UNIVERSITÀ DEGLI STUDI DI MILANO

FACOLTÀ DI SCIENZE E TECNOLOGIE

Corso di Laurea in Informatica Musicale

Mooncloud recommendation system

Relatore:
Claudio Agostino Ardagna
Correlatore:
Nome COGNOME

Tesi di Laurea di:
Andrea Michele Albonico
Matricola: 886667

Anno Accademico 2019/2020

Ringraziamenti

Andrea Michele Albonico

Prefazione

I sistemi di raccomandazione (*Recommendation System*) hanno avuto un forte sviluppo negli ultimi decenni e nascono proprio con lo scopo di identificare quegli oggetti (detti generalmente *item*) all'interno di un vasto mondo di informazioni che possono essere di nostro interesse e tanto maggiore è il grado di conoscenza dell'individuo e tanto più vengono ritenuti affidabili.
Il motivo di questo successo risiede nella riuscita integrazione di tali sistemi in applicazioni commerciali, soprattutto nel mondo dell'E-commerce e nel fatto che sono in grado di aiutare un utente a prendere una decisione che sia la scelta di un film per l'uscita con gli amici il sabato sera, di una playlist da ascoltare durante un viaggio in auto o in un momento di lettura, e via discorrendo.
MoonCloud è una piattaforma erogata come servizio che fornisce un meccanismo di *Security Governance* centralizzato. Garantisce il controllo della sicurezza informatica in modo semplice e intuitivo, attraverso attività di test e monitoraggio periodiche e programmate (*Security Assurance*). L'obbiettivo di questa tesi è stato quello di aggiungere, al già presente sistema per la scelta dei controlli all'interno delle attività di test, un sistema di raccomandazioni che possa consigliare all'utente delle possibili *evaluation* rispetto ai dati relativi al target indicato; in questo modo anche l'utente meno esperto può usufruire dei servizi offerti da MoonCloud in modo semplice e intuitivo.

La tesi è organizzata come segue:

Capitolo 1 – Introduzione a MoonCloud descrizione e funzionamento della piattaforma MoonCloud in ambito di Security Assurance.

Capitolo 2 – Tecnologie studi e analisi di soluzioni esistenti, studi delle tecnologie utilizzate nel seguito del lavoro.

Capitolo 3 – Descrizione delle attività svolte per conseguire gli obiettivi: Descrivere le attività svolte, riportando attivita, tempi, strumenti utilizzati, risultati conseguiti, problemi affrontati e modalita di risoluzione. Potranno essere qui descritte le attività anche dal punto di

vista strettamente tecnico, approfondendo le scelte effettuate, le motivazioni, le alternative prese in considerazione, l'uso o il possibile uso dei risultati del lavoro.

Capitolo 4 – Presentazione dei risultati e conclusioni] La presentazione dei risultati dovrebbe consistere in una descrizione tecnica dei risultati raggiunti,unitamente ad un commento critico e ad un'analisi della rispondenza agli obiettivi iniziali (si consiglia pertanto di motivare la rilevanza dei risultati e l'eventuale scostamento dagli obiettivi iniziali). La sezione relativa ai risultati dovrebbe infine contenere una sintesi critica e un giudizio sull'esperienza effettuata, che renda conto di aspetti positivi e negativi per il tirocinante e per l'ente ospitante, del valore formativo, professionale e umano, così via.

Indice

Elenco delle figure

Capitolo 1

Introduzione

In questo capitolo verrà descritto in modo più approfondito il funzionamento della piattaforma Moon Cloud e unitamente al motivo dell'implementazione della soluzione proposta.

1.1 MoonCloud overview

La diffusione di sistemi ICT (*Information and Communications Technology*) nella maggiorparte degli ambienti lavorativi e privati in termini di servizi offerti, automazione di processi e incremento delle performance. L'uso di questa tecnologia ha assunto importanza a partire dagli anni novanta come effetto del boom di Internet. Oggi le professionalità legate all'ICT crescono in numero e si evolvono per specificità, per operare in ambienti fortemente eterogenei ma sempre più interconnessi fra di loro come il cloud computing, i social newtwork, il marketing digitale, i sistemi IoT, la realtà virtuale, etc.

Gli immensi benefici del cloud in termini di flessibilità, consumo delle risorse e gestione semplificata, la rende la prima scelta per utenti e industrie per il deploy dei loro sistemi IT. Tuttavia il cloud computing solleva diverse problematiche legate alla mancanza di fiducia e trasparenza dove i clienti necessitano di avere delle garanzie sui servizi cloud ai quali si affidano; spesso i fornitori di servizi cloud non fornisco ai clienti le specifiche riguardanti le misure di sicurezza messe in atto.

Negli ultimi anni, sono state sviluppate tecniche e modi per rendere sicuri questi sistemi e proteggere i dati degli utenti, portando alla diffusione di approcci eterogenei che incrementaro la confusione negli utenti. Tecniche tradizionali di verifica della sicurezza basati su approcci di analisi statistica non sono più sufficenti e devono essere integrati con processi di raccolta di evidence da sistemi cloud in produzione e funzionamento. In generale il

cloud security definisce i modi (ad esempio criptazione, controllo degli accessi, etc.) per proteggere attivamente gli asset da minaccie interne ed esterne, e fornire un ambiente in cui i clienti possano affidarsi e interagire in totale sicurezza. Ma per rendere il cloud degno di fiducia e trasparente, sono state introdotte tecniche di *security assurance* le quali sono definite come il modo per ottenere la fiducia necessaria nelle infrastrutture e/o nelle applicazioni di dimostrare che siano garantite delle proprietà di sicurezza, e che operi normalmente anche se subisce attacchi; grazie alla raccolta e allo studio di queste evidence è possibile che venga accertata la validità e efficenza delle proprietà di sicurezza.

Il prezzo che paghiamo per i benefici di queste tecnologie è dato dall'incremento di violazioni di sicurezza, che oggigiorno preoccupa tutte le aziende e anche i loro clienti, con l'incremento del rischio di fallimento per i servizi più importanti dovuti a violazioni della privacy e al furto di dati.
Il mercato sta lentamente notando che non è l'inadequamento tecnologico dei sistemi di sicurezza che incrementa il rischio di furti di dati o delle violazioni di sicurezza; piuttosto, la mal configurazione e l'errata integrazione di questi sistemi nei processi di business. [2]

Per questo motivo anche se vengono usati i sistemi di sicurezza e di controllo migliori non è possibile garantire la sicurezza; ma è necessario implementare un processo continuo di diagnostica che verifica che i controlli siano configurati in modo corretto e il loro comportamento sia quello aspettato.

Il *Security Assesment* diventa allora un aspetto importante specialmente negli ambienti cloud e IoT. Questo processo deve essere portato avanti in modo continuo e olistico, per correlare le evidence raccolte da sempre maggiori meccanismi di protezione. [1]
Moon Cloud è una soluzione PaaS (Platform as a Service) che fornice una piattaforma B2B (Business To Business) innovativa per verifiche, diagnostiche e monitoraggio dell'adeguatezza dei sistemi ICT rispetto alle politiche di sicurezza, in modo continuo e su larga scala. Moon Cloud supporta una semplice ed efficente *ICT security governance*, dove le politiche di sicurezza possono essere definite dalle compagnie stesse (a partire da un semplice controllo sulle vulnerabilità a linee guida di sicurezza interna), da entità esterne, imposte da standard oppure da regolamentazioni nazionali/internazionali.
La sicurezza di un sistema o di un insieme di asset dipende solo parzialmente dalla forza dei singoli meccanismi di protezione isolati l'uno dall'altro; infatti, dipende anche dall'abilità di questi meccanismi di lavorare continuamente in sinergia per provvedere a una protezione olistica. In più, quando i sistemi cloud e i servizi IoT sono coinvolti, le dinamiche di questi servizi e la loro rapida evoluzione rende il controllo dei processi all'interno dell'azienda e le politiche di sicurezza più complesse e prone ad errori.

I requisiti ad alto livello fondamentali per poter garantire le Security Assurance sono:

sistema olistico è richiesta una visione globale e pulita dello stautus dei sistemi di sicurezza; inoltre è cruciale distribuire lo sforzo degli specialisti in sicurezza per migliorare il processo e le politiche messe in atto. Si parte da delle valutazioni fatte manualmente a quella semi-automatiche che ispezionano i meccanismi di sicurezza.

monitoraggio continuo ed efficiente è necessario un controllo continuo che valuti l'efficenza dei sistemi di sicurezza per ridurre l'impatto dell'errore umano, soprattutto dal punto di vista organizzativo. La mancata configurazione dovuta al cambiamento dell'ambiente, la coesistenza di componenenti in conflitto: sono scenari che richiedono un monitoraggio e un aggiornamento continuo.

singolo punto managment avere un solo punto in cui gestire tutti gli aspetti relativi alla sicurezza, permette di avere sotto controllo le politiche di sicurezza. Inoltre disporre di un inventario degli asset da proteggere, così da poter conoscere quali protezioni applicare.

reazioni rapide a incidenti di sicurezza spesso la reazione ad incidenti di sicurezza è ritardata da due fattori: il tempo richiesto per rilevare l'incidente e il tempo per analizzare il motivo dell'accaduto.

Moon Cloud è basato su una tecnica di Security Assurance garantendo che tutte le attività aziendali si compiano seguendo i requisiti prestabiliti da appropriate politiche e procedure.

Una *Security Compliance Evaluation* è il processo di verifica a cui un target viene sottoposto e il cui risultato deve soddisfare i requisiti richiesti da standard e politiche. A partire da questi processi di verifica, che devono a loro volta essere affidabili, si ottendo delle evidence; queste ultime possono essere raccolte monitorando l'attività del target oppure, come già menzionato, sottoponendo il target a scenari critici o di testing. In particolare, una Security Compliance Evaluation è un processo che verifica l'uniformità di un certo target a una o più politiche attraverso una serie di controlli che a seconda dvalore booleano associato ad ogni controllo viene prodotto un valore booleano per le politiche.

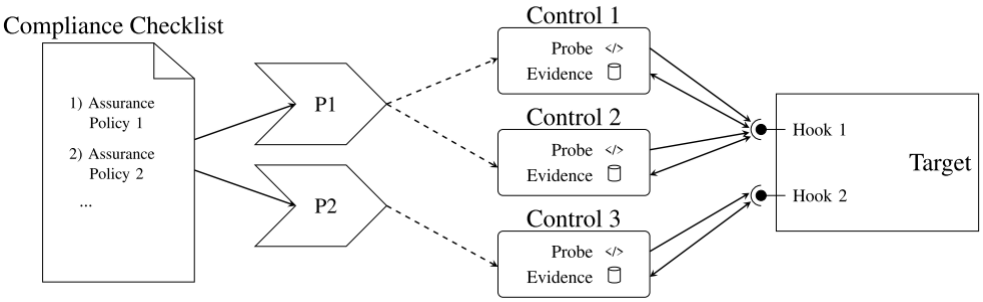

Figura 1.1: Security Compliance Evaluation

1.2 Processo di Evaluation

Moon Cloud implementa il processo di Security Compliance Evaluation in Figura 1.1 usando controlli di monitoraggio o di test personalizzabile. Inoltre garantisce, oltre a tutti i requisisti ad alto livello elencati prima, anche i seguenti:

Moon Cloud è una piattaforma cloud centralizzata presentando una visione olistica dello stato di sicurezza di un dato sistema.

Moon Cloud implementa un sistema di Assurance Evidence-based continuo, implementato come processo di Compliance, basato su politiche custom o standard.

Moon Cloud è offerto come un servizio - PaaS, dove le attività di evaluation possono essere facilmente ed efficentemente configurate su un target asset, senza l'intervento dell'uomo.

Moon Cloud permette di schedulare delle ispezioni automatiche, grazie all'inventario di asset protetto.

Moon Cloud evaluation engine può ispezionare dall'interno, gestendo così delle minaccie interne; permettendo anche reazioni rapide a incidenti di sicurezza e veloci rimedi, grazie alla raccolta continua di evidence.

L'architettura di Moon Cloud è costituita da un'Assurance Manager che gestisce i processi di Evaluation attraverso un set di *Execution Cluster*; ognuno dei quali gestisce ed esegue un set di probe che collezionano le evidence necessarie per le evaluation. Tutte le attività di collezione sono eseguite dal probe. Ogni probe è uno script di python fornito come una singola immagine di Docker, che viene inizializzata quando è triggerata una evaluation ed è

distrutta quando il processo di evaluation è terminato.
Accedendo alla piattaforma di Moon Cloud, l'utente può definire le proprie politiche di sicurezza e attività di evaluation come espressioni booleane di controlli di sicurezza e altre politiche predefinite. Una volta che una politica viene definita, l'utente può decidere quando schedulare l'evaluation; e nel momento in cui un processo di evaluation viene inizializzato, tutti i controlli vengono eseguiti e i risultati dell'espressioni booleane vengono memorizzati e restituiti all'utente. A questo punto l'utente può accedere a questi risultati a diversi gradi di precisione: una visione sommaria e generale di tutte le politche implentate e dello stato generale del sistema di sicurezza, al risultato di una specifica politica oppure alle evidence raccolte per una evaluation.

Per poter rendere ancora più intuituivo e semplice da utilizzare un sistema di questa importanza, si è pensato di introdurre un sistema che possa raccomandare agli utenti, in base agli asset forniti che vuole proteggere e monitorare, una serie di evaluation o politihe da applicare in quei casi; questo permette anche a utenti meno esperti di poter configurare in modo rapido ed efficente dei meccanismi di monitoraggio da minaccie.

Capitolo 2

Tecnologie

In questo capitolo verranno descritte le attività preliminari per la realizzazione di questo progetto, le tecnologie utilizzate unitamente alle motivazioni legate all'uso di questi sistemi rispetto ad altri.

2.1 Strutture dati gerarchiche

Le tabelle di un database relazione non sono gerarchiche (come nel XML), ma sono delle semplici liste piatte. I dati gerarchici sono constituiti da relazioni padre-figlio che non possono essere rappresentate in modo naturale nelle tabelle dei database relazionali. In questo caso, i dati gerarchici sono una collezione di informazioni dove ogni item ha un solo padre e nessuno o più figli (ad eccezione del nodo radice che non ha un nodo padre); questo genere di rappresentazione delle informazioni può essere trovato in diversi ambiti di applicazione di un database, incluse discussioni su forum e mailing list, grafici di organizzazione di un business, categorie per gestire contenuti e categorie di prodotti.

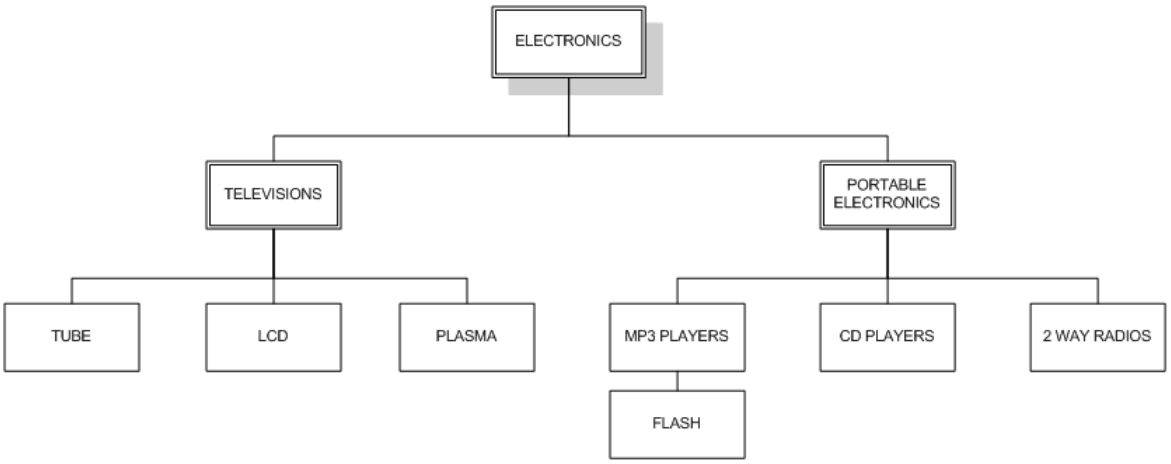

Figura 2.1: Esempio di una gestione di dati in modo gerarchico

Ci sono differenti modelli per poter gestire dati in modo gerarchico, i più importanti che sono stati presi in considerazione sono i seguenti:

2.1.1 The adjacency list model

Il primo approccio, e quello di più semplice implementazione, qui descritto è chiamato '*adjacency list model* o metodo ricorsivo; è definito tale perchè per funzionare necessita solo di una funzione che itera per tutto l'albero.
In questo modello, ogni item (nodo dell'albero) nella tablla contiene un puntatore al suo item padre; invece il nodo radice avrà un puntatore a un valore NULL per l'item padre.

Il vantaggio di usare questo modello sta nella sua semplicità di costruzione sopratutto a livello di codice client-side, e di restituzione dei figli di un nodo. Mentre diventa problematico se si lavora in puro SQL e nella maggior parte dei linguaggi di programmazione, è lento e poco efficente, perchè è necessaria una query per ogni nodo dell'albero, e visto che ogni query impiega un certo periodo di tempo, questo rende la funzione molto lente quando si lavora con alberi di grandi dimensioni. Inoltre molti linguaggi non sono ottimizzati per funzioni ricorsive. Per ogni nodo, la funzione inizia una nuova istanza di se stessa, ogni istanza occupa una porzione di memoria e impiega un certo tempo per inizializzarsi, e più grande è l'albero e più questo processo sarà portato a termine in maggior tempo.

```
CREATE TABLE category(
        category_id INT AUTO_INCREMENT PRIMARY KEY,
        name VARCHAR(20) NOT NULL,
        parent INT DEFAULT NULL
);

INSERT INTO category VALUES(1,'ELECTRONICS',NULL),(2,'TELEVISIONS',1),(3,'TUBE',2),
        (4,'LCD',2),(5,'PLASMA',2),(6,'PORTABLE ELECTRONICS',1),(7,'MP3 PLAYERS',6),
(8,'FLASH',7),
        (9,'CD PLAYERS',6),(10,'2 WAY RADIOS',6);

SELECT * FROM category ORDER BY category_id;
+-------------+----------------------+--------+
| category_id | name                 | parent |
+-------------+----------------------+--------+
|           1 | ELECTRONICS          |   NULL |
|           2 | TELEVISIONS          |      1 |
|           3 | TUBE                 |      2 |
|           4 | LCD                  |      2 |
|           5 | PLASMA               |      2 |
|           6 | PORTABLE ELECTRONICS |      1 |
|           7 | MP3 PLAYERS          |      6 |
|           8 | FLASH                |      7 |
|           9 | CD PLAYERS           |      6 |
|          10 | 2 WAY RADIOS         |      6 |
+-------------+----------------------+--------+
```

Figura 2.2: Esempio di una tabella per gestire dati in modo gerarchico secondo l'adjacency list model

2.1.2 The Nested set model

Il secondo approccio che viene proposto è il *Nested set model*, che permette di osservare la gerarchia in un modo diverso, non come nodi e linee, ma come container innestati.

La gerarchia dei dati viene rappresentata nella tabella attraverso l'uso degli attributi 'left' e 'right' per rappresentare l'annidamento dei nodi (il nome delle colonne: left e right, hanno significati speciali in SQL; per questo motivo si identificano questi campi con i nomi 'lft' e 'rght'). Ogni nodo dell'albero viene visitato due volte, assegnando i valori in ordine di visita, e in entrambe le visite. Quindi vengono associati ad ogni nodo due numeri, memorizzato come due attributi. I valori di left e right sono determinati come segue: si inizia a numerare a partire dal lato più a sinistra di ogni nodo e si continua verso destra. Lavorando con un albero, si parte da sinistra

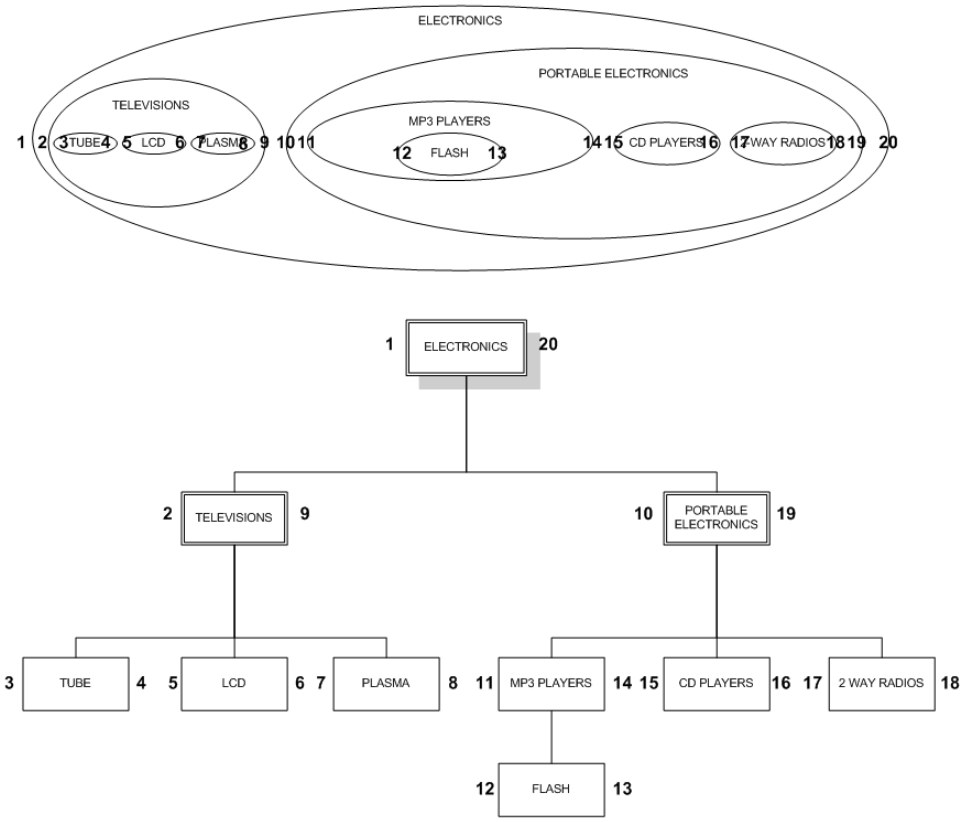

Figura 2.3: Esempio di una gestione di dati in modo gerarchico secondo il Nested set model

e si continua verso destra, un livello alla volta, scendendo per ogni nodo i suoi figli, assegnando i valori al campo left, prima di assegnare un valore al campo right, e successivamente si continua verso destra. Questo approccio è chiamato Modified preorder tree traversal algorithm.

A prima vista questo approccio può sembrare più complicato da comprendere rispetto all'adjacency list model, ma quest'utlimo metodo è molto più veloce quando si vuole recuperare i nodi, visto che basta una query, mentre più lento per operazioni di aggiornamento e cancellazione dei nodi; in quest ultimo il grado di complicatezza dell'operazione è determinato dal nodo che si vuole cancellare, a partire dal caso più semplice, il nodo foglia (nodo senza figli) fino al caso più complicato, quando si vuole cancellare il nodo radice.

```
CREATE TABLE nested_category (
        category_id INT AUTO_INCREMENT PRIMARY KEY,
        name VARCHAR(20) NOT NULL,
        lft INT NOT NULL,
        rgt INT NOT NULL
);

INSERT INTO nested_category VALUES(1,'ELECTRONICS',1,20),(2,'TELEVISIONS',2,9),(3,'TUBE',3,4),
 (4,'LCD',5,6),(5,'PLASMA',7,8),(6,'PORTABLE ELECTRONICS',10,19),(7,'MP3 PLAYERS',11,14),
(8,'FLASH',12,13),
 (9,'CD PLAYERS',15,16),(10,'2 WAY RADIOS',17,18);

SELECT * FROM nested_category ORDER BY category_id;

+-------------+----------------------+-----+-----+
| category_id | name                 | lft | rgt |
+-------------+----------------------+-----+-----+
|           1 | ELECTRONICS          |   1 |  20 |
|           2 | TELEVISIONS          |   2 |   9 |
|           3 | TUBE                 |   3 |   4 |
|           4 | LCD                  |   5 |   6 |
|           5 | PLASMA               |   7 |   8 |
|           6 | PORTABLE ELECTRONICS |  10 |  19 |
|           7 | MP3 PLAYERS          |  11 |  14 |
|           8 | FLASH                |  12 |  13 |
|           9 | CD PLAYERS           |  15 |  16 |
|          10 | 2 WAY RADIOS         |  17 |  18 |
```

Figura 2.4: Esempio di una tabelle per la gestione di dati in modo gerarchico secondo il Nested set model

2.2 Sistemi di raccomandazione

Recommendation system is a system which recommends items to users among a large number of existing items in database. Item is anything which users consider, such as product, book, and newspaper. There is expectation that recommended item are items that user will like most; in other words, such items are in accordance with user's interest. There are two common trends of recommendation systems: content-based filtering (CBF) and collaborative filtering (CF) as follows [1, pp. 3-13]: - CBF recommends an item to a user if such item is similar to other items that she/he likes much in the past (her/his rating for such item is high). Note that each item has contents which are properties and so all items compose a so-called item content matrix. - CF recommends an item to a user if her/his neighbors (other users similar to her/him) are interested in such item. Note that user's rating on an item expresses her/his interest. All users' ratings on items compose a so-called rating matrix. Both of them (CBF and CF) have their own strong points and weak points. Namely CBF focuses on content of item and user's own interest; it recommends different items to different users. Each user can receive unique recommendation; so this is the strong point of CBF. However

CBF doesn't tend towards community like CF. As items that user may like "are hidden under" user community, CBF has no ability to discover such implicit items. This is the most common weak point of CBF. If there are a lot of content associating with item (for example, items has many properties) then, CF consumes much system resource and time in order to analyze items whereas CF doesn't regard to content of items. That CF only works on users' ratings on items is strong point because CF doesn't encounter how to analyze rich content items. However it is also weak point because CF can do unexpected recommendation in some situations that items are considered to be suitable to user but they don't relate to user profile in fact. The problem gets more serious when there are many items that aren't rated and so rating matrix becomes spares matrix containing many missing values.

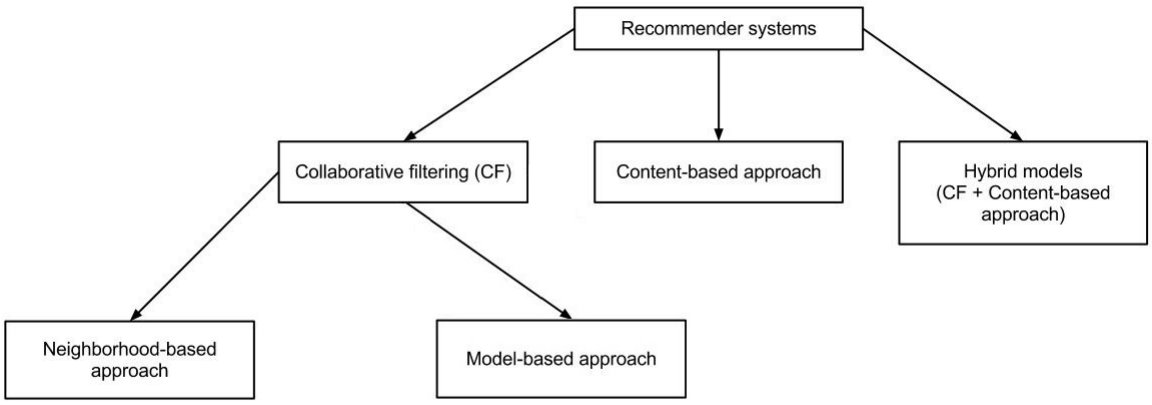

Figura 2.5:

[3] Un sistema di raccomandazione filtra i dati usando differenti algoritmi e raccomanda gli item più rilevanti agli utenti, attraverso un procedimento a 3 fasi:

raccolta di dati : questa è il primo step e anche quello più importante per poter costruire un sistema di raccomandazione che produca risultanti rilevanti e consistenti. I dati possono essere raccolti in due modi: esplicitamente, cioè attraverso i dati che vengono prodotti direttamente dagli utenti, ad esempio le valutazioni di un prodotto; mentre attarverso l'approccio implicito, vengono raccolti dati che non sono prodotti in modo intenzionale dall'utente ma raccolti dai costanti flussi di dati come la cronologia di ricerca, i click effettuati, lo storico degli ordini, etc.

memorizzazione di dati : la quantità di dati definisce quanto efficace un modello di raccomandazione possa di diventare. Ad esempio, in un sistema di raccomanzione per film, maggiori sono le valutazioni fornite dagli utenti, e migliore sarà il sistema di raccomandazione per gli altri utenti. Il tipo di dati che si vuole raccogliere determina anche il supporto di memorizzazione più adatto.

Filtraggio dei dati : dopo la fase di raccolta e memorizzazione dei dati, essi vanno filtrati per poter estrarre le informazioni rilevanti per poter effettuare le raccomandazioni finali, e sono già disponibili diversi algoritmi che semplificano quest ultima fase.

I sistemi di raccomandazione possono essere suddivisi nelle seguenti categorie, ma speso si preferisco degli approcci imbridi cioè delle combinazioni di sistemi di raccomandazione basati sul contenuto (*Content-based filtering*) e quelli collaborativi (*Collaborative filtering*) in modo da essere più efficaci sfruttando i pregi di entrambi gli approcci.

2.2.1 Content-based filtering

Un Content-based filtering è un sistema di raccomandazione in cui vengono suggeriti item simili a un particolare item (oggetti o prodotti).

Questo approccio sfrutta i metadati dell'item, che possono essere il genere, una descrizione, uno o più autori, la categoria di appartenenza etc. per fare queste raccomandazioni; l'idea base che sta dietro questi raccomandatori, è che se ad un utente piace o interessa un particolare item allora gli piaceranno anche altri item simili.

Questo algoritmo suggerisce prodotti che piacevano all'utente nel passato ed è limitato a item dello stesso tipo. Un content-based recommender fa riferimento a quegli approcci, che provvedono raccomandazioni comparano la rappresentazione del contenuto che descrive un item e la rappresentazione del contenuto dell'item interessato dall'utente.

Questi metodi sono usati quando si sanno a priori delle informazioni sugli item che si vuole suggerire, ma non sugli utenti. In questo sistema, delle keyword (parole chiave) sono utilizzate per caratterizzare gli item e un profilo dell'utente è costruito per indicare quali item gli piacciono. In altre parole, questi algoritmi cercano di raccomandare item che all'utente sono piaciuti o ha usato nel passato e sta esaminando nel presente. La costruzione del profilo dell'utente, spesso temporaneo, non viene basata su un modulo di registrazione che l'utente stesso deve compilare, ma su informazioni lasciate indirettamente dall'utente. Più precisamente, tra vari item candidati da

raccomandare all'utente si passa per un processo di confronto con gli item piaciuti dall'utente e gli item migliori vengono suggeriti.

2.2.2 Collaborative filtering

I filtri collaborativi (*Collaborative filtering*) lavorano costruendo un database di preferenze di utenti su item (o prodotti), sfruttano tecniche di analisi dei dati al problema di aiutare gli utenti a trovare gli item che gli potrebbero piacere producendo una lista dei top-N item da raccomandare per un dato utente. Un nuovo utente subisce un processo di matching all'interno del database per scoprire quali sono i possibili vicini (*neighbors*), che corrispondo agli altri utenti aventi storicamente simili preferenze al nuovo utente. Agli item maggiormente preferiti dai vicini sono raccomandati al nuovo utente, visto che potrebbero essere di suo interesse.

Questi sistemi tentano di predirre la valutazione o la preferenza che un utente darebbe a un item basandosi su preferenze date da altri utenti, queste preferenze possono essere ottenute o in modo esplicito dagli utenti o tramite qualche misurazione implicita. I filtri collaborativi non richiedono l'uso di metadati associati agli item come nella loro controparte, i filtri content-based. A un utente vengono raccomandati item basandosi su valutazioni passate collezionate da altri utenti.

Tuttavia, restano ancora oggi alcune sfide significative a cui sono sottoposti i sistemi di raccomandazione basati su filtraggio collaborativo. Il primo obbiettivo è quello di migliorare la scalabilità degli algoritmi di filtri collaborativi; questi algoritmi sono in grado di cercare anche diecimila di potenziali vicini (utenti simili) in tempo reale, ma la richiesta dei sistemi moderni è di cercare dieci milioni di potenziali vicini. Algoritmi esistenti hanno problemi di performance con i singoli utenti quando essi hanno molte informazioni. Il secondo obbiettivo è quello di migliorare la qualità dei sistemi di raccomandazione per gli utenti. Gli utenti vogliono raccomandazioni di cui possono fidarsi e che possono aiutarli a trovare item che potrebbero essere di loro gusto. Per certi versi questi due obbiettivi sono in conflitto tra di loro e per ottenere dei risultati validi e di una certa importanza è necessario trattarli in contemporanea perchè aumentare solamente la scalabilità diminuirebbe la sua qualità e viceversa. [5]

Il principale modello di filtro collaborativo studiato in questo elaborato è il metodo definito come *Memory-based* e il vantaggio di utilizzare queste tecniche sta nel fatto di essere semplici da implementare e i risultati ottenuti sono altrettanto semplici da spiegare; mentre ci possono essere anche filtri collaborativi che sfruttano metodi *Model-based* che si basano sulla fattorizzazione di matrici e sono molto più funzionali per gestire il problema della

sparsità dei dati. Questi ultimi sono sviluppati usando algoritmi di data mining e machine learning per predirre le valutazioni di utenti su item senza valutazioni, inoltre sono spesso associati a tecniche come la dimensionality reduction per migliorare la precisione.

Memory-based
I filtri collaborativi Memory-based sono stati introdotti per via delle osservazioni che vennero fatti sugl utenti, i quali si fidano maggiormente delle raccomandazioni di altri che la pensano allo stesso modo. Questi metodi mirano a calcolare le relazioni tra utenti e item attraverso lo schema dei vicini che identifica sia coppie di item che tendono ad essere usati insieme o hanno un grado di similarità alto o utenti con uno storico di item usati simile. [4] Questi approcci divennero molto famosi grazie alla loro semplicità di implementazione, molto intuitivi, non necessitano il training e aggiustamento di molti parametri, e l'utente può capire la ragione che sta dietro ogni raccomandazione

Filtri collaborativi che usano metodi Memory-based (definiti anche Neighborhood-based) possono essere classificati in altre due categorie:

User-based filtering
Questi sistemi, definiti anche con l'acronimo UB-CF (*User-based Collaborative Filter*) raccomandano una serie di item a un utente che utenti simili hanno usato o valutato. Questo algoritmo prima trova un valore che rappresenta la similarità tra utenti. E basandosi su questi valori, prende uno o più utenti tra quelli che risultano simili e raccomanda item che questi utenti simili hanno usato o valutato in precedenza.

Molti di questi approcci possono essere generalizzati dall'algoritmo definito dai seguenti step:

1. Specificare qual'è il l'utente a cui si vuole applicare l'algoritmo di raccomandazione e recuperare quali utenti possono avere dato valutazioni o usato item simili al utente target. Piuttosto che recuperare tutti gli utenti, per velocizzare l'esecuzione dell'algoritmo, è possibile selezionare soltanto un gruppo di utenti in modo casuale oppure associare dei valori di similarità tra tutti gli utenti e confrontando questi valori con quello dell'utente target, selezionare i relativi utenti che superano una soglia scelta, oppure utilizzare tecniche di clustering. un numero limitato per effettuare la raccomandazione,

2. Estrarre quegli item a cui l'utente target non ha mai interagito e per questo motivo gli possono interessare, e mostrarli all'utente target.

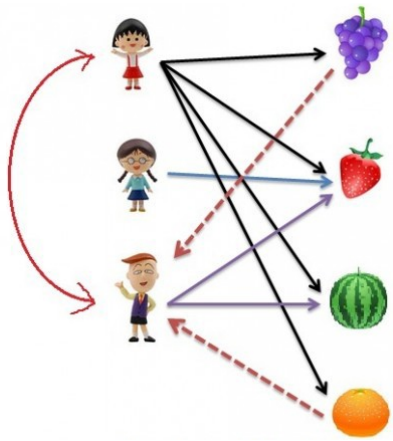

Figura 2.6: Esempio di applicazione di un sistema di raccomandazione User-based

Questi approcci sono facilmente implementabili, indipendenti dal contesto in cui sono applicati e possono essere più accurati rispetto a tecniche basate sul content-based dall'altra parte all'aumentare del numero di utenti che vado a considerare per fare le raccomandazioni migliore è la precisione di questo processo ma anche è maggiore il costo per compiere questo procedimento. Altro problema che affligge questi sistemi, e che verrà approfondito nel prossimo paragrafo, è definito di Cold-start.

Item-based filtering

Quando viene applicato per milioni di utenti e item, l'algoritmo UB-CF non è molto efficente, per via della complessa computazione della ricerca di utenti simili; così in alternativa è stato introdotto l'algoritmo di filtraggio Item-based, definito anche IB-CF (*Item-based Collaborative Filter*): dove piuttosto che effetuare il confrotto tra utenti simili, viene fatto un confronto tra gli item dell'utente a cui si vule raccomandare e i possibili item simili.

Questi sistemi sono estremamente simili ai sistemi di raccomandazione Content-based, e identificano item simili in base a come utenti gli hanno usati nel passato.

[5]

Model-based filtering Model-based algorithm tries to compress huge database into a model and performs recommendation task by applying reference mechanism into this model. Model-based CF can response user's request instantly. This paper surveys common techniques for implementing model-

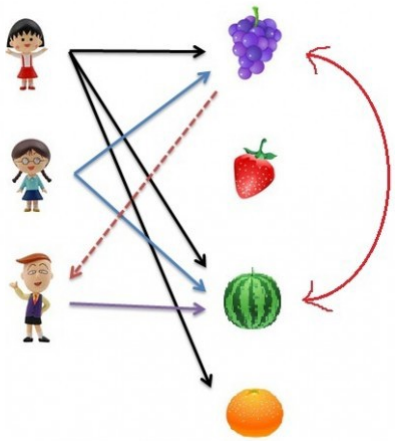

Figura 2.7: Esempio di applicazione di un sistema di raccomandazione Item-based

based algorithms. We also give a new idea for model-based approach so as to gain high accuracy and solve the problem of sparse matrix by applying evidence-based inference techniques.

2.2.3 Challenges and limitations

Cold start problem What will happen if a new user or a new item is added in the dataset? It is called a Cold Start. There can be two types of cold start: Visitor Cold Start: means that a new user is introduced in the dataset. Since there is no history of that user, the system does not know the preferences of that user. It becomes harder to recommend products to that user. So, how can we solve this problem? One basic approach could be to apply a popularity based strategy, i.e. recommend the most popular products. These can be determined by what has been popular recently overall or regionally. Once we know the preferences of the user, recommending products will be easier. Product Cold Start: means that a new product is launched in the market or added to the system. User action is most important to determine the value of any product. More the interaction a product receives, the easier it is for our model to recommend that product to the right user. We can make use of Content based filtering to solve this problem. The system first uses the content of the new product for recommendations and then eventually the user actions on that product.

Cons Scalability: The more K neighbors we consider (under a certain threshold), the better my classification should be. Nevertheless, the more users there are in the system, the greater the cost of finding the nearest K neighbors will be. Cold-start: New users will have no to little information about them to be compared with other users. New item: Just like the last point, new items will lack of ratings to create a solid ranking (More of this on 'How to sort and rank items').

Sparsity Stated simply, most users do not rate most items and, hence, the user ratings matrix is typically very sparse. is is a problem for collaborative +ltering systems, since it decreases the probability of +nding a set of users with similar ratings. This problem often occurs when a system has a very high item-to-user ratio, or the system is in the initial stages of use. This issue can be mitigated by using additional domain information or making assumptions about the data generation process that allows for high-quality imputation.

Capitolo 3

Sistemi di raccomandazione

Capitolo 4

Descrizione approfondita del progetto

Capitolo 5

Conclusioni

Bibliografia

[1] M. Anisetti et al. «A semi-automatic and trustworthy scheme for continuous cloud service certification». In: *IEEE TRANSACTIONS ON SERVICES COMPUTING* (2017). DOI: 10.1109/TSC.2017.2657505.

[2] M. Anisetti et al. «Moon Cloud: A Cloud Platform for ICT Security Governance». In: (dic. 2018), pp. 1–7. DOI: 10.1109/GLOCOM.2018.8647247.

[3] Minh-Phung Do, Dung Nguyen e Academic Network of Loc Nguyen. «Model-based approach for Collaborative Filtering». In: ago. 2010.

[4] Miquel Montaner, Beatriz López e Josep Lluís de la Rosa. «A Taxonomy of Recommender Agents on the Internet». In: *Artificial Intelligence Review* 19.4 (giu. 2003), pp. 285–330. ISSN: 1573-7462. DOI: 10.1023/A:1022850703159. URL: https://doi.org/10.1023/A:1022850703159.

[5] Badrul Sarwar et al. «Item-based Collaborative Filtering Recommendation Algorithms». In: WWW '01 (2001), pp. 285–295. DOI: 10.1145/371920.372071. URL: http://doi.acm.org/10.1145/371920.372071.