. Conoscere questi campioni permetterebbe all'antagonista di costruire attacchi specificatamente progettati per sfruttare queste debolezze.

Le diverse tipologie di attacco possono essere categorizzate in quattro macro-tipologie:

- **Attacchi di estrazione del modello**: attacchi con l'obiettivo di replicare il modello di machine learning [21],
- **Attacchi di generazione campioni avversari**: attacchi con l'obiettivo di trovare specifici input che portino il modello a effettuare predizioni errate,
- **Attacchi di avvelenamento dati**: attacchi che alterano l'insieme dei campioni di addestramento del modello,
- **Attacchi di evasione del modello**: attacchi con l'obiettivo di evadere il rilevamento da parte di un modello già addestrato [22].

1.1 Attacchi di estrazione del modello

Questa tipologia di attacchi sta ottenendo molta popolarità in seguito alla diffusione di servizi di cloud computing chiamati MLaaS (Machine Learning as a Service). Essi sono servizi che offrono piattaforme di predizione dove l'utente può inserire i propri dati di addestramento e addestrare modelli di machine learning. Il fornitore del servizio definisce lo specifico modello e algoritmo di apprendimento più adatto per lo scenario applicativo e fornisce un'API per interrogare il modello. La monetizzazione viene effettuata sulle query fatte dall'utente.

L'obiettivo degli attacchi di estrazione è quello di costruire un modello che produca gli stessi risultati del modello target, aggirando i costi di interrogazione effettuando delle query al modello offline. Questo è particolarmente vantaggioso nei casi in cui l'addestramento del

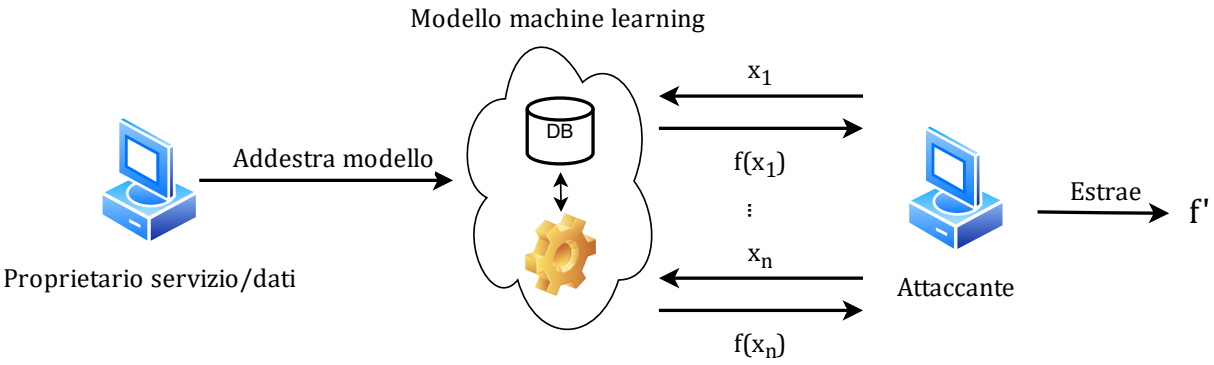

Figura 1.1: Attacco di estrazione del modello

modello originale comporti l'impiego di una grossa quantità di risorse, come la raccolta del dataset e la potenza computazionale.

L'output di un modello di classificazione è solitamente composto da:

- Predizione sulla classe dell'input inserito,
- Valore di confidenza che indica la sicurezza del modello nella predizione effettuata.

Questi valori possono essere utilizzati per dedurre il funzionamento del modello e replicare i suoi confini decisionali. Maggiori sono le informazioni di risposta del modello, maggiore sarà l'efficacia dell'attacco [21]. Tuttavia anche con le sole informazioni delle classi predette è possibile costruire un modello surrogato con performance simili al modello originale [23].

A seconda del modello target e delle informazioni a disposizione dell'attaccante è possibile identificare tre categorie di attacchi di estrazione:

- **Attacchi basati sulla risoluzione di equazioni**: Alcuni modelli di machine learning, come la regressione logistica, impiegano semplici equazioni alla base della predizione. Nella regressione logistica, ad esempio, il modello è formato da una equazione i cui unici parametri non noti all'attaccante sono pesi e bias. Interrogando il modello con un insieme di input costruiti appositamente, l'attaccante può costruire un sistema lineare e risolvere per i pesi e bias. Questa tipologia di attacchi è capace di riprodurre il modello con quasi totale accuratezza [21], tuttavia è difficilmente impiegabile contro modelli più complessi come le reti neurali.

- **Attacchi di ricerca del percorso**: Questo metodo parte dall'assunzione che ogni foglia di un albero decisionale abbia una distribuzione univoca, e che quindi sia possibile tracciare la foglia in cui terminano i dati inseriti. Il metodo consiste nel variare i dati in input una caratteristica alla volta fino a determinare tutti i possibili rami dell'albero decisionale, ricostruendo così il modello.
- **Attacchi di interrogazione adattiva**: Questo metodo consiste nella creazione di un modello surrogato e nel suo addestramento in maniera adattiva. Esso viene addestrato a partire da alcuni dati etichettati e riaddestrato in base ai risultati delle interrogazioni al modello target. Al fine di massimizzare il tasso di apprendimento, verranno richieste le etichette dei dati nei quali il modello locale ha un basso livello di confidenza [23, 24].

1.2 Attacchi di generazione campioni avversari

OMISSIS

1.3 Attacchi di avvelenamento dati

OMISSIS

1.4 Attacchi di evasione del modello

OMISSIS

Capitolo 2

Sistemi di raccomandazione

Questo capitolo presenta una panoramica del campo dei sistemi di raccomandazione e descrive lo stato dell'arte delle metodologie di raccomandazione più diffuse. I moderni approcci possono essere categorizzati in tre tipologie: basati sul contenuto, collaborativi e ibridi.

2.1 Problema della raccomandazione

I sistemi di raccomandazione cominciarono ad essere oggetto di ricerca intorno alla metà degli anni '90 [25, 26, 27], quando i ricercatori cominciarono a dedicarsi a problemi di raccomandazione basati esplicitamente sulle valutazioni. Nella sua formulazione più comune, il problema della raccomandazione consiste nello stimare le valutazione dei prodotti non ancora visualizzati dall'utente. Questa stima è solitamente basata sulle valutazione date dagli utenti su altri prodotti e su diverse altre informazioni descritte successivamente. A partire dalla stima delle valutazioni è possibile raccomandare agli utenti i prodotti con le più alte valutazioni stimate.

Più formalmente, il problema della raccomandazione può essere formulato come segue. Sia C l'insieme di tutti gli utenti e sia S l'insieme di tutti i prodotti che possono essere raccomandati, come libri, film o ristoranti. Lo spazio S di tutti i possibili prodotti può essere molto vasto, dalle centinaia di migliaia fino ai milioni di prodotti in certe applicazioni come libri o brani musicali, e allo stesso modo lo spazio degli utenti può raggiungere i milioni. Sia u la funzione di utilità che misura l'utilità di un prodotto s per un utente c, ovvero, $C \times S \rightarrow R$, dove R è un insieme totalmente ordinato (es. interi non negativi o un intervallo di numeri reali). Per ogni utente $c \in C$ vogliamo quindi trovare un prodotto $s' \in S$ che massimizzi la funzione di utilità, formalmente:

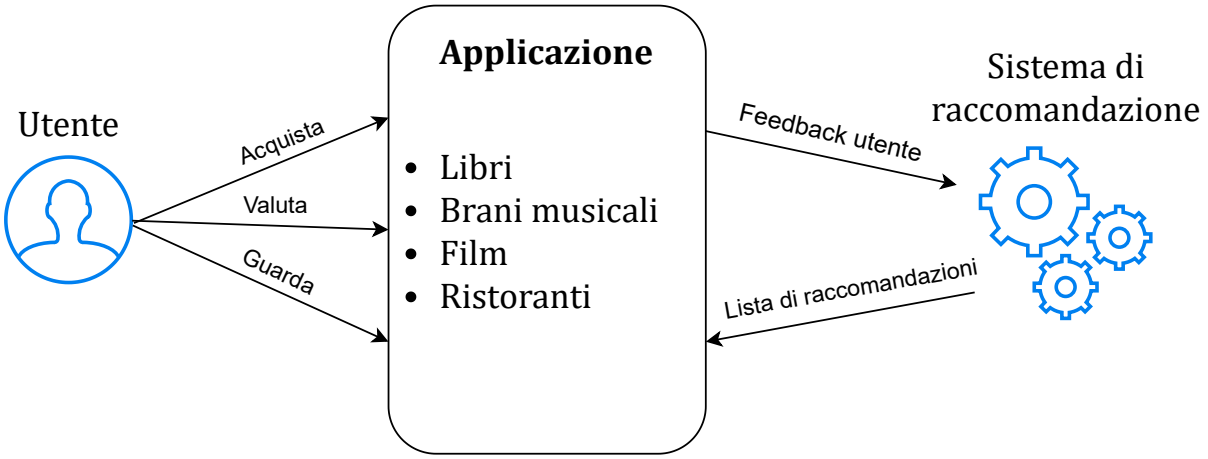

Figura 2.1: Panoramica funzionamento sistema di raccomandazione

$$\forall c \in C,\ s'_c = \arg\max_{s \in S} u(c, s) \tag{2.1}$$

Nei sistemi di raccomandazione l'utilità di un prodotto è solitamente rappresentata da una valutazione, che indica quanto un particolare utente abbia apprezzato un particolare prodotto (es. valutazione di un film da 1 a 5). Tuttavia l'utilità può essere una funzione arbitraria e, in base all'applicazione, può essere specificata dall'utente o calcolata in maniera automatica dal sistema.

Ogni elemento nello spazio degli utenti C può essere definito con un profilo che include diverse caratteristiche dell'utente, come dati anagrafici, reddito, ecc. Nel caso più semplice il profilo contiene un singolo elemento unico, come l'ID Utente. Allo stesso modo ogni prodotto nello spazio S è definito con un insieme di caratteristiche. Ad esempio nel caso di un sistema di raccomandazione per film ogni prodotto è caratterizzato, oltre che dal suo ID, anche da titolo, genere, anno di uscita, ecc.

Il problema fondamentale dei sistemi di raccomandazione risiede nel fatto che l'utilità u non è definita per l'intero spazio $C \times S$, ma solamente in un suo sottoinsieme. Questo comporta che la funzione u deve essere estrapolata per l'intero spazio $C \times S$. Nei sistemi di raccomandazione, l'utilità è spesso definita da valutazioni ed è inizilamente presente solo nei prodotti precedentemente valutati dagli utenti. Ad esempio, in un sistema di raccomandazione per film, gli utenti valutano un sottoinsieme dei film che hanno già visto. La Tabella 2.1 fornisce un esempio di matrice di interazione per un tale sistema, dove le valutazioni vanno da 1 a 5. Il

simbolo ∅ rappresenta la mancanza di valutazione per quel film.

Tabella 2.1: Esempio matrice di interazione per un sistema di raccomandazione di film

	Film 1	**Film 2**	**Film 3**	**Film 4**
Utente 1	4	2	1	∅
Utente 2	3	∅	1	3
Utente 3	1	4	2	∅
Utente 4	4	∅	1	∅

Il sistema di raccomandazione dovrà quindi stimare le valutazioni per le coppie utente-film senza valutazione e provvedere raccomandazioni appropriate a partire da esse.

L'estrapolazione delle valutazioni sconosciute a partire da quelle conosciute è solitamente effettuata applicando due diverse strategie:

- Specificare una euristica che definisca la funzione di utilità e valutare sperimentalmente la performance
- Stimare la funzione di utilità che ottimizzi dei criteri di valutazione, come l'errore quadratico medio,

Una volta stimate le valutazioni sconosciute, la raccomandazione vera e propria è effettuata selezionando il prodotto con la valutazione più alta tra quelle stimate, seguendo l'Eq. 2.1. È possibile inoltre raccomandare i migliori K prodotti agli utenti o viceversa selezionare un insieme di utenti più interessati per un prodotto.

Le nuove valutazioni per i prodotti non ancora valutati possono essere stimate in diversi modi utilizzando tecniche di machine learning, teoria dell'approssimazione e varie euristiche. I sistemi di raccomandazione sono classificati in base al loro approccio per la stima delle valutazioni e di seguito verrà presentata una tale classificazione come proposta nella letteratura. I sistemi di raccomandazione vengono classificati in:

- **Approcci basati sul contenuto**: all'utente sono raccomandati prodotti simili a quelli che l'utente stesso ha preferito nel passato
- **Approcci collaborativi**: all'utente sono raccomandati prodotti che utenti con gusti e preferenze simili hanno preferito nel passato

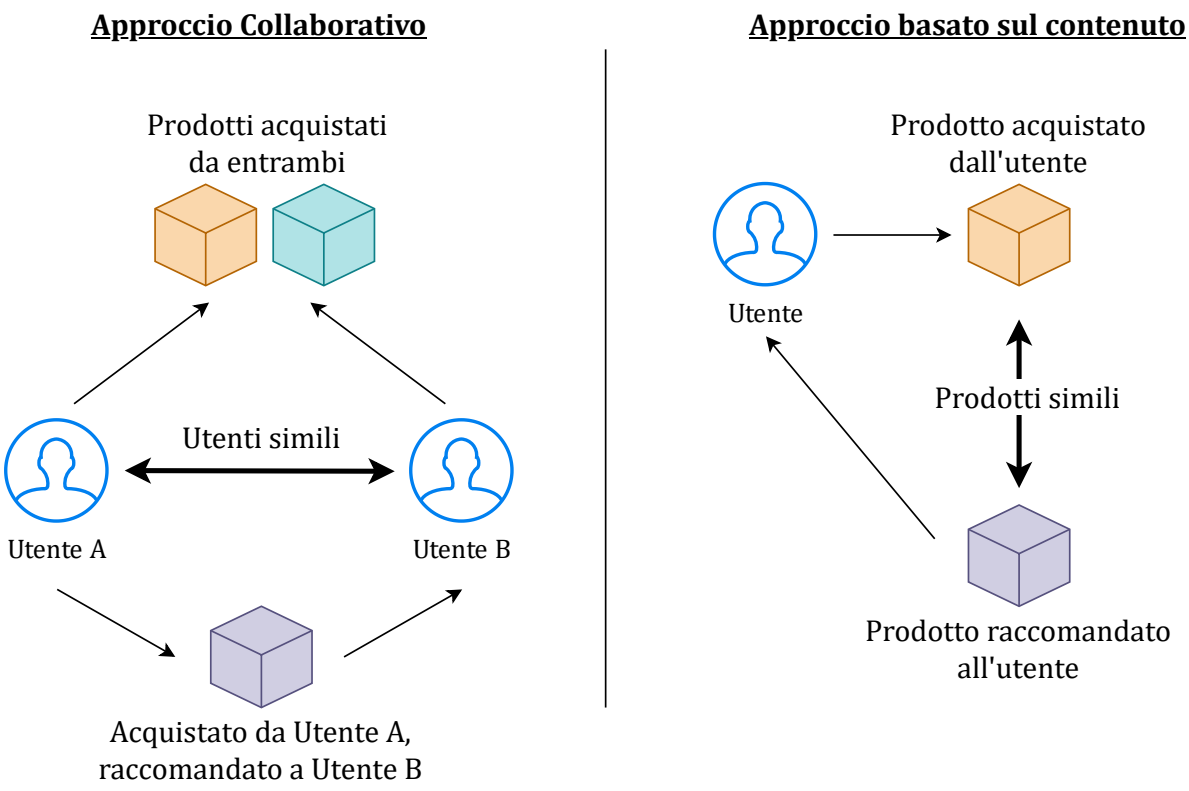

Figura 2.2: Panoramica funzionamento dei sistemi collaborativi e basati sul contenuto

- **Approcci ibridi**: questi approcci combinano i metodi collaborativi e basati sul contenuto.

Oltre a sistemi di raccomandazione che stimano il valore assoluto delle valutazioni che utenti darebbero a prodotti non visualizzati (come discusso sopra), esistono metodi come il filtraggio basato sulle preferenze che hanno l'obiettivo di predire le preferenze relative degli utenti [28, 29, 30]. Ad esempio, in un sistema di raccomandazione per film, le tecniche di filtraggio basate sulle preferenze hanno l'obiettivo di predire l'ordine relativo dei film, piuttosto che la loro singola valutazione. In questo capitolo tuttavia ci concentreremo sui sistemi di raccomandazione basati sulle valutazioni, in quanto rappresentano l'approccio più diffuso nei moderni sistemi.

2.2 Approcci basati sul contenuto

OMISSIS

2.2.1 Metodi basati su euristiche

OMISSIS

2.2.2 Metodi basati su modelli

OMISSIS

2.2.3 Limiti dell'approccio

I sistemi di raccomandazione basati sul contenuto presentano diverse limitazioni descritte di seguito.

Limitata analisi del contenuto

OMISSIS

Sovraspecializzazione

OMISSIS

Problema della partenza a freddo

OMISSIS

2.3 Approcci collaborativi

OMISSIS

2.3.1 Metodi basati su memoria

OMISSIS

Approcci basati sulla correlazione

OMISSIS

Approcci basati sulla similarità del coseno

OMISSIS

2.3.2 Metodi basati su modelli

OMISSIS

2.3.3 Limiti dell'approccio

OMISSIS

Problema del nuovo utente

Come i sistemi basati sul contenuto, anche quelli collaborativi necessitano di conoscere le preferenze dell'utente (ricavate dalla valutazione di prodotti) al fine di fornirne raccomandazioni rilevanti. Esistono diverse tecniche capaci di risolvere questo problema, molte delle quali consistono nella combinazione di approcci basati sul contenuto e collaborativi. Questa combinazione prende il nome di sistema ibrido e verrà descritto nel capitolo successivo. Un'altra soluzione è quella di determinare il sottoinsieme di prodotti, da far valutare ai nuovi utenti, con il più alto contenuto informativo.

Problema del nuovo prodotto

I sistemi di raccomandazione collaborativi si basano unicamente sulle preferenze degli utenti per produrre raccomandazioni. Di conseguenza, quando viene inserito un nuovo prodotto, il sistema non sarà capace di raccomandarlo finchè un numero sufficientemente grande di utenti lo avranno valutato.

Sparsità

In tutti i sistemi di raccomandazione, il numero di valutazioni raccolte è molto inferiore rispetto al numero di valutazioni che è necessario stimare. È necessario quindi che i sistemi impiegati siano capaci di produrre risultati rilevanti a partire da un basso numero di campioni. Un modo per ovviare a questo problema è quello di utilizzare le informazioni del profilo utente nel calcolo della similarità tra utenti. Questo consisterebbe nel considerare due utenti simili,

non solo se hanno valutato gli stessi prodotti similmente, ma anche se appartengono allo stesso segmento demografico o altre caratteristiche. Ad esempio, [31] utilizza sesso, età, codice postale, grado di istruzione e stato occupazionale dell'utente in un sistema di raccomandazione per ristoranti. Questa estensione delle tradizionali tecniche collaborative viene chiamata "filtraggio demografico" [31].

2.4 Approcci ibridi

OMISSIS

Combinazione di sistemi separati

OMISSIS

Aggiungere caratteristiche basate sul contenuto a sistemi collaborativi

OMISSIS

Aggiungere caratteristiche collaborative a sistemi basati sul contenuto

OMISSIS

Singolo sistema unificante

OMISSIS

Capitolo 3

Attacchi a sistemi di raccomandazione

Questo capitolo presenta una panoramica delle tipologie di attacchi verso sistemi di raccomandazione. Verrano inoltre descritti alcuni moderni attacchi di avvelenamento dati ampiamente studiati nella letteratura.

3.1 Tipologie di attacchi

Gli attacchi ai sistemi di raccomandazione possono essere categorizzati nel seguente modo:

- **Attacchi di manipolazione**
  - Attacchi di avvelenamento dei dati
  - Attacchi di inquinamento dei profili
- **Attacchi alla privacy**
  - Attacchi di inferenza dei prodotti
  - Attacchi di inferenza degli attributi

3.1.1 Attacchi di manipolazione

Gli attacchi di manipolazione hanno l'obiettivo di manipolare il sistema di raccomandazione aumentando o riducendo il numero di volte in cui un prodotto viene raccomandato agli utenti. Questi tipi di attacchi possono essere suddivisi in: attacchi di avvelenamento dei dati [32, 33,

34] e attacchi di inquinamento dei profili [35]. Gli attacchi di avvelenamento dei dati mirano a introdurre utenti e valutazioni fittizie [36] nel sistema in modo da compromettere la fase di addestramento. Gli attacchi di inquinamento dei profili mirano invece a inquinare le valutazioni di utenti reali in modo da manipolare le loro specifiche raccomandazioni. In analogia con l'adversarial machine learning, gli attacchi di avvelenamento manipolano il sistema in fase di addestramento, mentre gli attacchi di inquinamento manipolano il sistema in fase di testing.

Attacchi di avvelenamento dei dati

I primi attacchi di avvelenamento ai sistemi di raccomandazione risalgono a più di un decennio fa [32, 33, 34]. Tuttavia questi attacchi si basano su euristiche manuali e non sono ottimizzati per specifici sistemi di raccomandazione. Esempi di questi attacchi sono gli attacchi casuali [32], dove, dato il numero di utenti fittizi che è possibile introdurre nel sistema, l'attaccante sceglie un insieme casuale di prodotti per ogni utente fittizio e genera una valutazione per ogni prodotto selezionato a partire da una distribuzione normale, la cui media e varianza è calcolata a partire dalle reali valutazioni degli utenti. Un altro esempio sono gli attacchi di media [32], dove l'attaccante genera una valutazione per un prodotto selezionato a partire da una distribuzione normale, la cui media e varianza sono calcolate a partire dalla media delle valutazioni per quello specifico prodotto.

Metodi più recenti [37, 38] generano valutazioni o comportamenti fittizi ottimizzati per una particolare tipologia di sistemi di raccomandazione. Nello specifico [37] propone un attacco di avvelenamento verso sistemi basati sulla fattorizzazione matriciale, mentre [38] propone un attacco verso sistemi basati su regole associative.

I capitoli 3.2 e 3.3 si occupano di descrivere ancora più nel dettaglio gli attacchi di avvelenamento dei dati.

Attacchi di inquinamento dei profili

Gli attacchi di inquinamento dei profili mirano a modificare il comportamento dell'utente al fine di manipolare le raccomandazioni da lui ottenute. Ad esempio [35] utilizza la tecnica di cross-site request forgery (CSRF) [39] per modificare la cronologia di ricerca dell'utente al fine di inquinare il suo profilo nel sistema di raccomandazione. La letteratura mostra che piattaforme popolari come YouTube, Amazon e Google sono vulnerabili ad attacchi di questo tipo. L'utilizzo della CSRF rende però difficile l'esecuzione a larga scala di questo attacco.

3.1.2 Attacchi alla privacy

Gli attacchi alla privacy [40] hanno l'obiettivo di estrarre informazioni sensibili degli utenti a partire dalle loro interazioni con un sistema di raccomandazione. Essi possono essere suddivisi in: attacchi di inferenza dei prodotti e attacchi di inferenza degli attributi.

Attacchi di inferenza dei prodotti

Il lavoro di Calandrino et al. [41] propone un attacco alla privacy capace di inferire i prodotti che un utente ha valutato in passato, come ad esempio prodotti acquistati, brani musicali ascoltati o libri letti. L'attacco proposto si basa sul fatto che un sistema collaborativo produce raccomandazioni a partire dalle interazioni passati dell'utente. Questo comporta che le raccomandazioni prodotte contengono informazioni riguardante il comportamento passato dell'utente. È possibile quindi, tenendo traccia delle raccomandazioni prodotte per un utente, risalire ai prodotti precedentemente valutati dall'utente.

Attacchi di inferenza degli attributi

Le valutazioni prodotte da un utente sono statisticamente correlate agli attributi che lo caratterizzano, come orientamento politico, interessi, posizione geografica ecc. Un attaccante potrebbe quindi dedurre gli attributi privati di un utente a partire dalle sue valutazioni utilizzando tecniche di machine learning capaci di catturare le correlazioni statistiche tra comportamento e attributi. Questa tipologia di attacchi prende il nome in letteratura di *attacchi di inferenza degli attributi* [42]. In particolare, la tecnica prevede l'utilizzo di un classificatore basato sul machine learning addestrato su un insieme di utenti le quali valutazioni e caratteristiche sono conosciuti all'attaccante. Il modello viene quindi impiegato per dedurre gli attributi di utenti le cui caratteristiche non sono note. Uno dei modi [43] per rendere il sistema di raccomandazione più robusto a questo tipo di attacchi è l'introduzione di rumore, specificatamente progettato per compromettere le predizioni del classificatore, nelle valutazioni degli utenti.

3.2 Scenari di attacco

La tipologia di attacco eseguita varia a seconda della conoscenza sul sistema target, dell'intento del particolare attacco e dalle risorse disponibili all'attaccante.

L'obiettivo per l'attaccante è quello di trovare l'attacco più efficace in termini di impatto sul sistema attaccato e di impiego di risorse. Un modo per valutare l'impegno necessario per un attacco è misurare la quantità di conoscenza necessaria per lanciarlo. La conoscenza di un sistema di raccomandazione riguarda dettagli implementativi come lo specifico algoritmo alla base o informazioni statistiche come la distribuzione delle valutazioni. Questo tipo di informazioni sono spesso più difficili da ottenere rispetto a informazioni riguardanti i prodotti come la classifica dei prodotti più venduti. È possibile dunque distinguere due scenari di attacco in base alle informazioni conosciute:

- **Alta conoscenza**: quando l'attacco richiede la conoscenza di specifiche informazioni come dati statistici aggregati. Alcuni attacchi, ad esempio, utilizzano la media o la deviazione standard delle valutazioni di un prodotto per condurre l'attacco.
- **Bassa conoscenza**: quando l'attacco richiede solamente informazioni indipendenti dal sistema e publicamente accessibili.

Un'altra distinzione possibile riguarda l'intento dell'attaccante. Spesso gli incentivi economici riguardano la promozione di uno specifico prodotto o la retrocessione di un prodotto concorrente, entrambi con il fine di aumentare le vendite. In alternativa un attaccante potrebbe essere interessato a compromettere il funzionamento dell'intero sistema, senza aumentare o diminuire il punteggio di alcuno specifico prodotto.

Infine è necessario analizzare la scala dell'attacco. Essa dipende dalle risorse disponibili all'attaccante e viene spesso misurata nel numero di profili utente fittizi che è possibile controllare. Le moderne piattaforme di e-commerce, infatti, impongono l'interazione con un umano al momento della registrazione tramite tecnologie come il CAPTCHA. Questo rende impossibile l'automatizzazione del processo di registrazione, associando quindi un costo ad ogni profilo utente. Un'altra misura potrebbe essere il numero di valutazioni effettuate da ciascun utente, tuttavia questa risulta molto più accessibile rispetto alla creazione di un profilo utente. I profili fittizi introdotti dall'attaccante corrono inoltre il rischio di essere rilevati da eventuali sistemi di anomaly detection impiegati nella piattaforma. Spesso infatti i profili controllati dall'attaccante presentano pattern di comportamento molto diversi dai normali utenti, che solitamente si limitano a valutare un basso numero di prodotti.

3.3 Modelli di attacco

OMISSIS

3.3.1 Attacchi di retrocessione

OMISSIS

Capitolo 4

Metodo Proposto

In questo capitolo viene presentato il metodo proposto per attacchi di avvelenamento dati contro sistemi di raccomandazione. Le sezioni che precedono la descrizione del metodo agiscono come linee guida per comprendere i criteri alla base della sua formulazione, con l'obiettivo di collocare il metodo proposto in relazione agli argomenti affrontati nel Capitolo 2 e alle limitazioni che riesce a superare rispetto ai sistemi presenti in letteratura.

4.1 Limiti degli approcci esistenti

Gli approcci esistenti possono essere categorizzati in due tipologie:

- **Attacchi basati su regole euristiche**: essi si basano su regole definite manualmente che vanno a sfruttare intuizoni come "prodotti acquistati insieme sono altamente correlati all'interno del sistema di raccomandazione".
- **Attacchi a specifici sistemi**: essi sono progettati per sfruttare l'implementazione dello specifico sistema di raccomandazione da colpire.

La prima tipologia di attacchi soffre di scarsa performance, dovuta al fatto che tali regole euristiche non riescono a catturare i diversi pattern di comportamento degli utenti. Un approccio esplorato da O'Mahony et al. [44], ad esempio, propone di utilizzare degli utenti controllati per inserire false co-occorenze tra i prodotti target e i prodotti più popolari. Questo approccio, seppur permetta di ottenere dei risultati, non tiene conto di comportamenti più complessi degli utenti, come preferenze personali o acquisti passati.

La seconda tipologia di attacchi soffre della necessità di avere estensiva conoscenza del sistema di raccomandazione da colpire. Spesso infatti, seppur i risultati di raccomandazione siano disponibili al pubblico, i dettagli sul modello utilizzato non sono noti. L'attacco proposto da Li et al. [44], ad esempio, è progettato per colpire sistemi di raccomandazioni basati sulla matrix factorization (Cap. 4.1).

4.1.1 Funzione di influenza negli approcci esistenti

Esistono alcuni metodi in letteratura [45] [46] [47] che utilizzano la funzione di influenza per la costruzione di attacchi di avvelenamento dati. Essi utilizzano la funzione di influenza per trovare il sottoinsieme di utenti più influenti.

In particolare il lavoro di Fang et al. (2020) [45] si basa sull'osservazione fatta in letteratura [48] [49] che diversi campioni di addestramento hanno diversi contributi sulla qualità della soluzione di un problema di ottimizzazione, e che la performance dell'addestramento di un modello può essere migliorata rimuovendo i campioni con basso contributo. A partire da questa osservazione essi hanno modificato il problema di ottimizzazione in modo da coinvolgere solamente gli utenti più influenti, migliorando l'efficienza dell'attacco.

4.2 Scenario applicativo

OMISSIS

4.3 Simulatore locale

OMISSIS

4.4 Agente di attacco

OMISSIS

Capitolo 5

Valutazione Sperimentale

In questo capitolo viene esposta la metodologia utilizzata per l'esecuzione degli esperimenti e il loro risultato. Vengono descritti i dataset utilizzati esplorando la distribuzione dei dati ed altre statistiche. Vengono descritte le metriche utilizzate per l'addestramento dei modelli e la verifica dei risultati. Vengono descritti i diversi esperimenti eseguiti per testare l'effetto del clustering e delle strategie di aggregazione nel tasso di raccomandazione dei prodotti target.

5.1 Descrizione dei dataset

OMISSIS

5.2 Descrizione metriche

OMISSIS

5.3 Setup sperimentale

OMISSIS

5.4 Descrizione esperimenti

OMISSIS

Capitolo 6

Conclusioni

In questo lavoro si sono studiate le moderne tecniche di adversarial machine learning, con particolare riguardo agli attacchi di avvelenamento del dataset. In seguito a una panoramica sullo stato dell'arte dei sistemi di raccomandazione, sono state descritte le diverse vulnerabilità di cui soffrono tali sistemi. Al fine di costruire una base teorica per i moderni attacchi verso i sistemi di raccomandazione, è stato descritto il concetto di modello di attacco e le sue molteplici tipologie. Infine sono state descritte le moderne tecniche di attacco, con particolare riguardo agli attacchi volti a promuovere un sottoinsieme di prodotti target.

A tal fine è stato proposto un nuovo framework per la realizzazione di attacchi di avvelenamento dati, utilizzando diverse tecniche di adversarial machine learning per migliorare l'efficienza degli attacchi. La prima tecnica riguarda l'utilizzo di un modello surrogato locale, capace di simulare il sistema di raccomandazione target, riducendo il numero di interazioni con esso. Essa risulta essenziale per ridurre il carico computazionale dell'attacco, e di conseguenza il costo e la durata. La seconda è l'utilizzo innovativo di una funzione di influenza, capace di stimare l'effetto dei campioni avversari senza riaddestrare il modello, riducendo ancora l'utilizzo di risorse.

Il sistema proposto è formato da due componenti fondamentali: il simulatore locale del sistema di raccomandazione target blackbox e l'agente di attacco che si occupa di generare i campioni di avvelenamento. Per convalidare l'approccio il sistema è stato valutato ampiamente con moltepici esperimenti su quattro dataset reali.

I risultati mostrano che il sistema genera campioni di avvelenamento capaci di migliorare il tasso di raccomandazione dei prodotti target, evidenziando l'efficacia del clustering nel campionamento delle interazioni e della strategia di aggregazione basata sulla distanza di

Kendall. Il metodo proposto è stato confrontato con due strategie di attacco di uso comune, risultando più efficace in maniera significativa in due dei tre dataset testati.

La principale sfida affrontata durante lo sviluppo del metodo proposto riguarda l'implementazione efficiente della funzione di influenza. Essa infatti richiede il calcolo della matrice Hessiana del modello, il quale corrisponde al calcolo del gradiente di secondo grado della funzione di loss per ciascun campione del dataset. Questo calcolo risulta eccessivamente oneroso, dovendo essere ripetuto per ogni campione avversario che si desidera testare. Il metodo propone infatti l'utilizzo di un algoritmo di approssimazione per il calcolo del valore che comprende la matrice Hessiana. Tuttavia l'algoritmo impiegato ha una complessità computazionale proporzionale al numero di campioni nell'insieme di addestramento, rendendo oneroso l'utilizzo in scenari d'uso con dataset molto vasti.

Alcuni sviluppi futuri potrebbero riguardare l'utilizzo di modelli sequenziali nel simulatore locale, il che permetterebbe al framework la capacità di eseguire attacchi su sistemi di raccomandazione target di tipo sequenziale. Questo sviluppo comporterebbe un grosso aumento nello spazio delle possibili interazioni, anche in considerazione del fatto che l'ordine in cui i prodotti vengono acquistati ha una rilevanza per l'utente. Un modo per ovviare a questo problema potrebbe essere quello di addestrare un agente tramite reinforcement learning per imparare una policy che generi sequenze di interazioni.

Elenco delle figure

Elenco delle tabelle

Bibliografia

[1] *AMZScout Amazon Statistics*. https://amzscout.net/blog/amazon-statistics/. Accessed: 2023-08-31.

[2] M. Fang, G. Yang, N. Z. Gong e J. Liu. «Poisoning attacks to graph-based recommender systems». In: *Proceedings of the 34th annual computer security applications conference*. 2018, pp. 381–392.

[3] H. Huang, J. Mu, N. Z. Gong, Q. Li, B. Liu e M. Xu. «Data poisoning attacks to deep learning based recommender systems». In: *arXiv preprint arXiv:2101.02644* (2021).

[4] L. Chen, Y. Xu, F. Xie, M. Huang e Z. Zheng. «Data poisoning attacks on neighborhood-based recommender systems». In: *Transactions on Emerging Telecommunications Technologies* 32.6 (2021), e3872.

[5] H. Hu e J. Pang. «Stealing machine learning models: Attacks and countermeasures for generative adversarial networks». In: *Annual Computer Security Applications Conference*. 2021, pp. 1–16.

[6] I. Portugal, P. Alencar e D. Cowan. «The use of machine learning algorithms in recommender systems: A systematic review». In: *Expert Systems with Applications* 97 (2018), pp. 205–227.

[7] A. Demontis, M. Melis, M. Pintor, M. Jagielski, B. Biggio, A. Oprea, C. Nita-Rotaru e F. Roli. «Why do adversarial attacks transfer? explaining transferability of evasion and poisoning attacks». In: *28th USENIX security symposium (USENIX security 19)*. 2019, pp. 321–338.

[8] P. W. Koh e P. Liang. «Understanding black-box predictions via influence functions». In: *International conference on machine learning*. PMLR. 2017, pp. 1885–1894.

[9] *MovieLens Dataset*. https://grouplens.org/datasets/movielens/. Accessed: 2023-09-04.

[10] R. He e J. McAuley. «Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering». In: *proceedings of the 25th international conference on world wide web*. 2016, pp. 507–517.

[11] R. Misra, M. Wan e J. McAuley. «Decomposing fit semantics for product size recommendation in metric spaces». In: *Proceedings of the 12th ACM Conference on Recommender Systems*. 2018, pp. 422–426.

[12] *Yelp Dataset*. https://www.yelp.com/dataset. Accessed: 2023-09-10.

[13] S. Rendle, C. Freudenthaler, Z. Gantner e L. Schmidt-Thieme. «BPR: Bayesian personalized ranking from implicit feedback». In: *arXiv preprint arXiv:1205.2618* (2012).

[14] Y. Wu, C. DuBois, A. X. Zheng e M. Ester. «Collaborative denoising auto-encoders for top-n recommender systems». In: *Proceedings of the ninth ACM international conference on web search and data mining*. 2016, pp. 153–162.

[15] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang e M. Wang. «Lightgcn: Simplifying and powering graph convolution network for recommendation». In: *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*. 2020, pp. 639–648.

[16] D. Liang, R. G. Krishnan, M. D. Hoffman e T. Jebara. «Variational autoencoders for collaborative filtering». In: *Proceedings of the 2018 world wide web conference*. 2018, pp. 689–698.

[17] P. Laskov e R. Lippmann. *Machine learning in adversarial environments*. 2010.

[18] N. Dalvi, P. Domingos, Mausam, S. Sanghai e D. Verma. «Adversarial classification». In: *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. 2004, pp. 99–108.

[19] V. Agate, F. M. D'Anna, A. De Paola, P. Ferraro, G. L. Re e M. Morana. «A behavior-based intrusion detection system using ensemble learning techniques». In: *ITASEC* (2022).

[20] J. Tygar. «Adversarial machine learning». In: *IEEE Internet Computing* 15.5 (2011), pp. 4–6.

[21] F. Tramèr, F. Zhang, A. Juels, M. K. Reiter e T. Ristenpart. «Stealing machine learning models via prediction {APIs}». In: *25th USENIX security symposium (USENIX Security 16)*. 2016, pp. 601–618.

[22] K. N. Khasawneh, N. Abu-Ghazaleh, D. Ponomarev e L. Yu. «RHMD: Evasion-resilient hardware malware detectors». In: *Proceedings of the 50th Annual IEEE/ACM international symposium on microarchitecture*. 2017, pp. 315–327.

[23] D. Lowd e C. Meek. «Adversarial learning». In: *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*. 2005, pp. 641–647.

[24] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik e A. Swami. «Practical black-box attacks against machine learning». In: *Proceedings of the 2017 ACM on Asia conference on computer and communications security*. 2017, pp. 506–519.

[25] W. Hill, L. Stead, M. Rosenstein e G. Furnas. «Recommending and evaluating choices in a virtual community of use». In: *Proceedings of the SIGCHI conference on Human factors in computing systems*. 1995, pp. 194–201.

[26] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom e J. Riedl. «Grouplens: An open architecture for collaborative filtering of netnews». In: *Proceedings of the 1994 ACM conference on Computer supported cooperative work*. 1994, pp. 175–186.

[27] U. Shardanand e P. Maes. «Social information filtering: Algorithms for automating "word of mouth"». In: *Proceedings of the SIGCHI conference on Human factors in computing systems*. 1995, pp. 210–217.

[28] W. W. Cohen, R. E. Schapire e Y. Singer. «Learning to order things». In: *Advances in neural information processing systems* 10 (1997).

[29] Y. Freund, R. Iyer, R. E. Schapire e Y. Singer. «An efficient boosting algorithm for combining preferences». In: *Journal of machine learning research* 4.Nov (2003), pp. 933–969.

[30] R. Jin, L. Si e C. Zhai. «Preference-based graphic models for collaborative filtering». In: *arXiv preprint arXiv:1212.2478* (2012).

[31] M. J. Pazzani. «A framework for collaborative, content-based and demographic filtering». In: *Artificial intelligence review* 13 (1999), pp. 393–408.

[32] S. K. Lam e J. Riedl. «Shilling recommender systems for fun and profit». In: *Proceedings of the 13th international conference on World Wide Web*. 2004, pp. 393–402.

[33] B. Mobasher, R. Burke, R. Bhaumik e C. Williams. «Toward trustworthy recommender systems: An analysis of attack models and algorithm robustness». In: *ACM Transactions on Internet Technology (TOIT)* 7.4 (2007), 23–es.

[34] M. O'Mahony, N. Hurley, N. Kushmerick e G. Silvestre. «Collaborative recommendation: A robustness analysis». In: *ACM Transactions on Internet Technology (TOIT)* 4.4 (2004), pp. 344–377.

[35] X. Xing, W. Meng, D. Doozan, A. C. Snoeren, N. Feamster e W. Lee. «Take this personally: Pollution attacks on personalized services». In: *22nd USENIX Security Symposium (USENIX Security 13)*. 2013, pp. 671–686.

[36] V. Agate, A. De Paola, G. Lo Re e M. Morana. «DRESS: A Distributed RMS Evaluation Simulation Software». In: *International Journal of Intelligent Information Technologies (IJIIT)* 16.3 (2020), pp. 1–18.

[37] B. Li, Y. Wang, A. Singh e Y. Vorobeychik. «Data poisoning attacks on factorization-based collaborative filtering». In: *Advances in neural information processing systems* 29 (2016).

[38] G. Yang, N. Z. Gong e Y. Cai. «Fake Co-visitation Injection Attacks to Recommender Systems.» In: *NDSS*. 2017.

[39] W. Zeller e E. W. Felten. «Cross-site request forgeries: Exploitation and prevention». In: *The New York Times* (2008), pp. 1–13.

[40] V. Agate, P. Ferraro, G. L. Re e S. K. Das. «BLIND: A privacy preserving truth discovery system for mobile crowdsensing». In: *Journal of Network and Computer Applications* (2023), p. 103811.

[41] J. A. Calandrino, A. Kilzer, A. Narayanan, E. W. Felten e V. Shmatikov. «" You might also like:" Privacy risks of collaborative filtering». In: *2011 IEEE symposium on security and privacy*. IEEE. 2011, pp. 231–246.

[42] N. Z. Gong e B. Liu. «You are who you know and how you behave: Attribute inference attacks via users' social friends and behaviors». In: *25th USENIX Security Symposium (USENIX Security 16)*. 2016, pp. 979–995.

[43] J. Jia e G. AttriGuard. «A practical defense against attribute inference attacks via adversarial machine learning». In: *Proceedings of the 27th USENIX Security Symposium (USENIX Security 18)*. 2018, pp. 513–529.

[44] M. P. O'Mahony, N. J. Hurley e G. C. Silvestre. «Recommender systems: Attack types and strategies». In: *AAAI*. 2005, pp. 334–339.

[45] M. Fang, N. Z. Gong e J. Liu. «Influence function based data poisoning attacks to top-n recommender systems». In: *Proceedings of The Web Conference 2020*. 2020, pp. 3019–3025.

[46] C. Wu, D. Lian, Y. Ge, Z. Zhu e E. Chen. «Influence-Driven Data Poisoning for Robust Recommender Systems». In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2023).

[47] C. Wu, D. Lian, Y. Ge, Z. Zhu e E. Chen. «Triple adversarial learning for influence based poisoning attack in recommender systems». In: *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 2021, pp. 1830–1840.

[48] A. Lapedriza, H. Pirsiavash, Z. Bylinskii e A. Torralba. «Are all training examples equally valuable?» In: *arXiv preprint arXiv:1311.6510* (2013).

[49] T. Wang, J. Huan e B. Li. «Data dropout: Optimizing training data for convolutional neural networks». In: *2018 IEEE 30th international conference on tools with artificial intelligence (ICTAI)*. IEEE. 2018, pp. 39–46.

[50] H. Zhang, Y. Li, B. Ding e J. Gao. «Practical data poisoning attack against next-item recommendation». In: *Proceedings of The Web Conference 2020*. 2020, pp. 2458–2464.

[51] M. G. Kendall. «A new measure of rank correlation». In: *Biometrika* 30.1/2 (1938), pp. 81–93.

[52] R. D. Cook e S. Weisberg. «Residuals and influence in regression». In: (1982).

[53] G. K. Nilsen, A. Z. Munthe-Kaas, H. J. Skaug e M. Brun. «Efficient computation of hessian matrices in tensorflow». In: *arXiv preprint arXiv:1905.05559* (2019).

[54] N. Agarwal, B. Bullins e E. Hazan. «Second-order stochastic optimization for machine learning in linear time». In: *The Journal of Machine Learning Research* 18.1 (2017), pp. 4148–4187.

[55] D. Lee e H. S. Seung. «Algorithms for non-negative matrix factorization». In: *Advances in neural information processing systems* 13 (2000).

[56] J. MacQueen. «Classification and analysis of multivariate observations». In: *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*. 1967, pp. 281–297.

[57] A. Y. Ng. «Feature selection, L 1 vs. L 2 regularization, and rotational invariance». In: *Proceedings of the twenty-first international conference on Machine learning*. 2004, p. 78.

[58] N. Idrissi e A. Zellou. «A systematic literature review of sparsity issues in recommender systems». In: *Social Network Analysis and Mining* 10 (2020), pp. 1–23.

[59] E. Voorhees. «Proceedings of the 8th text retrieval conference». In: *TREC-8 Question Answering Track Report* (1999), pp. 77–82.

[60] W. X. Zhao, S. Mu, Y. Hou, Z. Lin, Y. Chen, X. Pan, K. Li, Y. Lu, H. Wang, C. Tian, Y. Min, Z. Feng, X. Fan, X. Chen, P. Wang, W. Ji, Y. Li, X. Wang e J. Wen. «RecBole: Towards a Unified, Comprehensive and Efficient Framework for Recommendation Algorithms». In: *CIKM*. ACM, 2021, pp. 4653–4664.

[61] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot e E. Duchesnay. «Scikit-learn: Machine Learning in Python». In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[62] S. Wold, K. Esbensen e P. Geladi. «Principal component analysis». In: *Chemometrics and intelligent laboratory systems* 2.1-3 (1987), pp. 37–52.

[63] H. Zhang, Y. Li, B. Ding e J. Gao. «LOKI: a practical data poisoning attack framework against next item recommendations». In: *IEEE Transactions on Knowledge and Data Engineering* 35.5 (2022), pp. 5047–5059.

[64] C. Faloutsos e D. W. Oard. *A survey of information retrieval and filtering methods*. Rapp. tecn. 1998.

[65] G. Salton e M. Smith. «On the application of syntactic methodologies in automatic text analysis». In: *Proceedings of the 12th annual international ACM SIGIR conference on Research and development in information retrieval*. 1989, pp. 137–150.

[66] J. J. Rocchio Jr. «Relevance feedback in information retrieval». In: *The SMART retrieval system: experiments in automatic document processing* (1971).

[67] M. Pazzani e D. Billsus. «Learning and revising user profiles: The identification of interesting web sites». In: *Machine learning* 27 (1997), pp. 313–331.

[68] N. Littlestone e M. K. Warmuth. «The weighted majority algorithm». In: *Information and computation* 108.2 (1994), pp. 212–261.

[69] R. Baeza-Yates, B. Ribeiro-Neto et al. *Modern information retrieval*. Vol. 463. 1999. ACM press New York, 1999.

[70] R. J. Mooney, P. N. Bennett e L. Roy. «Book recommending using text categorization with extracted information». In: *Proc. Recommender Systems Papers from 1998 Workshop, Technical Report WS-98-08*. Vol. 1188. Citeseer. 1998.

[71] B. Sheth e P. Maes. «Evolving agents for personalized information filtering». In: *Proceedings of 9th ieee conference on artificial intelligence for applications*. IEEE. 1993, pp. 345–352.

[72] D. Billsus e M. J. Pazzani. «User modeling for adaptive news access». In: *User modeling and user-adapted interaction* 10 (2000), pp. 147–180.

[73] C. C. Aggarwal, J. L. Wolf, K.-L. Wu e P. S. Yu. «Horting hatches an egg: A new graph-theoretic approach to collaborative filtering». In: *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*. 1999, pp. 201–212.

[74] J. S. Breese, D. Heckerman e C. Kadie. «Empirical analysis of predictive algorithms for collaborative filtering». In: *arXiv preprint arXiv:1301.7363* (2013).

[75] D. M. Pennock, E. J. Horvitz, S. Lawrence e C. L. Giles. «Collaborative filtering by personality diagnosis: A hybrid memory-and model-based approach». In: *arXiv preprint arXiv:1301.3885* (2013).

[76] M. Balabanović e Y. Shoham. «Fab: content-based, collaborative recommendation». In: *Communications of the ACM* 40.3 (1997), pp. 66–72.

[77] C. Basu, H. Hirsh, W. Cohen et al. «Recommendation as classification: Using social and content-based information in recommendation». In: *Aaai/iaai*. 1998, pp. 714–720.

[78] I. Soboroff e C. Nicholas. «Combining content and collaboration in text filtering». In: *Proceedings of the IJCAI*. Vol. 99. 1999. sn. 1999, pp. 86–91.

[79] A. Popescul, L. H. Ungar, D. M. Pennock e S. Lawrence. «Probabilistic models for unified collaborative and content-based recommendation in sparse-data environments». In: *arXiv preprint arXiv:1301.2303* (2013).

[80] A. I. Schein, A. Popescul, L. H. Ungar e D. M. Pennock. «Methods and metrics for cold-start recommendations». In: *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*. 2002, pp. 253–260.

[81] R. Burke, B. Mobasher e R. Bhaumik. «Limited knowledge shilling attacks in collaborative filtering systems». In: *Proceedings of 3rd international workshop on intelligent techniques for web personalization (ITWP 2005), 19th international joint conference on artificial intelligence (IJCAI 2005)*. 2005, pp. 17–24.

[82] S.-M. Moosavi-Dezfooli, A. Fawzi e P. Frossard. «Deepfool: a simple and accurate method to fool deep neural networks». In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 2574–2582.

[83] G. Elsayed, S. Shankar, B. Cheung, N. Papernot, A. Kurakin, I. Goodfellow e J. Sohl-Dickstein. «Adversarial examples that fool both computer vision and time-limited humans». In: *Advances in neural information processing systems* 31 (2018).

[84] N. Papernot, P. McDaniel e I. Goodfellow. «Transferability in machine learning: from phenomena to black-box attacks using adversarial samples». In: *arXiv preprint arXiv:1605.07277* (2016).

[85] Q. Wang, W. Guo, K. Zhang, A. G. Ororbia, X. Xing, X. Liu e C. L. Giles. «Adversary resistant deep neural networks with an application to malware detection». In: *Proceedings of the 23rd ACM sigkdd international conference on knowledge discovery and data mining*. 2017, pp. 1145–1153.

[86] V. Agate, P. Ferraro, S. Gaglio, G. L. Re e M. Morana. «VASARI Project: a Recommendation System for Cultural Heritage». In: *Proc. of the 5th Italian Conference on ICT for Smart Cities And Communities (I-CiTies 2019)*. 2019, pp. 1–3.

[87] L. Huang, A. D. Joseph, B. Nelson, B. I. Rubinstein e J. D. Tygar. «Adversarial machine learning». In: *Proceedings of the 4th ACM workshop on Security and artificial intelligence*. 2011, pp. 43–58.

[88] F. Tramèr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh e P. McDaniel. «Ensemble adversarial training: Attacks and defenses». In: *arXiv preprint arXiv:1705.07204* (2017).

[89] B. Biggio, B. Nelson e P. Laskov. «Poisoning attacks against support vector machines». In: *arXiv preprint arXiv:1206.6389* (2012).

[90] C. Cortes e V. Vapnik. «Support-vector networks». In: *Machine learning* 20.3 (1995), pp. 273–297.

[91] B. Biggio, G. Fumera e F. Roli. «Multiple classifier systems for adversarial classification tasks». In: *Multiple Classifier Systems: 8th International Workshop, MCS 2009, Reykjavik, Iceland, June 10-12, 2009. Proceedings 8*. Springer. 2009, pp. 132–141.

[92] S. Taheri, A. Khormali, M. Salem e J.-S. Yuan. «Developing a robust defensive system against adversarial examples using generative adversarial networks». In: *Big Data and Cognitive Computing* 4.2 (2020), p. 11.

[93] I. J. Goodfellow, J. Shlens e C. Szegedy. «Explaining and harnessing adversarial examples». In: *arXiv preprint arXiv:1412.6572* (2014).

[94] A. De Paola, A. Giammanco, G. Lo Re e M. Morana. «VASARI Project: Blended Recommendation for Cultural Heritage». In: *Proc. of the 6th Italian Conference on ICT for Smart Cities And Communities (I-CiTies 2020)*. 2019, pp. 1–3.

[95] P. Ferraro e G. Lo Re. «Designing ontology-driven recommender systems for tourism». In: *Advances onto the Internet of Things: How Ontologies Make the Internet of Things Meaningful* (2014), pp. 339–352.