Màster Universitari en Enginyeria Informàtica

«SISTEMA DE RECOMANACIÓ DE PRODUCTES»

TREBALL FINAL DE MÀSTER
ÀREA D'INTEL·LIGÈNCIA ARTIFICIAL

MEMÒRIA FINAL
Juny de 2014

ALUMNE: Josep Lluís Amador Teruel
CONSULTOR: Samir Kanaan Izquierdo

ÍNDEX DE CONTINGUTS

1. INTRODUCCIÓ

L'objectiu d'aquest Treball Final de Màster (TFM) és dissenyar un sistema de recomanació de diferents tipus de productes, i que sigui capaç d'oferir recomanacions tenint en compte només algunes de les valoracions dels usuaris, segons ho indiquin ells en les seves preferències.

1.1. Per què un recomanador?

Actualment ens trobem en ple segle XXI, l'era del Big Data, i el volum d'informació que rebem des de tots els àmbits és cada vegada més ingovernable. És per això que cal buscar eines per filtrar les informacions que podrien ser més adequades per a cada persona.

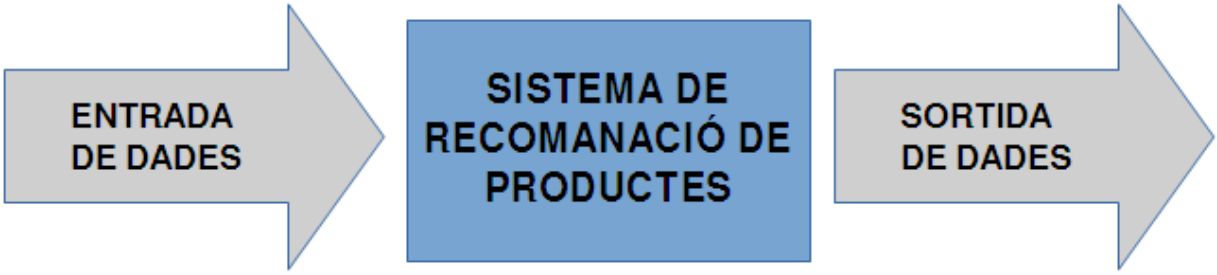
Un recomanador de productes que sigui configurable per diferents tipus ajudaria a les persones a perdre menys temps a l'hora de buscar els productes que més li podrien agradar. A partir de les recomanacions que es puguin generar pel propi recomanador, l'usuari podria afegir altres filtres per ajustar més la seva consulta, com per exemple, filtrar els productes per preu.

En aquest projecte es tindrà en compte la possibilitat de poder continuar desenvolupant l'eina en el futur, de manera que les seves dades es puguin explotar de forma comercial amb altres aplicacions web i/o mòbils.

1.2. Sistemes de recomanació de productes

Un sistema de recomanació de productes ha de ser capaç de gestionar les dades dels seus usuaris i productes, per a poder oferir les millors recomanacions possibles als usuaris que ho demanin.

Per a completar tot el circuit del sistema de recomanació de productes, també es proposa per aquest TFM el dissenyar una eina d'entrada de dades que faciliti als usuaris la introducció de les seves valoracions, i alhora, que l'exportació dels seus resultats sigui el més oberta possible per a que els puguin explotar fàcilment d'altres aplicacions.



L'entrada de dades al sistema de recomanació es poden fer per 2 motius diferents:

- Per fer peticions de recomanacions al sistema.
- Per afegir a la base de dades nous usuaris, productes i puntuacions.

Per el cas de la sortida de dades també podríem tenir 2 tipus de sortides diferents:

- Per respondre les peticions de recomanacions al sistema.
- Per generar informes i estadístiques de les dades incloses en el sistema de recomanació.

Un sistema de recomanació pot executar-se de formes diferents en funció de les dades que es disposen. Les recomanacions es poden fer en base a productes de característiques semblants, o en base a usuaris semblants que han valorat positivament uns productes. La manera de com es valora els productes per part dels usuaris també és un tema rellevant, ja que el recomanador funciona de forma diferent si es tenen en compte valoracions binaries del tipus "M'agrada / No m'agrada", o si es tenen en compte altres valoracions més detallades amb puntuacions de 0 a 10.

Hi ha diferents tipus de recomanadors:

- Basats en memòria: es disposa de totes les dades per a realitzar els càlculs del recomanador, i no es viable per a gran volums de dades, ja que en incorporar noves dades pel recomanador s'hauria de recalcular tot de nou.

- Basats en algorismes d'agrupament (clustering): es pot fer una abstracció de totes les dades que simplificaria les operacions en grans volums de dades.

- Basats en models: es disposa de diferents "vistes" de les dades, per grups d'usuaris, de productes, etc. D'aquesta manera, en cada circumstància es pot trobar una recomanació adequada de manera ràpida.

Els recomanadors de productes els fan servir un gran nombre d'empreses, ja que permeten oferir als usuaris altres productes amb una alta probabilitat de que els hi puguin interessar, i poden aconseguir aquesta informació veient quines altres compres han fet usuaris similars. Entre aquestes empreses que utilitzen recomanadors de diferents tipus podem trobar grans empreses americanes com Amazon, AOL o Foursquare.

També hi ha empreses que ofereixen el servei d'afegir recomanadors de productes als clients d'una botiga, com per exemple l'empresa BrainSINS [http://www.brainsins.com/es/].

Actualment existeixen diferents eines o llibreries que permeten gestionar dades per oferir diferents tipus de recomanacions. Entre aquestes sistemes hi ha les següents:

- «Apache Mahout»: http://mahout.apache.org/
- «Crab»: https://github.com/muricoca/crab
- «Lenskit»: http://lenskit.grouplens.org/
- «RecDB»: http://www-users.cs.umn.edu/~sarwat/RecDB/
- «Recommendable»: https://github.com/davidcelis/recommendable
- «Slim»: http://www-users.cs.umn.edu/~xning/slim/html/
- «Weka»: http://www.cs.waikato.ac.nz/ml/weka/

Després de fer un primer anàlisi d'aquestes diferents llibreries, finalment s'ha seleccionat **«Apache Mahout»** com a motor principal d'aquest TFM, ja que disposa de totes les característiques previstes per fer recomanacions de productes de diferents maneres, té una bona documentació del seu sistema, i alhora disposa d'una comunitat molt activa.

1.3. Què és «Apache Mahout»?

«Apache Mahout» és una llibreria desenvolupada en Java, de programari lliure amb llicència Apache License versió 2 [http://www.apache.org/licenses/LICENSE-2.0], i que implementa diferents algoritmes d'aprenentatge automàtic de forma escalable. Aquests algoritmes poden utilitzar-se per classificar dades, agrupar elements per clúster, o implementar un recomanador de productes per filtratge col·laboratiu.

Per començar amb «Apache Mahout» es pot fer amb les llibreries «Taste», que permeten executar els seus algoritmes en una màquina individual sense necessitat d'una infraestructura distribuïda. En el cas de disposar d'un gran volum d'usuaris i productes, les dades es poden moure a una estructura de servidors distribuïts, i aquestes es podrien gestionar mitjançant la llibreria «Apache Hadoop» i el paradigma «MapReduce».

El desenvolupament de «Apache Mahout» es troba en una fase molt activa, i actualment es disposa de la versió 0.9, disponible des de febrer de 2014. La comunitat d'usuaris està molt implicada en el seu desenvolupament, i es pot participar en ella mitjançant llistes de correus (una per usuaris i una altra per desenvolupadors).

El seu codi font en fase de desenvolupament es pot consultar via web pel seu control de versions:

- http://svn.apache.org/viewvc/mahout/

Les incidències obertes també es poden consultar via web:

- https://issues.apache.org/jira/browse/MAHOUT/

Els algoritmes disponibles en «Apache Mahout» poden executar-se en una màquina individual i/o de forma distribuïda mitjançant MapReduce, i són els següents:

Filtratge col·laboratiu

Algoritme	Màquina individual	MapReduce
Filtratge col·laboratiu basat en usuaris	OK	
Filtratge col·laboratiu basat en productes	OK	OK
Matriu de factorització amb alternança de mínims quadrats	OK	OK
Matriu de factorització amb alternança de mínims quadrats amb retroalimentació implícita	OK	OK
Matriu de factorització ponderat	OK	

Classificació

Algoritme	Màquina individual	MapReduce
Bosc aleatori		OK
Models ocults de Markov	OK	
Naive Bayes		OK
Perceptró multicapa	OK	
Regressió logística via SGD	OK	

Agrupaments

Algoritme	Màquina individual	MapReduce
Agrupament Canopy (en desús)	OK	OK
Agrupament espectral		OK
Agrupament k-means	OK	OK
Fuzzy k-means	OK	OK
Streaming k-means	OK	OK

Reducció de dimensions

Algoritme	Màquina individual	MapReduce
Algoritme de Lanczos	OK	OK
Anàlisi de component principal (PCA)	OK	OK
Descomposició en valors singulars (SVD)	OK	
Estocàstic SVD	OK	OK

Altres

Algoritme	Màquina individual	MapReduce
Assignació Dirichlet latent	OK	OK
Mineria de patró freqüent		OK
Similitud de files de matrius		OK
Concatenació de matrius		OK
Col·locacions: trobar paraules en text		OK

2. OBJECTIUS

En aquest TFM es pretén dissenyar un sistema de recomanació de productes que es pugui explotar de forma fàcil i ràpida per diferents tipus de productes, i que estaria format per 3 parts:

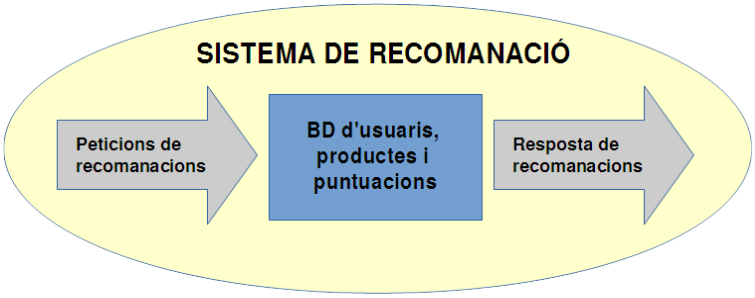
- El sistema de recomanació de productes per usuaris (la part principal del TFM).
- L'entrada de dades a la base de dades d'usuaris, productes i puntuacions.
- La sortida de dades oberta per a ser explotada fàcilment per altres aplicacions.

A continuació es detallen els objectius desitjats per a cadascuna d'aquestes 3 parts.

2.1. Sistema de recomanació de productes

La part del sistema de recomanació de productes serà la base i el punt més important del TFM, i es farà amb la llibreria «Apache Mahout». Aquest sistema ha de calcular les recomanacions per diferents criteris de forma dinàmica, segons convingui, ja siguin per les característiques d'un tipus de producte, o per diferents tipus de productes alhora. D'aquesta manera es podrien fer recomanacions creuades que podrien variar els seus resultats segons els tipus de productes que l'usuari seleccioni per realitzar els càlculs.

El sistema de recomanació rebrà diferents tipus de peticions de recomanacions, i s'encarregarà de donar les respostes als usuaris en el menor temps possible.



Cal tenir la possibilitat de poder integrar dades des de diferents bases de dades d'entrada, ja siguin públiques (opendata) o privades, així d'aquesta manera es podran fer proves de recomanacions amb grans volums de dades.

Aquest sistema de recomanació s'executarà de forma individual en un servidor, però també ha de tenir la possibilitat de poder executar-se per a grans volums de dades de forma distribuïda amb tecnologia «Apache Hadoop».

Per assegurar que el sistema de recomanació funciona correctament, s'ha de realitzar diferents proves de validació utilitzant part de les dades ja existents.

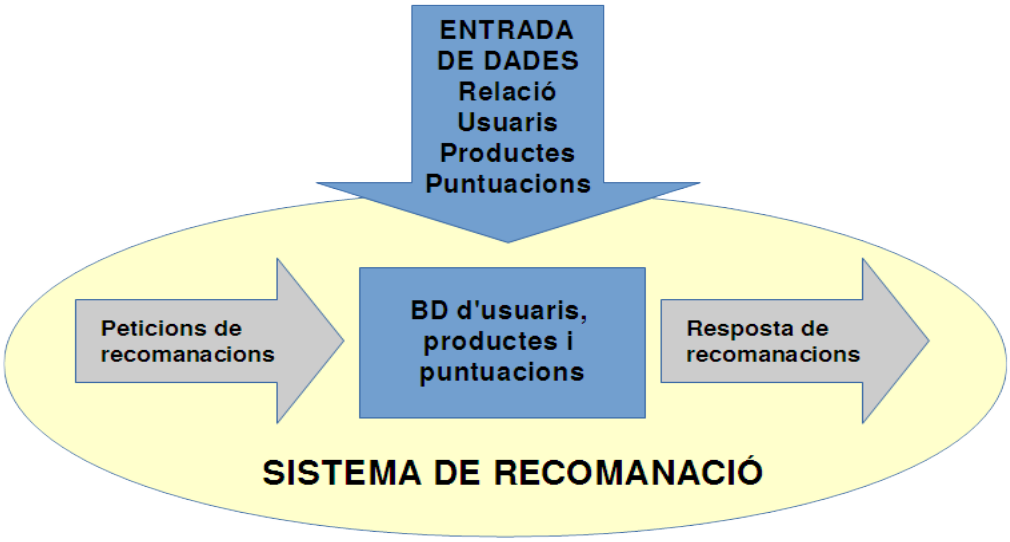
2.2. Entrada de dades

El sistema de recollida de dades ha de permetre emmagatzemar les dades dels usuaris, els productes, els tipus de productes, i les característiques d'aquests productes que puguin ser interessants per a ser valorades pels usuaris.

Alhora, cal que els usuaris puguin valorar les seves preferències per qualsevol producte i tipus de producte registrat, i consultar l'històric de les seves valoracions per veure com evolucionen en el temps. Aquesta informació serà molt valuosa, ja que apart de poder servir pel sistema de recomanació per usuaris, pot servir per generar diferents informes i estadístiques que mostrin les tendències en la valoració de productes.

També cal tenir recursos per poder entrar dades de productes d'altres bases de dades públiques, i les dades registrades han de ser fàcilment exportables al sistema de recomanació.

Per a facilitar la recollida de dades dels usuaris, caldria fer un bon estudi d'usabilitat per a poder aconseguir el màxim d'informació, de manera que els usuaris puguin fer moltes valoracions de productes de forma fàcil i en poc temps.



Aquesta part d'entrada de dades es considera una part complementària del TFM. Tot i així, cal remarcar que l'entrada de peticions de recomanacions al sistema no pertany només a aquesta part, sinó que també correspon a la primera fase i part principal del TFM, el "Sistema de recomanació" comentat anteriorment.

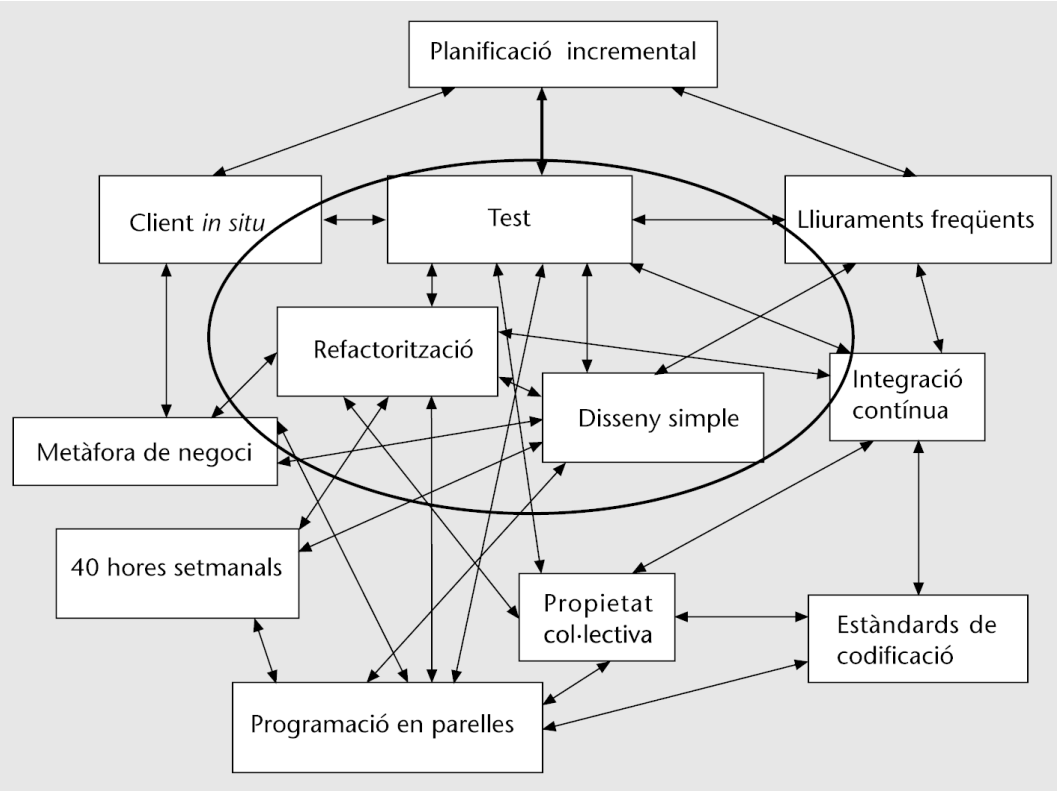
2.3. Sortida de dades

La sortida de dades del sistema de recomanació ha de ser el més oberta possible, de forma que sigui fàcilment explotable per altres aplicacions que hi tinguin accés a les dades, donant la possibilitat d'obtenir simples fitxers CSV, fitxers XML, o altres.

Aquesta part de sortida de dades es considera una part complementària del TFM. Tot i així, cal remarcar que la resposta de peticions de recomanacions al sistema no pertany només a aquesta part, sinó que també correspon a la primera fase i part principal del TFM, el "Sistema de recomanació" comentat anteriorment.

2.4. Metodologia de desenvolupament

Un altre dels objectius marcats pel desenvolupament d'aquest TFM és utilitzar metodologies àgils, amb tècniques d'XP (eXtreme Programming), ja que d'aquesta manera es podran obtenir, en poques setmanes de diferència, noves versions funcionals i una documentació actualitzada durant tot el projecte.



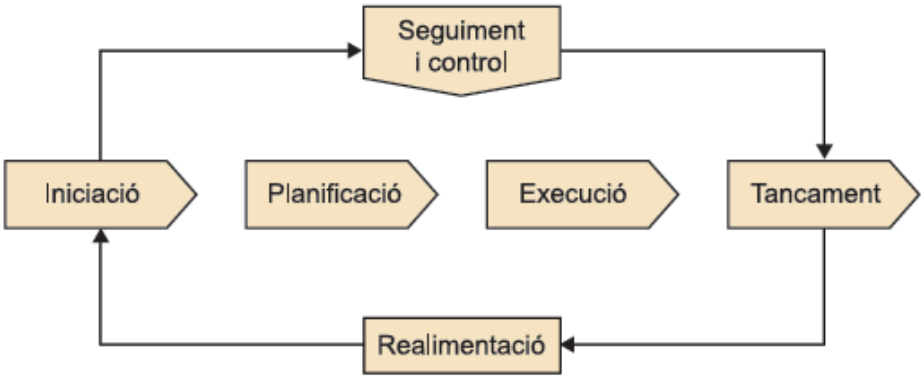
Tenint en compte les 12 pràctiques bàsiques d'XP de la imatge anterior, es donarà rellevància a les tècniques següents:

- Disseny simple: sense codi redundant ni duplicat, i amb el menor nombre possible de classes i mètodes.
- Refactorització: mantenir en el codi final la senzillesa i claredat del codi original.
- Estàndards de codificació: aplicar el mateix criteri en la definició de variables i mètodes.
- Lliuraments freqüents: crear noves versions cada vegada que s'aporti un valor important al projecte.
- Planificació incremental: en funció de l'evolució del treball, es prioritzaran les necessitats que aportin més valor.

Tot i que en aquest cas l'equip de treball tan sols estarà composat per l'alumne i el consultor, s'intentarà treure el màxim profit d'aquesta metodologia pensada pel treball de petits equips de desenvolupament.

2.5. Gestió de projecte

El darrer objectiu d'aquest TFM és realitzar una correcta gestió del projecte en totes les seves fases: Iniciació, Planificació, Execució, Tancament, i Seguiment.



Per realitzar aquesta gestió es farà servir com a guia PMBOK (project management body of knowledge), un estàndard internacional per a la gestió de projectes. Tot i així, en aquest TFM només s'aplicaran els processos que puguin ajustar-se per aquest tipus de projecte, atès que no és un projecte a gran escala i no hi ha un nombre elevat de persones que hi participin.

2.6. Resum d'objectius del TFM

Tots els objectius enumerats en aquest apartat es poden resumir de la següent manera:

Sistema de recomanació de productes
Poder generar recomanacions per diferents criteris, productes o tipus de productes
Rebre peticions de recomanació i donar resposta en poc temps
Capturar les dades d'entrada des de diferents orígens
Funcionar de forma individual en un servidor, o de forma distribuïda per grans volums de dades
Validació dels resultats de les recomanacions a partir de les dades ja existents

Entrada de dades
Importar productes i les seves característiques fàcilment d'altres fonts de dades
Gestionar usuaris que puguin consultar les seves dades
Afegir noves valoracions de productes per usuaris
Exportació de dades al sistema de recomanació

Sortida de dades
Exportar dades en formats oberts, tipus CSV o XML

Altres objectius del TFM
Desenvolupar utilitzant metodologies àgils
Gestionar el projecte utilitzant PMBOK com a guia

3. DESCRIPCIÓ DEL TREBALL

Un cop definits els objectius principals per a cadascuna de les parts del projecte, s'explicarà de forma més detallada com es realitzaran aquests desenvolupaments.

3.1. Descripció del sistema de recomanació de productes

El sistema de recomanació ha de calcular els productes de forma dinàmica, segons els criteris que es vulguin utilitzar per fer les recomanacions. Utilitzant el món de la cervesa com a exemple, un usuari pot donar les seves preferències sobre els fabricants de cerveses, a més de donar les seves preferències sobre cerveses concretes, o fins i tot podria puntuar per tipus de varietat de cervesa segons la seva fermentació (Lager Pilsen, Pale Ale...) o per graduació alcohòlica.

Alhora de executar el procés de recomanació de productes, es podrien utilitzar els criteris només d'una d'aquestes preferències, o de totes elles alhora.

I continuant amb l'exemple de les cerveses, la idea és que es puguin afegir al mateix sistema de recomanació nous tipus de productes i característiques (fabricants de vi, vins en concret, varietat de raïm dels vins...), i que es pogués fer els càlculs de recomanacions per qualsevol dels criteris existents, ja siguin de forma independent, o barrejats segons el criteri de l'usuari.

El sistema de recomanació podrà rebre diferents tipus de peticions de recomanacions. Per exemple, sabent les cerveses que agraden a l'usuari, es podria preguntar el següent:

- Quines cerveses em podrien agradar?
- Quines cerveses no m'agradarien?
- Aquesta cervesa en concret que he seleccionat, m'agradaria o no?
- Entre aquestes cerveses que he seleccionat, quina em recomanaries?

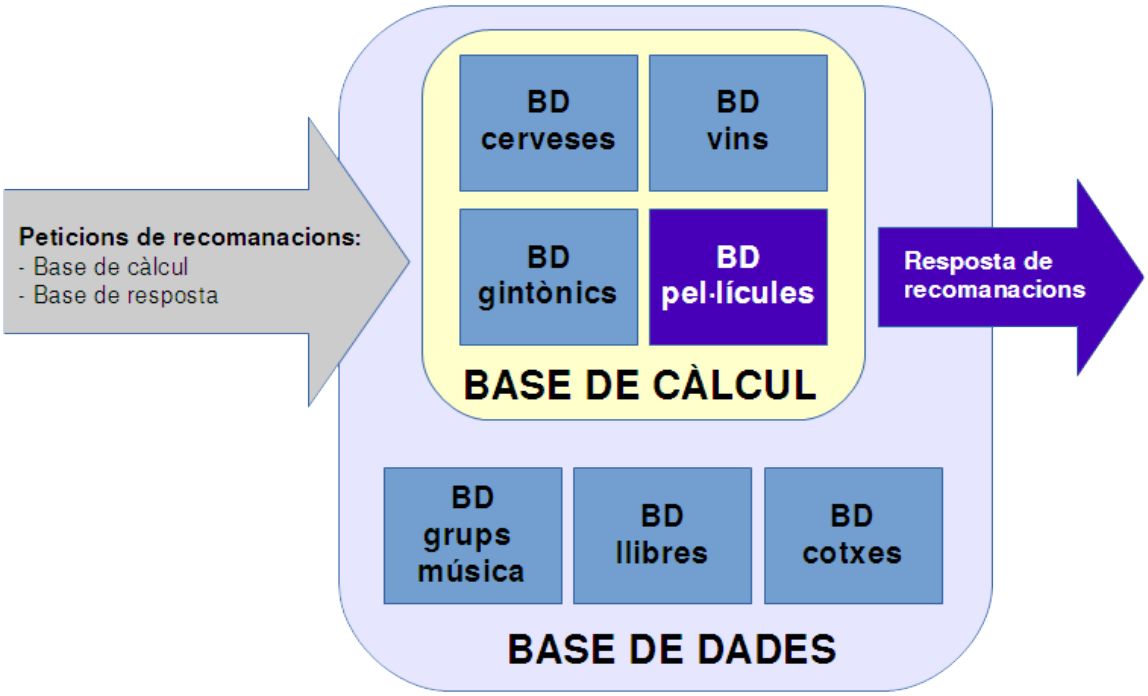
I aquestes mateixes preguntes es podrien fer per altres tipus de productes diferents, però utilitzant la informació dels usuaris respecte a altres productes. Per exemple, un usuari que mai ha provat un Gintònic, però utilitzant les seves preferències en cerveses i vins:

- Quin Gintònic em recomanaries?
- Entre els diferents Gintònics que he seleccionat, quin em recomanaries?

Per fer les peticions de recomanacions amb els filtres que l'usuari consideri, caldrà seleccionar les dades necessàries entre totes les dades disponibles d'usuaris i productes. Els filtres a realitzar poden ser de 2 tipus diferents:

- Base de càlcul: dades d'entrada filtrades amb les que calcular el model de dades
- Base de resposta: dades de sortida que compleixen les característiques desitjades

Per exemple, es pot tenir una base de dades de diferents tipus de productes (cerveses, vins, gintònics, pel·lícules, grups de música, llibres, cotxes, ...), i un usuari que té les seves valoracions de cerveses, vins, i gintònics, vol que se li recomani una pel·lícula basant-se només amb les valoracions d'altres usuaris en cerveses, vins, gintònics i pel·lícules. En la base de resposta es podrien afegir altres filtres, com que sigui una pel·lícula de ciència-ficció, o que sigui una pel·lícula d'estrena.



3.2. Fases del projecte

3.2.1. Sistema de recomanació de productes

El sistema recomanador de productes, basat en «Apache Mahout», estarà desenvolupat en Java, de manera que al ser multiplataforma es podria desenvolupar i executar en qualsevol sistema operatiu. Per preparar l'entorn de desenvolupament es pot utilitzar «Eclipse IDE».

Per fer proves amb el recomanador caldrà tenir dades suficients per a fer les proves. Per facilitar la tasca d'integració de dades, s'haurà de permetre capturar les dades de múltiples fonts, ja siguin altres bases de dades MySQL, o simples fitxers de text amb dades estructurades.

Els càlculs del sistema de recomanació es realitzaran amb la llibreria «Apache Mahout», la qual ofereix una gran escalabilitat per treballar amb petits o grans volums de dades amb filtratge col·laboratiu, agrupaments, i classificacions. Els algoritmes disponibles de filtratge col·laboratiu són els següents:

- Filtratge col·laboratiu basat en usuaris (no disponible en entorns distribuïts)
- Filtratge col·laboratiu basat en productes
- Matriu de factorització amb alternança de mínims quadrats
- Matriu de factorització amb alternança de mínims quadrats amb retroalimentació implícita
- Matriu de factorització ponderat (no disponible en entorns distribuïts)

Es pot començar amb un simple recomanador no distribuït, i en el cas de tenir un gran volum d'usuaris i productes, moure la base de dades a una estructura de servidors distribuïts amb la llibreria «Apache Hadoop» i el paradigma «MapReduce».

Les dades a utilitzar es poden aprofitar de les múltiples bases de dades públiques que existeixen a la xarxa, com per exemple MovieLens [http://grouplens.org/datasets/movielens/]. Amb MovieLens es disposa de diferents jocs de dades de recomanacions aproximadament amb el següent nombre de registres:

1. 100.000 puntuacions de 1.000 usuaris sobre 1.700 pel·lícules
2. 1.000.000 de puntuacions de 6.000 usuaris sobre 4.000 pel·lícules
3. 10.000.000 de puntuacions de 72.000 usuaris sobre 10.000 pel·lícules

Per temes de velocitat, només es farien servir els jocs de dades 1 i 2 en el cas del recomanador d'un servidor individual. Mentre que el joc de dades més gran, el 3, es podria utilitzar en l'entorn de dades distribuït.

Amb aquestes dades de MovieLens es disposa d'un gran volum valoracions de pel·lícules per part de milers d'usuaris. El sistema de recomanació hauria de ser capaç de respondre als seus usuaris a preguntes similars a les plantejades anteriorment:

- Quines pel·lícules em podrien agradar?
- Aquesta pel·lícula en concret que he seleccionat, m'agradaria o no?
- Entre aquestes pel·lícules que he seleccionat, quina em recomanaries?

També es poden aplicar filtres en aquestes dades per poder fer peticions de recomanació més concretes. Per exemple, per demanar recomanacions només d'una categoria concreta.

- Quines pel·lícules de «ciència-ficció» em podrien agradar?

Per assegurar que el sistema de recomanació funciona correctament, s'ha de realitzar diferents proves de validació utilitzant part de les dades ja existents, separant la part d'entrenament i la part de test.

A nivell de Hardware, serà interessant disposar d'un servidor on poder executar les aplicacions necessàries per utilitzar el recomanador. Es poden valorar diferents opcions:

- Un servidor virtual propi amb tot el software necessari per executar el recomanador.
- Un grup de servidors al núvol, amb el software necessari per poder executar el recomanador de forma distribuïda. En aquest cas es podria utilitzar els serveis de Google App Engine, o els serveis de Amazon Web Services, els quals ofereixen preus molt competitius per utilitzar les seves infraestructures d'altes prestacions.

A nivell de Sofware serà necessari disposar de les següents aplicacions:

- Apache Mahout
- Apache Hadoop
- Eclipse IDE
- Java Developement Kit

3.2.2. Entrada de dades

En el sistema de recollida de dades de productes, els usuaris hauran de puntuar les seves preferències de productes de forma fàcil i usable. I alhora aquestes dades han de ser exportades el més ràpid possible al sistema de recomanació.

En l'entrada de dades cal recollir 3 tipus d'informacions diferents:

- Dades d'usuaris
- Dades de productes
- Puntuacions de productes per usuaris

Pel sistema de recomanació només interessen les dades de la taula *Puntuacions de productes per usuaris*, ja que la resta d'informació és irrellevant pel sistema. Tot i així, cal tenir les eines necessàries per poder filtrar aquestes puntuacions i enviar al recomanador de productes només les puntuacions necessàries.

Per aquest sistema s'utilitzarà el gestor de continguts Drupal [https://www.drupal.org], de programari lliure amb llicència GNU GPL v2, que s'executa sobre una base de dades MySQL, i el qual permet una gran parametrització per definir diferents tipus de continguts (productes), i diferents mòduls per gestionar usuaris i les seves valoracions de productes.

També es poden aprofitar altres fonts de dades obertes ja existents a la xarxa per poder importar productes de forma massiva, com per exemple des de Freebase [http://www.freebase.com/].

3.2.3. Sortida de dades

Per a realitzar la sortida de dades del sistema de recomanació, es posaran disponibles diferents formats de dades obertes per a que siguin fàcilment utilitzables per altres aplicacions. Entre els formats interessants hi ha el format CSV, XML i JSON.

Com a complement del projecte, i si el temps ho permetés, podria ser interessant realitzar un estudi de mineria de dades amb privacitat. Donat que disposaríem de milers de dades d'usuaris i les seves preferències, és imprescindible poder garantir la seva privacitat alhora que s'obté un bon rendiment de les dades.

3.3. RISCOS

En un projecte d'aquestes característiques hi ha varis riscos que cal tenir en compte:

- No disposar de les infraestructures suficients per a realitzar el TFM, donat que es requereix un servidor amb certa memòria RAM, i que estigui configurat amb diferents aplicacions de Java per a poder executar Mahout correctament. A més s'afegeix la intenció d'aplicar el recomanador sobre una estructura de servidors distribuïts per poder processar grans volums de dades, i un cost econòmic elevat d'aquesta infraestructura limitaria les proves a realitzar.

- No disposar d'un nombre de dades suficients i de prou qualitat per a realitzar el treball amb les màximes garanties. Un dels objectius es poder crear dades de diferents tipus de productes, i pot ser complicat fer aquest encreuament utilitzant els mateixos usuaris d'altres dades.

- No disposar del temps necessari per desenvolupar el treball, per no poder compaginar-ho amb la feina actual en empresa, i per dedicar part del temps a la família.

3.4. Planificació inicial

La planificació inicial pel TFM s'ha calculat tenint en compte la dedicació de 20 hores setmanals per a realitzar el treball, i el qual te una durada de 15 setmanes (300 hores en total).

Tenint en compte les dates d'entrega de cada PAC i el treball final, s'ha fet una primera planificació setmanal de la feina a fer.

Setmana	Treball
Setmana 1 (24 febrer- 2 març)	Lectura de documents del TFM i cerques d'informació
Setmana 2 (3 març – 9 març)	Cerques d'informació, referències i planificació general
Setmana 3 (10 març – 16 març)	**ENTREGA PAC1**
Setmana 4 (17 març – 23 març)	Pla de treball detallat
Setmana 5 (24 març – 30 març)	Sistema de recomanació simple d'un tipus de producte
Setmana 6 (31 març – 6 abril)	Sistema de recomanació variable amb dades creuades
Setmana 7 (7 abril – 13 abril)	**ENTREGA PAC2**
Setmana 8 (14 abril – 20 abril)	Validació de resultats del recomanador Proves amb dades distribuïdes
Setmana 9 (21 abril – 27 abril)	Sistema d'entrada de dades amb Drupal
Setmana 10 (28 abril – 4 maig)	Sortida de dades obertes del sistema
Setmana 11 (5 maig – 11 maig)	**ENTREGA PAC3**
Setmana 12 (12 maig – 18 maig)	Millores del projecte i nova validació del recomanador
Setmana 13 (19 maig – 25 maig)	Finalització de memòria
Setmana 14 (26 maig – 1 juny)	Preparació de presentació
Setmana 15 (2 juny)	**ENTREGA FINAL**

3.5. Pressupost

Per a realitzar aquest TFM pot ser necessari disposar d'un Hardware que compleixi certes característiques. Els preus del Hardware podrien variar en funció de la mida del projecte. Entre les opcions a revisar tenim els següents tipus de servidors:

Servei	Cost
Servidor virtual per executar el recomanador	100€/any
Servidor virtual en Google App Engine per executar el recomanador	gratuït fins a un límit
Servidor virtual en Amazon Web Services per executar el recomanador	gratuït fins a un límit

A nivell de Software, les aplicacions utilitzades són totes de programari lliure, de manera que no suposen cap cost, i tan sols s'haurà de fer referència a la seva utilització en les llicències del software generat a partir d'elles.

4. PLA DETALLAT DEL PROJECTE

Un cop definits els objectius del TFM i la seva planificació inicial, és el moment de definir un pla detallat de projecte amb les diferents tasques a fer. S'han definit 11 tasques, i aquestes es reparteixen en funció de les dates d'entrega previstes en l'avaluació continuada del TFM:

- La primera versió del TFM coincideix amb l'entrega de la PAC2 (9 d'abril de 2014)
  1. Preparació de l'entorn de desenvolupament
  2. Recomanador d'un tipus de producte concret
  3. Recomanador de diferents tipus de productes

- La segona versió del TFM coincideix amb l'entrega de la PAC3 (7 de maig de 2014)
  4. Validació de resultats del recomanador
  5. Recomanador amb dades distribuïdes
  6. Sistema d'entrada de dades pel recomanador
  7. Sortida de dades oberta per altres aplicacions

- L'entrega final juntament amb la memòria i la presentació (4 de juny de 2014)
  8. Millores generals en totes les tasques
  9. Validació general de resultats
  10. Pròxims passos després del TFM
  11. Memòria i presentació

La planificació del projecte s'ha realitzat tenint en compte un esforç aproximat de 20 hores setmanals, suposant que es dediquen 4 hores diàries durant 5 dies de la setmana. Tot i així, en la realitat aquestes hores es repartiran durant la setmana en funció de la compatibilitat laboral i familiar, algunes setmanes amb la feina concentrada al cap de setmana, i en d'altres repartida en els dies laborables.

També cal destacar que, tot i que no apareixen com a tasques, es reserven 4 dies de feina per a cadascuna de les 2 entregues parcials PAC2 i PAC3, per a poder tancar la documentació a presentar en cadascuna de les entregues.

4.1. Descripció de tasques

En al descripció detallada de les tasques a fer es farà un resum de la feina a realitzar, els coneixements previs que es disposen sobre les eines a utilitzar, els riscos relacionats en cada tasca, i el temps previst d'execució de la tasca.

4.1.1. Preparació de l'entorn de desenvolupament

Descripció
• Muntar l'entorn de desenvolupament del recomanador amb Apache Mahout, el qual està composat per llibreries Java. Caldrà utilitzar eines compatibles amb Java com Eclipse i Maven. • Es podrà fer alguna prova simple de recomanador amb Mahout per comprovar que tot funciona correctament i a una velocitat acceptable. • També es poden fer proves en sistemes operatius diferents, com Windows XP o Ubuntu.

Coneixements previs
• S'han fet alguns petits projectes amb Java i amb Eclipse, tot i que no és el llenguatge de programació ni l'entorn de desenvolupament habitual. • Amb Apache Mahout no s'ha treballat mai.

Riscos
• Tenir una corba d'aprenentatge superior a la prevista. • No disposar del maquina adequat per la preparació de l'entorn i el desenvolupament del TFM.

Temps previst
• 2 dies (8 hores)

4.1.2. Recomanador d'un tipus de producte concret

Descripció
• Començar utilitzant el recomanador de productes per un tipus de producte concret amb les dades públiques de MovieLens [http://grouplens.org/datasets/movielens/]. D'aquesta manera es podran fer proves, de forma ràpida, amb diferents volums de dades reals, i així es podran veure els seus diferents temps de resposta. Les dades disponibles són les següents: ◦ 100.000 puntuacions de 1.000 usuaris sobre 1.700 pel·lícules ◦ 1.000.000 de puntuacions de 6.000 usuaris sobre 4.000 pel·lícules ◦ 10.000.000 de puntuacions de 72.000 usuaris sobre 10.000 pel·lícules • Utilitzant les llibreries estàndards de Mahout, s'hauria de poder respondre a les diferents preguntes que un usuari pugui fer al recomanador utilitzant la mateixa base de càlcul i base de resposta, com per exemple: ◦ Dóna'm un nombre X de productes que recomanaries. ◦ Aquest producte concret me'l recomanaries? Quina valoració li donaria jo? ◦ Entre aquests productes que he seleccionat, quin em recomanaries?

Coneixements previs
• S'ha treballat en diferents projectes per integrar dades d'un sistema a altres. • Amb Apache Mahout no s'ha treballat mai.

Riscos
• Tenir una corba d'aprenentatge superior a la prevista.

Temps previst
• 2 dies (8 hores)

4.1.3. Recomanador de diferents tipus de productes

Descripció

• A partir del mateix exemple de dades de MovieLens es podran obtenir valoracions dels usuaris de diferents maneres:
  ◦ En funció del gènere de les pel·lícules, es podrà demanar al sistema que faci recomanacions de pel·lícules d'un gènere en concret però utilitzant les valoracions dels usuaris en tots els gèneres.
  ◦ En funció d'altres dades dels usuaris i/o pel·lícules, aplicar el filtre de dades abans de calcular el model de dades, per exemple tenint en compte només les valoracions dels usuaris d'edat similar, o tenint en compte només les valoracions d'uns gèneres en concret.
• Per fer aquests filtres caldrà importar les dades de MovieLens en una base de dades relacional, com MySQL, que permeti exportar les dades a calcular de forma ràpida.
• Amb aquestes dades un usuari ja podrà fer noves preguntes més concretes al recomanador. I en aquest cas ja s'utilitzarà una base de càlcul diferent a la base de resposta:
  ◦ Dóna'm un nombre X de productes que recomanaries d'aquest gènere en concret.
  ◦ Tenint en compte només les valoracions dels gèneres 1, 2, i 3, dóna'm un nombre X de productes que recomanaries

Coneixements previs
• S'ha treballat en diferents projectes amb bases de dades relacionals com MySQL. • Amb Apache Mahout no s'ha treballat mai.

Riscos
• Funcionalitats de filtratge no disponibles en Apache Mahout que requereixin un desenvolupament específic.

Temps previst
• 5 dies (20 hores)

4.1.4. Validació de resultats del recomanador

Descripció
• Amb les mateixes dades de MovieLens utilitzades en el desenvolupament del recomanador, caldrà fer diferents proves i validar que els resultats obtinguts per una gran part de les dades s'ajusten a la realitat en comparar-ho amb una altra petita part de les dades. • En aquesta validació es faran diferents proves per veure quin seria el mínim de dades necessàries per a tenir un resultat acceptable.

Coneixements previs
• Ja s'han fet validacions de resultats en altres casos, com en la assignatura de Intel·ligència Artificial Avançada d'aquest mateix Màster.

Riscos
• No obtenir uns resultats adequats que facin tornar a desenvolupar alguna fase anterior.

Temps previst
• 3 dies (12 hores)

4.1.5. Recomanador amb dades distribuïdes

Descripció
• Per utilitzar el recomanador amb dades distribuïdes, mitjançant les llibreries de "Apache Hadoop" i el paradigma «MapReduce», s'ha d'utilitzar algun servei privat que ofereixi aquest entorn, com per exemple Google App Engine, i Amazon Web Services. • Les dades per fer les proves tornaran a ser les de MovieLens, però en aquest cas només es farà servir l'exemple amb un volum de dades més gran: ◦ 10.000.000 de puntuacions de 72.000 usuaris sobre 10.000 pel·lícules

Coneixements previs
• Es tenen alguns coneixements bàsics sobre «Apache Hadoop», però s'hi ha treballat mai. • No s'ha treballat mai amb entorns distribuïts reals de Google App Engine, ni Amazon Web Services.

Riscos
• No obtenir uns resultats adequats que facin tornar a desenvolupar alguna fase anterior.

Temps previst
• 5 dies (20 hores)

4.1.6. Sistema d'entrada de dades pel recomanador

Descripció
• Dissenyar un sistema, amb Drupal, que permeti recollir dades d'usuaris reals i les seves valoracions a productes, poder exportar-les a un recomanador. En aquest sistema caldrà: ◦ Gestionar usuaris. ◦ Importar dades de productes d'altres fonts de dades i gestionar aquests productes. Crear una base de dades de cerveses classificades per diferents característiques i per fabricants. Valorar la captura de dades dels productes a partir d'altres bases de dades públiques, com per exemple Freebase [http://www.freebase.com/]. ◦ Crear enquestes per usuaris per a que puguin valorar els diferents productes i/o característiques. Caldrà decidir quin rang de puntuacions s'han de registrar (de 0 a 10, de 1 a 5, ...). Per aconseguir el màxim d'enquestes d'usuaris s'utilitzarà la pròpia xarxa de contactes d'amics i familiars, i les xarxes socials per animar a tothom a participar en l'estudi, amb l'objectiu és superar el mig miler d'enquestes. ◦ Crear una interfície per a que els usuaris puguin enviar els diferents tipus de peticions de recomanacions al sistema recomanador de productes dissenyat en fases anteriors. Per a un correcte funcionament del recomanador, es valorarà quin serà el nombre mínim de puntuacions que ha de fer un usuari abans no rebi cap recomanació.

Coneixements previs
• Experiència en varis projectes desenvolupats en Drupal i bases de dades MySQL. • No s'ha treballat mai amb bases de dades públiques com Freebase.

Riscos
• Dificultat en obtenir productes rellevants i de qualitat de les bases de dades públiques.

Temps previst
• 5 dies (20 hores)

4.1.7. Sortida de dades oberta per altres aplicacions

Descripció
• Quan ja es disposa d'una base de dades de productes, usuaris, i valoracions, i es poden rebre peticions de recomanacions de diferents usuari, cal retornar les respostes als usuaris de forma que sigui fàcilment exportable i oberta a qualsevol aplicació que ho requereixi, per exemple en formats JSON, XML, CSV o altres. • Com a complement del projecte, i si el temps ho permetés, podria ser interessant realitzar un estudi de mineria de dades amb privacitat.

Coneixements previs
• S'ha treballat amb formats XML o CSV, però no es coneix el format JSON o altres tipus de dades estàndards.

Riscos

Temps previst
• 3 dies (12 hores)

4.1.8. Millores generals en totes les tasques realitzades

Descripció
• Revisar totes les funcionalitats desenvolupades anteriorment per millorar els punts febles trobats durant el desenvolupament. Amb aquest procés de millora contínua es pretén aconseguir una aplicació més fiable i usable. Les tasques anteriors a revisar seran les següents: ◦ Recomanador d'un tipus de producte concret ◦ Recomanador de diferents tipus de productes ◦ Recomanador amb dades distribuïdes ◦ Sistema d'entrada de dades pel recomanador ◦ Sortida de dades oberta per altres aplicacions

Coneixements previs

Riscos
• Qualsevol desviació de temps en fases anteriors pot provocar que aquesta tasca de millora no es pugui desenvolupar, de manera que s'obtindria una aplicació menys fiable.

Temps previst
• 5 dies (20 hores)

4.1.9. Validació general de resultats

Descripció
• Com a fase final del desenvolupament del TFM, caldrà fer una nova validació general dels resultats obtinguts pel recomanador, per a comprovar que els darrers canvis han millorat els resultats, o que almenys no els han empitjorat.

Coneixements previs

Riscos
• Qualsevol desviació de temps en fases anteriors pot provocar que aquesta tasca de millora no es pugui desenvolupar, de manera que s'obtindria una aplicació menys fiable.

Temps previst
• 2 dies (8 hores)

4.1.10. Pròxims passos després del TFM

Descripció

• Un cop finalitzat el desenvolupament del TFM, serà el moment de valorar la feina feta i veure quins serien els següents passos que es podrien realitzar.
  ◦ Noves funcionalitats interessants a desenvolupar
  ◦ Valorar com es podria explotar comercialment una aplicació d'aquest tipus
  ◦ Provar el recomanador distribuït amb «Apache Spark», el qual millora la velocitat en les operacions MapReduce més de 10 vegades respecte Hadoop.
  ◦ Estudiar el impacte en el rendiment en utilitzar tècniques de mineria de dades amb privacitat

Coneixements previs

Riscos

Temps previst
• 2 dies (8 hores)

4.1.11. Memòria i presentació

Descripció
• Després de tota la feina realitzada durant les setmanes del TFM, caldrà fer un últim esforç d'hores per explicar de forma clara i sintetitzada quines han sigut les informacions més rellevants del TFM. • Tot i que durant la realització de les diferents tasques del TFM, així com en cada entrega parcial, ja s'hauran redactat una bona part de les informacions de la memòria, serà important agrupar totes aquestes informacions de forma coherent en l'entrega final.

Coneixements previs

Riscos

Temps previst
• 10 dies (40 hores)

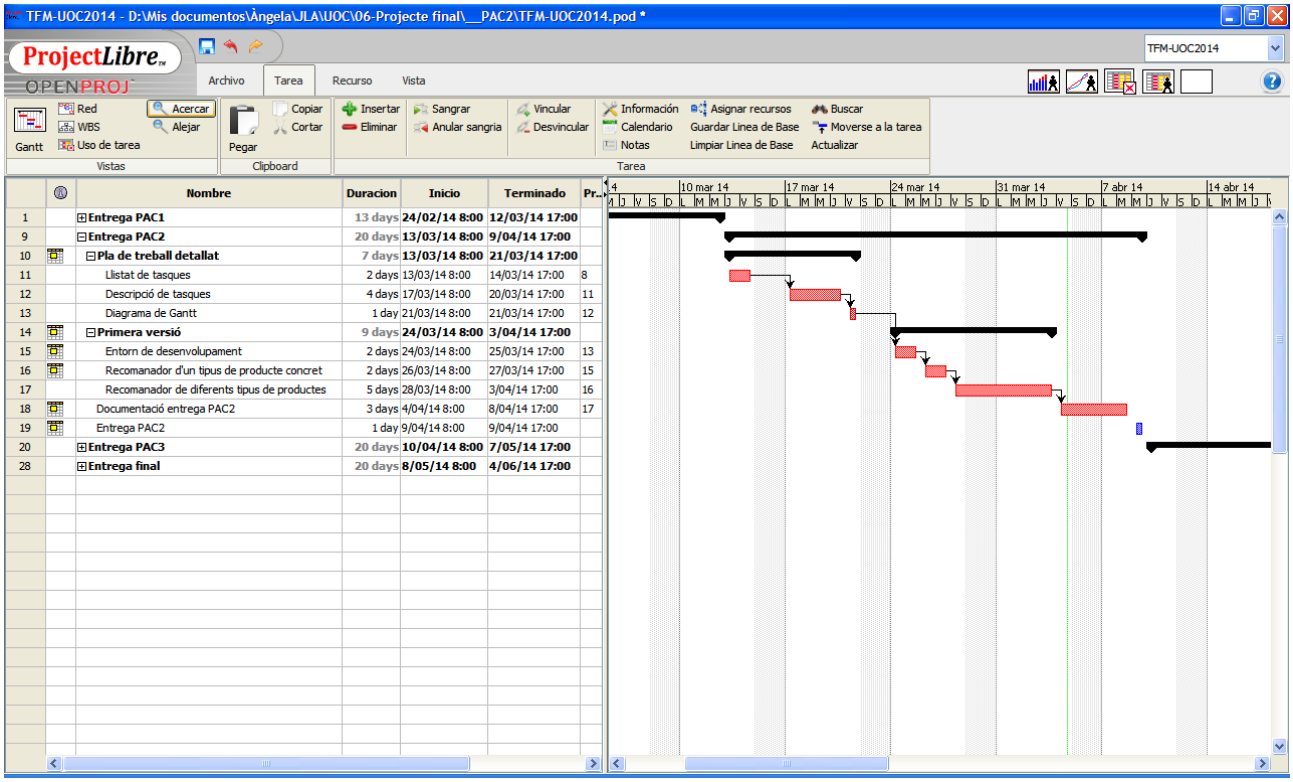
4.2. Diagrama de Gannt

En planificar cadascuna de les tasques i relacionar-les amb les entregues previstes pel TFM, s'obté el següent calendari global del projecte per a cada entrega:

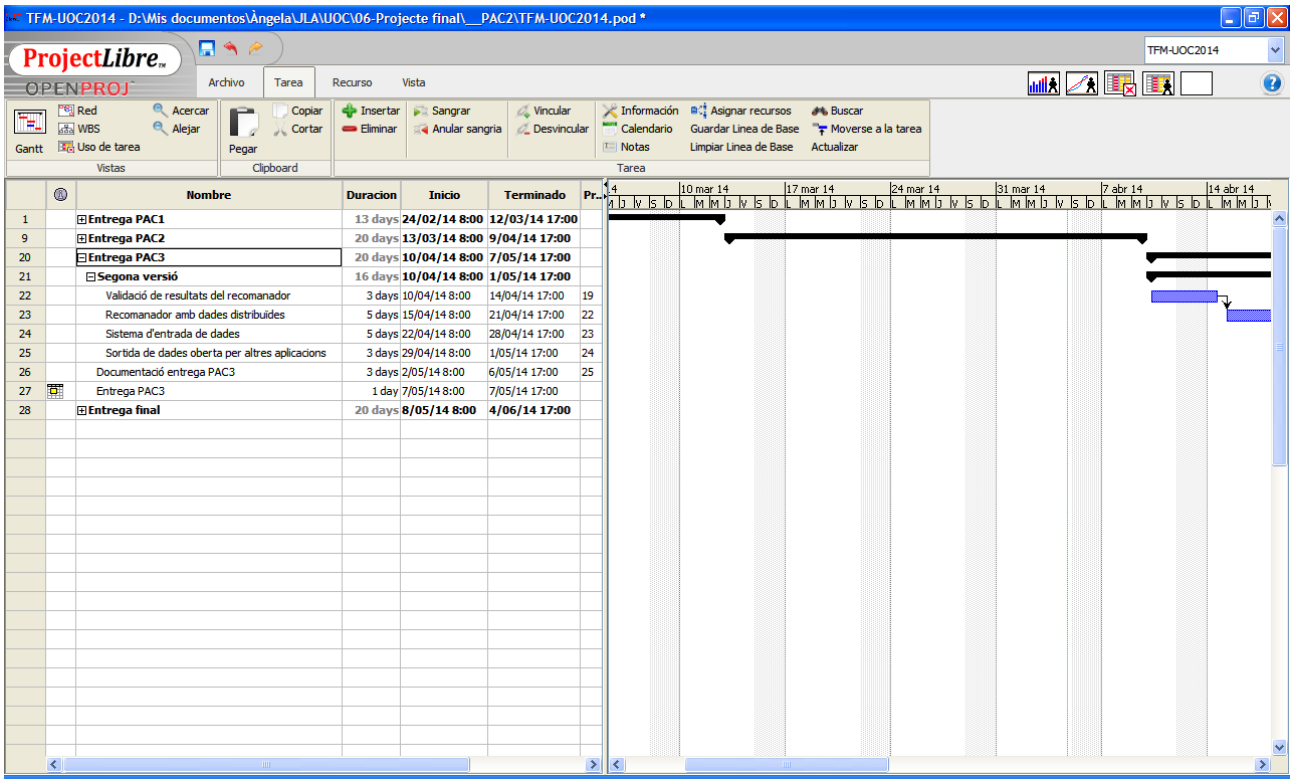
	Nombre	Duracion	Inicio	Terminado
1	Entrega PAC1	13 days	24/02/14 8:00	12/03/14 17:00
9	Entrega PAC2	20 days	13/03/14 8:00	9/04/14 17:00
20	Entrega PAC3	20 days	10/04/14 8:00	7/05/14 17:00
28	Entrega final	20 days	8/05/14 8:00	4/06/14 17:00

Per a l'entrega de la PAC2, es mostren 2 fases diferents, i les seves diferents tasques:

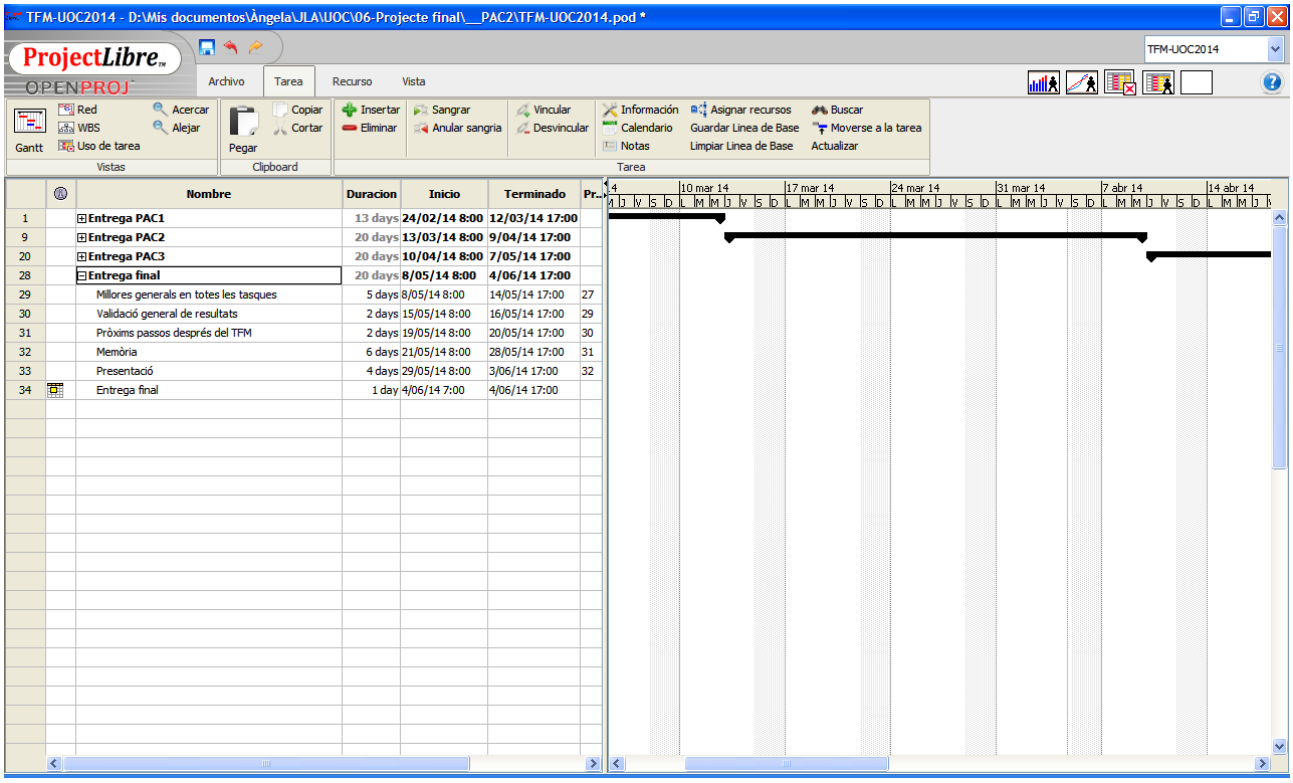
- La planificació del TFM
- I la primera versió d'entrega



Per l'entrega de la PAC3 es mostren els temps previstos per les diferents tasques a desenvolupar.



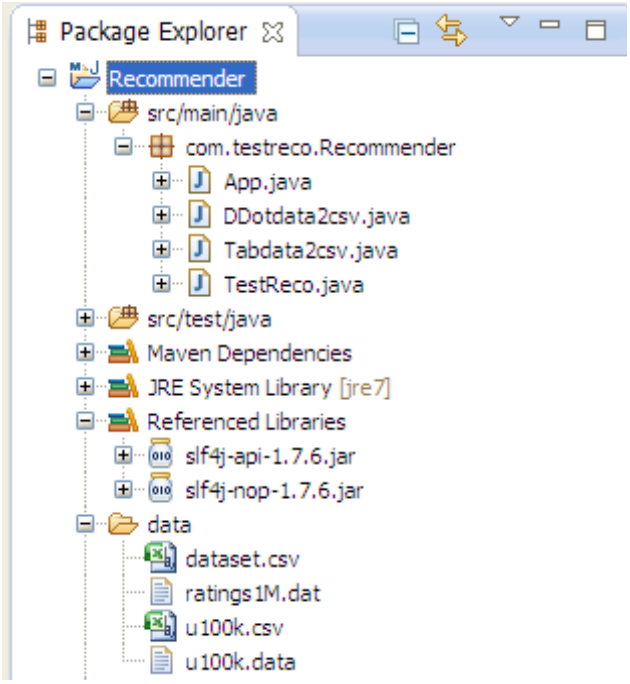
I ja per acabar, per a l'entrega final, es mostren els temps previstos per a les últimes tasques del TFM.



5. DESENVOLUPAMENT DEL PROJECTE

5.1. ENTORN DE DESENVOLUPAMENT

Mahout està composat per llibreries Java, de manera que s'ha utilitzat Eclipse com a IDE de desenvolupament, i s'ha integrat amb Maven per a gestionar el projecte a desenvolupar en Java.



Les versions del programari utilitzat són les següents:

- Apache Mahout 0.9 (versió de febrer de 2014)
- Apache Maven 3.2.1 (versió de febrer de 2014)
- Eclipse 4.3.2 (versió de febrer de 2014)

Per integrar la llibreria Apache Mahout a un projecte amb Maven, tan sols cal afegir la següent dependència al fitxer *pom.xml*:

```
<dependency>
      <groupId>org.apache.mahout</groupId>
      <artifactId>mahout-core</artifactId>
      <version>0.9</version>
</dependency>
```

Un cop muntat tot l'entorn s'han fet algunes proves simples de recomanador amb Mahout per comprovar que tot funciona correctament. Des de les pàgines oficials de Mahout es pot seguir un petit tutorial que permet crear aquest sistema i fer les primeres proves:

[http://mahout.apache.org/users/recommender/userbased-5-minutes.html]

Gràcies a que Java pot executar-se en diferents sistemes operatius, s'ha provat de muntar l'entorn de desenvolupament en 2 sistemes diferents (Windows XP i Ubuntu 12.04), i ambdós amb els mateixos resultats.

5.2. Recomanador d'un tipus de producte concret

Inicialment s'ha creat un recomanador de productes basat en memòria, ja que no s'utilitzen volums de dades molt grans, i basat en usuaris. D'aquesta manera es pretén que un usuari rebi recomanacions de productes que altres usuaris semblants a ell han valorat positivament.

S'han fet proves amb les dades públiques de MovieLens, tant amb les dades que contenen 100.000 registres de puntuacions, com amb les dades de contenen 1.000.000 de registres de puntuacions. Donat que aquestes dades es troben en un simple fitxer de text, el seu temps de processament és molt ràpid.

5.2.1. Transformació de dades originals

Tot i així, s'han fet algunes adaptacions al fitxer de dades de MovieLens per poder ser utilitzat pel recomanador de «Apache Mahout». En les dades originals els camps es troben separats per tabuladors (per la mostra de 100.000 registres) o per dobles dos punts «::» (per la mostra de 1.000.000 registres), mentre que el recomanador necessita les dades en format CSV (separades per comes) i només amb els 3 camps rellevants:

1. id d'usuari
2. id de pel·lícula
3. puntuació

A continuació es mostra un exemple del fitxer original de MovieLens amb 100.000 registres, i el fitxer modificat que podrà llegir el recomanador:

Dades originals Movielens 100.000 registres	Dades adaptades pel recomanador
196 242 3 881250949	196,242,3
186 302 3 891717742	186,302,3
22 377 1 878887116	22,377,1
244 51 2 880606923	244,51,2
166 346 1 886397596	166,346,1

Per a fer aquesta adaptació de forma ràpida s'ha creat una classe en Java per a que sigui capaç de fer aquestes transformacions de forma automàtica, per exemple executant la següent instrucció:

```
RecomFromTab2Csv ''fitxer_origen.data'' ''fitxer_desti.csv''
```

I com en el cas anterior es mostra un exemple del fitxer original de MovieLens amb 1.000.000 de registres, i el fitxer modificat que podrà llegir el recomanador:

Dades originals Movielens 1.000.000 registres	Dades adaptades pel recomanador
1::1193::5::978300760	1,1193,5
1::661::3::978302109	1,661,3
1::914::3::978301968	1,914,3
1::3408::4::978300275	1,3408,4
1::2355::5::978824291	1,2355,5

També s'ha creat una classe en Java per a que sigui capaç de fer aquestes transformacions de forma automàtica, per exemple executant la següent instrucció:

```
RecomFromDDot2Csv ''fitxer_origen.data'' ''fitxer_desti.csv''
```

5.2.2. Preparació del model de dades

Per poder fer les recomanacions s'ha de preparar el model de dades on fer els càlculs:

1. Crear un model de dades a partir dels fitxers CSV amb totes les dades de puntuacions de MovieLens. En aquest cas la base de càlcul són totes les dades de MovieLens. També es poden capturar les dades d'altres orígens de dades JDBC (per exemple amb JDBCDataModel), però el més ràpid serà recollir-ho d'un simple fitxer de text tipus CSV.

```
DataModel model = new FileDataModel(new File(vsFitxerDades));
```

2. A continuació es poden calcular les similituds entre usuaris en funció de la valoració que han fet als productes. Aquest càlcul es pot fer amb diferents algorismes, com per exemple:
   - Correlació de Pearson:

```
similarity = new PearsonCorrelationSimilarity(model);
```

   - Distància Euclidiana:

```
similarity = new EuclideanDistanceSimilarity(model);
```

   - Distància Manhattan (o CityBlock):

```
similarity = new CityBlockSimilarity(model);
```

3. I finalment, calcular quin grup d'usuaris seran els veïns de cada usuari. Aquest càlcul es pot fer a partir d'un llindar mínim de similitud, o per un nombre N de veïns propers, o per tots dos càlculs alhora.
   - Llindar mínim de similitud:

```
neighborhood = new ThresholdUserNeighborhood(llindar,similarity, model);
```

   - Els N veïns més propers:

```
neighborhood = new NearestNUserNeighborhood(N,similarity, model);
```

   - Els N veïns més propers dins del llindar mínim:

```
neighborhood = new NearestNUserNeighborhood(N,llindar,similarity, model);
```

5.2.3. Consulta de recomanacions

Amb aquestes dades ja es poden generar consultes al recomanador de «Apache Mahout», i es retornarà resultats diferents segons la configuració que s'hagi triat. Al no aplicar cap filtre de resposta, la base de resposta és la mateixa que la base de càlcul. Per exemple:

- L'usuari número 42 demana que es recomanin 5 productes
  - Funció per executar sobre el model de dades:

```
recommender(42, 5);
```

  - Segons la configuració, es poden obtenir resultats diferents en l'execució, mostrant els 5 identificadors de producte recomanats i la predicció de la seva valoració.
    - Configuració 1 (correlació de Pearson, amb llindar 0.1):

```
RecommendedItem[item:1643, value:5.0]
RecommendedItem[item:1189, value:5.0]
RecommendedItem[item:1431, value:4.742226]
RecommendedItem[item:1154, value:4.5770674]
RecommendedItem[item:868, value:4.5692]
```

    - Configuració 2 (correlació de Pearson, amb llindar 0.4):

```
RecommendedItem[item:707, value:5.0]
RecommendedItem[item:502, value:5.0]
RecommendedItem[item:320, value:5.0]
RecommendedItem[item:155, value:5.0]
RecommendedItem[item:251, value:5.0]
```

    - Configuració 3 (distància Euclidiana, amb els 300 veïns més propers):

```
RecommendedItem[item:1159, value:5.0]
RecommendedItem[item:1154, value:5.0]
RecommendedItem[item:1143, value:5.0]
RecommendedItem[item:320, value:5.0]
RecommendedItem[item:611, value:4.863957]
```

- L'usuari número 42 demana quina valoració li donaria al producte amb id 100:
  - Funció per executar sobre el model de dades:

```
estimatePreference(42, 100);
```

  - Resultat de l'execució amb la predicció de la valoració:

```
4.092262
```

- L'usuari número 42 demana quina valoració quin producte se li recomanaria entre els productes 100, 122, 425, i 512. Utilitzant la mateixa funció anterior, amb la valoració més alta s'obtindria el resultat:
  - Configuració 1 (correlació de Pearson, amb llindar 0.1):

```
4.092262
2.2011065
3.5492985
3.7558765
```

  - Configuració 2 (correlació de Pearson, amb llindar 0.4):

```
4.01747
1.1854162
2.5640533
NaN
```

  - Configuració 3 (distància Euclidiana, amb els 300 veïns més propers):

```
4.089667
2.174432
3.6447628
4.8127937
```

Segons la configuració del model de dades es poden obtenir resultats diferents, i en aquest mateix exemple, dues d'aquestes configuracions recomanarien el producte 100, mentre que una altra recomanaria el producte 512.

5.3. Recomanador de diferents tipus de productes

Com s'ha vist fins ara, en les dades utilitzades pel recomanador tan sols es necessiten 3 camps (id d'usuari, id de pel·lícula i puntuació).

Per a poder executar el recomanador filtrant per algun altre paràmetre cal tenir més dades dels usuaris i/o productes, i que aquestes dades estiguin disponibles en una base de dades relacional per a poder fer consultes i filtres de forma més fàcil.

En una base de dades MySQL s'han importat les dades de usuaris, pel·lícules, i valoracions en diferents taules. A partir d'aquí s'han aplicat els filtres a realitzar en el càlcul de recomanacions de 2 maneres diferents:

- En primer lloc, filtrant les dades que ha d'incloure el model de dades (base de càlcul).
  - Si s'ha de tenir en compte el càlcul de tot el model de dades, aquest serà el mateix fitxer CSV que l'utilitzat en la tasca anterior.
  - Si s'ha de tenir en compte només les valoracions d'usuaris d'edats similars, o només les valoracions de les pel·lícules d'un gènere concret, cal realitzar la consulta SQL corresponent i exportar les dades de les valoracions resultats en un fitxer de tipus CSV compatible amb el recomanador de Apache Mahout (en format id usuari, id producte i puntuació), i calcular el seu model de dades.
  - Cal tenir en compte que cada vegada que es canvia el filtre, s'ha d'exportar de nou les dades relacionades i calcular de nou el model de dades pel recomanador.

- I en segon lloc, tot i tenir el model de dades filtrat o no, demanant recomanacions en tipus de productes concrets (base de resposta).
  - En aquest cas cal tenir en compte que no és necessari recalcular el model de dades generat anteriorment.
  - Els identificadors dels productes rellevants, dels quals s'ha de calcular quin s'ha de recomanar, s'exporten en un fitxer de tipus CSV per a ser tractat en «Apache Mahout» amb la funció *idrescorer*. Aquest filtre, que es pot programar de la manera que es desitgi, també pot servir per aplicar un % d'augment en el càlcul de la valoració d'un producte que és nou, o que interessa que aparegui més ben valorat per algun motiu concret.

Continuant amb l'exemple anterior de l'usuari número 42, es demana que es recomanin 5 pel·lícules del tipus de gènere 9.

- Funció per executar sobre tot el model de dades (base de càlcul), i filtrant pels identificadors de pel·lícules indicades en el paràmetre *idrescorer* (base de resposta):

```
recommender(42, 5, idrescorer);
```

- Els resultats de l'execució, segons la configuració són els següents:
  - Configuració 1 (correlació de Pearson, amb llindar 0.1):

```
RecommendedItem[item:472, value:3.2033129]
RecommendedItem[item:558, value:3.0404155]
RecommendedItem[item:951, value:3.0284185]
RecommendedItem[item:1133, value:2.8818817]
RecommendedItem[item:560, value:2.8188598]
```

  - Configuració 2 (correlació de Pearson, amb llindar 0.4):

```
RecommendedItem[item:472, value:3.3179512]
RecommendedItem[item:951, value:3.3016078]
RecommendedItem[item:1133, value:3.0]
RecommendedItem[item:560, value:3.0]
RecommendedItem[item:820, value:2.4691901]
```

  - Configuració 3 (distància Euclidiana, amb els 300 veïns més propers):

```
RecommendedItem[item:472, value:3.4988563]
RecommendedItem[item:1615, value:3.4496505]
RecommendedItem[item:1133, value:3.3007448]
RecommendedItem[item:560, value:3.2799623]
RecommendedItem[item:558, value:3.2187672]
```

En aquest cas les valoracions dels productes són molt més baixes, ja que el nombre de pel·lícules a triar ja no es gaire elevat, i augmenten les possibilitats de que no puguin agradar a l'usuari. El que sí trobem en aquest cas és que les 3 configuracions diferents coincideixen en el primer producte a recomanar.

5.4. VALIDACIÓ DE RESULTATS DEL RECOMANADOR

Com s'ha pogut comprovar en les primeres proves, hi ha múltiples combinacions de configuració pel recomanador que donen resultats diferents.

Per validar els resultats d'aquests càlculs del recomanador s'ha utilitzat la mateixa llibreria «Apache Mahout», ja que disposa d'una funció que permet avaluar un model de dades dividint un percentatge de dades d'entrenament, i la resta de dades de test (*trainPercent*). Aquesta selecció es fa de forma aleatòria, de manera que en repetir la mateixa prova es poden obtenir resultats diferents. També es pot indicar quin percentatge de dades es vol avaluar (*evalPercent*), o tot el model de dades, o només les dades d'un percentatge d'usuaris que també se seleccionarien de forma aleatòria.

```
evaluator.evaluate(recomanador, null, model, trainPercent, evalPercent);
```

Amb «Apache Mahout» es poden aplicar diferents algorismes de càlcul sobre el model de dades que es disposa, ja sigui per calcular la similitud entre usuaris, o per calcular els veïns més propers de cada usuari. Per veure més detalls d'aquests diferents algoritmes es pot consultar l'**Annex 1** d'aquest TFM.

Per avaluar els diferents algorismes, s'ha automatitzat aquestes proves per veure els resultats de cada algorisme amb diferents paràmetres. Per facilitar la seva recollida de dades, els seus resultats s'exporten de forma automàtica en un fitxer CSV. I a partir d'aquest fitxer es podran extreure gràfiques i diferents estadístiques per valorar quina seria la millor configuració pel model de dades avaluat. A continuació es mostra un exemple del fitxer resultant d'una avaluació.

```
Pearson;Threshold;0.4;0.8041562420950923;4,145 min
Pearson;Threshold;0.4;0.8038634541569608;4,105 min
Pearson;Threshold;0.4;0.8121097792254234;4,124 min
Pearson;NearestN;500-0.2min;0.7964109339066519;4,632 min
Pearson;NearestN;500-0.2min;0.7894135740416373;4,633 min
Pearson;NearestN;500-0.2min;0.7909649626556554;4,692 min
Euclidean;Threshold;0.4;0.7669897747524815;4,942 min
Euclidean;Threshold;0.4;0.7692763007068124;4,919 min
Euclidean;Threshold;0.4;0.7663852472123697;4,955 min
Euclidean;NearestN;500-0.2min;0.7603728125086958;4,887 min
Euclidean;NearestN;500-0.2min;0.7640133338117746;4,921 min
Euclidean;NearestN;500-0.2min;0.7679003953067033;4,902 min
```

El significat de cada columna és el següent:

1. Indica l'algorisme de similitud utilitzat
2. Indica l'algorisme de veïnatge utilitzat
3. Indica els paràmetres utilitzats en el càlcul de veïnatge
4. Mostra el resultat de l'avaluació
5. Mostra el temps que ha trigat en fer els càlculs

5.4.1. Validació 1: 100.000 registres

En primer lloc s'ha avaluat el recomanador amb les dades de MoviLens que consten de 100.000 registres de puntuacions, entre 943 usuaris i 1.682 pel·lícules. Cada usuari ha puntuat almenys 20 pel·lícules, i els valors de les puntuacions d'aquestes pel·lícules van del 1 al 5 (mínima i màxima puntuació respectivament).

L'avaluació d'aquestes dades s'ha fet utilitzant un 90% de les dades d'entrenament, i el 10% restant com a test. Donat que la partició de cada grup de dades es fa de forma aleatòria, les proves de cada configuració s'han executat com a mínim 3 vegades per poder fer una mitjana del resultat de la seva configuració.

CÀLCULS VEINS	SIMILITUD	PEARSON RESULTAT	PEARSON UNCENTERED RESULTAT	EUCLIDIANA RESULTAT	LOGLIKELIHOOD RESULTAT	CITY BLOCK RESULTAT	SPEARMAN RESULTAT
Threshold	0,01	**0,795**					
Threshold	0,02	**0,794**	0,807	0,809			
Threshold	0,10	**0,794**	0,812	0,803	0,811	NaN	
Threshold	0,20	**0,794**					
Threshold	0,30	**0,794**	0,816	0,806	0,812	NaN	0,795
Threshold	0,50	0,819					
Threshold	0,80	0,902					
NearestN	5,00	0,955					
NearestN	15,00	0,909					
NearestN	25,00	0,895					
NearestN	30,00	0,891	0,882	0,819			
NearestN	35,00	0,889					
NearestN	45,00	0,868					
NearestN	100,00	0,829	0,829	**0,764**	0,803	0,833	
NearestN	150,00	0,802					
NearestN	200,00	0,801					
NearestN	300,00	0,799	0,797	**0,754**	0,801	0,819	
NearestN	**300 (0.3)**	**0,792**	**0,787**	**0,748**			**0,801**
NearestN	400,00	0,802					
NearestN	500,00	0,802	0,797	**0,767**			
NearestN	1000,00	0,826					

Per interpretar els resultats obtinguts en l'avaluació, els valors més petits són els que indiquen un millor resultat, on el valor 0 seria el test perfecte.

A mesura que s'han anat executant proves, s'ha comprovat que amb el càlcul de veïns per un llindar mínim, els millors resultats s'obtenien amb la similitud calculada amb la correlació de Pearson. Tot i així, els resultats han sigut molt similars amb la resta de càlculs de similitud.

D'altra banda, amb el càlcul de veïns pels N més propers, s'han obtingut uns resultats molt semblants, excepte amb la similitud calculada amb la distància Euclidiana, en la que es millorava lleugerament els resultats.

Un cop donats aquests resultats, s'han fet més proves per ajustar el càlcul de veïns pels N més propers, de manera que al afegir un mínim de similitud de 0.3 s'aconseguia ajustar una mica més aquest resultat. D'aquesta manera, si entre els N veïns més propers n'hi havia algun de molt allunyat, ja no es tindria en compte en el seu veïnatge.

5.4.2. Validació 2: 1.000.000 registres

Després d'aquests primers resultats, s'ha fet una nova avaluació amb les dades de MoviLens que consten de 1.000.209 de registres de puntuacions, entre 6.040 usuaris i 3.900 pel·lícules. I com en les dades anteriors, cada usuari ha puntuat almenys 20 pel·lícules amb valors del 1 al 5.

En executar l'avaluació de les proves amb tots els registres, el temps de càlcul era molt elevat i complicava l'estudi d'aquestes dades, així que per agilitzar les proves, s'ha optat per avaluar només el 20% del usuaris (aproximadament uns 200.000 registres) triats de forma aleatòria.

Pel que fa al percentatge de dades d'entrenament s'ha fet la mateixa selecció que en el cas anterior amb un 90% d'entrenament, i només un 10% de test. Aquestes proves també s'han fet com a mínim 3 vegades per a cada configuració.

CÀLCULS VEINS	SIMILITUD	PEARSON RESULTAT	PEARSON UNCENTERED RESULTAT	EUCLIDIANA RESULTAT	LOGLIKELIHOOD RESULTAT
Threshold	0,01	0,789			
Threshold	0,10	**0,773**		0,779	0,789
Threshold	0,20	**0,773**	0,798	0,781	0,788
Threshold	0,30	**0,769**		0,775	0,776
Threshold	0,40	0,788	0,781	**0,746**	0,793
Threshold	0,45	0,788		**0,729**	
Threshold	0,50	0,796		**0,714**	0,784
Threshold	0,55	0,813		**0,726**	
Threshold	0,60	0,812		**0,751**	0,795
Threshold	0,70	0,841	0,781	0,812	0,783
NearestN	100 (0,5)	0,819		0,744	
NearestN	200 (0,4)	0,785		0,739	0,777
NearestN	300 (0,1)	0,777		0,744	0,783
NearestN	300 (0,5)	0,801		**0,723**	
NearestN	400 (0,1)	0,773	0,773	**0,726**	0,775
NearestN	400 (0,4)	0,781	0,777	**0,729**	0,781
NearestN	500 (0,3)	0,777		0,741	0,784
NearestN	500 (0,5)	0,793		**0,719**	0,791
NearestN	600 (0,3)	0,776	0,781	0,742	
NearestN	800 (0,3)	0,778		0,754	
NearestN	900 (0,5)	0,803		**0,713**	

En aquestes proves, en general en tots els algorismes,, s'aconsegueixen millors resultats que en la primera validació. En aquest cas, al tenir un nombre de dades més elevat, es poden aconseguir millors resultats, tot i que el temps de càlcul també ha augmentat considerablement.

Amb el càlcul de distància Euclidiana s'aconsegueixen els millors resultats, ja sigui amb els N primers, o amb un valor mínim de llindar.

5.5. Recomanador amb dades distribuïdes

Un recomanador de productes amb dades distribuïdes requereix muntar una infraestructura específica. Per fer aquesta tasca és necessari diferent maquinari i programari que permeti aquesta tasca.

En primer lloc, gràcies als serveis Amazon Web Services (AWS), es pot disposar de diferents serveis de forma gratuïta en el paquet «AWS Free Tier»:

- http://aws.amazon.com/es/free/

Amb la opció AWS Elastic Cloud Computing (EC2) es poden crear diferents servidors virtuals distribuïts, els quals es poden utilitzar gratuïtament durant 1 any, i fins a 750h mensuals de computació. Si se supera l'espai necessari i les hores de computació, aquestes s'aplicaran les tarifes del servei IaaS de Amazon. Per exemple, per afegir 4 noves CPUs en maquinari Linux, hi hauria un preu de 0.308$/hora.

- http://aws.amazon.com/es/ec2/pricing/

Per crear la infraestructura distribuïda amb Amazon EC2, cal instal·lar diferents aplicacions:

- **Apache Hadoop:** Permet executar operacions de tipus MapReduce de forma distribuïda en diferents servidors alhora.

- **Apache Whirr:** Permet configurar noves instàncies de servidors virtuals de forma fàcil i ràpida per línia de comandes, per utilitzar amb «Apache Hadoop» i utilitzant la llibreria de serveis al núvol «*jclouds*» [http://jclouds.apache.org/].

En aquesta instal·lació i configuració han aparegut alguns problemes que es detallen a continuació, ja que les actuals versions del programari utilitzat no eren compatibles entre elles, de manera que s'ha tingut que buscar les versions que fossin compatibles.

A continuació es mostra una taula amb les versions utilitzades en comparació amb les versions del programari actual.

Versió utilitzada	Última versió disponible
Apache Whirr 0.82 (abril de 2013)	Apache Whirr 0.82 (abril de 2013)
Apache Hadoop 1.2.1 (juliol de 2013)	ApacheHadoop 2.2.0 (octubre de 2013)
Java Developement Kit 1.7.0_45 (octubre de 2013)	Java Developement Kit 1.7.0_55 (abril de 2014)

Els servidors virtuals d'Amazon venen instal·lats amb l'última versió de Java, però aquesta és incompatible amb la versió 0.82 de «Apache Whirr». Després de fer alguns intents de solucionar-ho (https://issues.apache.org/jira/browse/WHIRR-757?jql=project%20%3D%20WHIRR), finalment s'ha optat per utilitzar una versió de Java anterior que fos compatible amb «Apache Whirr», concretament la versió Java JDK 1.7.0_45 (octubre de 2013).

D'altra banda, la versió 0.82 de «Apache Whirr» tampoc és compatible amb la darrera versió estable de «Apache Hadoop» 2.2.0, de manera que també s'ha canviat per utilitzar en el seu lloc la darrera versió estable de la seva branca 1, concretament la versió 1.2.1.

Per veure els detalls d'aquesta instal·lació i configuració es pot consultar l'**Annex 2** d'aquest TFM.

5.5.1. Execució del recomanador distribuït

Per executar el recomanador de forma distribuïda no es poden utilitzar els mateixos algoritmes que en una màquina individual. El filtratge col·laboratiu basat en usuaris utilitzat en les proves anteriors no està disponible amb «MapReduce», així que en aquestes proves s'utilitzaran altres algoritmes.

Filtratge col·laboratiu

Algoritme	Màquina individual	MapReduce
Filtratge col·laboratiu basat en usuaris	**OK**	
Filtratge col·laboratiu basat en productes	**OK**	**OK**
Matriu de factorització amb alternança de mínims quadrats	**OK**	**OK**
Matriu de factorització amb alternança de mínims quadrats amb retroalimentació implícita	**OK**	**OK**
Matriu de factorització ponderat	OK	

Per instal·lar el recomanador distribuït amb «Apache Mahout» cal fer-ho de la següent manera amb la darrera versió estable 0.9:

```
~$ tar xvzf mahout-distribution-0.9.tar.gz
```

S'afegeix el fitxer «*u.data*» amb les dades de MovieLens al sistema de fitxers de Hadoop «HDFS»:

```
~$ hadoop fs -put u.data u.data
```

I ja directament es pot calcular la similitud de tots els registres de les dades d'entrada «*u.data*» que es desaran en el fitxer de sortida indicat, en aquest cas «*output100k*»:

```
~$ mahout-distribution-0.9/bin/mahout recommenditembased -s SIMILARITY_
PEARSON_CORRELATION -i u.data -o output100k --numRecommendations 10
```

Per poder veure els resultats del càlcul, es copien les dades del «HDFS» al propi sistema de fitxers:

```
~$ hadoop fs -getmerge output100k output100k.txt
```

En comprovar els resultats, es revisa els primers 10 valors obtinguts per l'usuari 42, el mateix que s'ha revisat en les proves del recomanador en una màquina individual. Els resultats indiquen en primer lloc el número de producte, i en segon lloc la seva valoració estimada:

```
[     655:5.0,
      869:5.0,
      100:5.0,
      69:5.0,
      449:5.0,
      495:5.0,
      474:5.0,
      838:5.0,
      853:5.0,
      1487:5.0
]
```

Si ho comparem amb els resultats del recomanador en una màquina individual, es pot comprovar que no hi ha cap registre que coincideixi, ja que al tenir molts productes amb la màxima valoració no coincideixen els mateixos resultats en la prova. Per corregir-ho s'hauria d'ajustar més el càlcul per obtenir millors resultats.

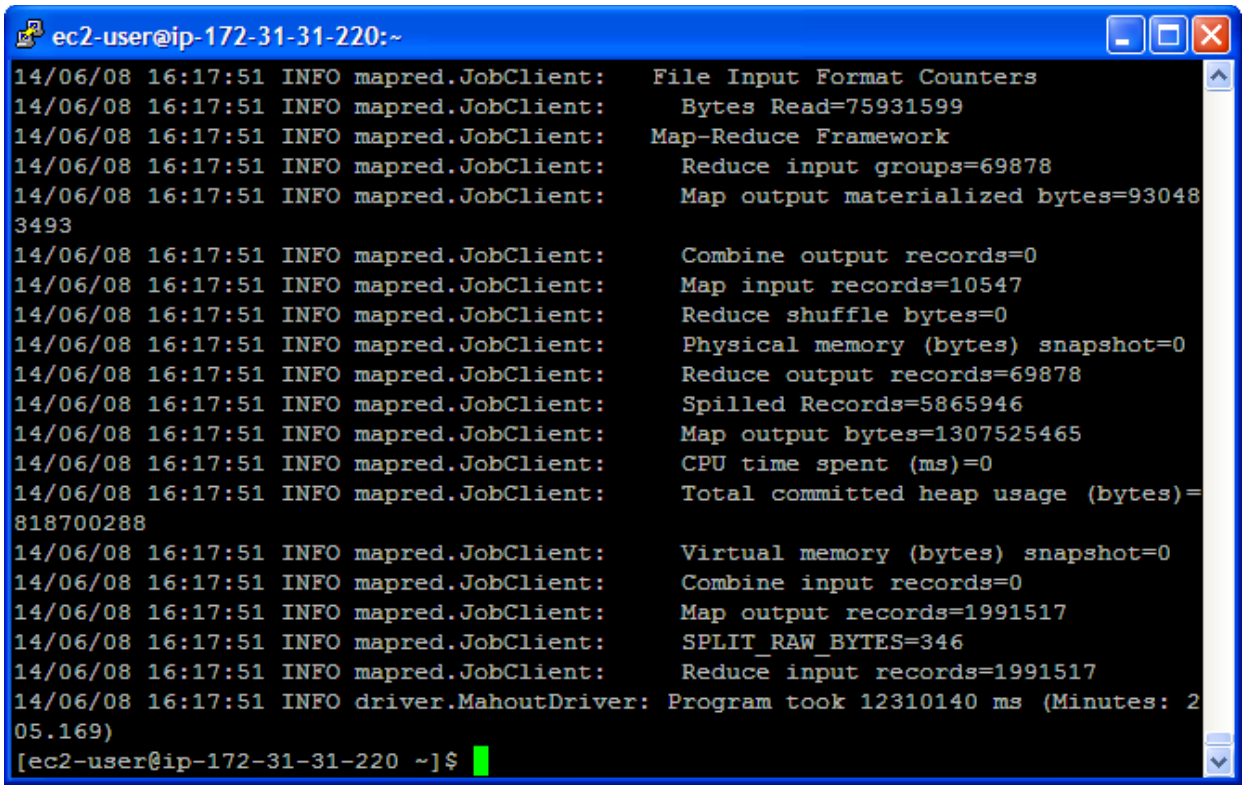
La diferència d'aquesta prova és que s'ha executat el recomanador basat en productes, enlloc del recomanador basat en usuaris que no està disponible en Hadoop. El càlcul de similitud s'ha utilitzat la correlació de Pearson, igual com en la primera prova del recomanador individual. Els diferents càlculs de similitud que es poden utilitzar són els següents:

- SIMILARITY_CITY_BLOCK
- SIMILARITY_COOCCURRENCE
- SIMILARITY_COSINE
- SIMILARITY_EUCLIDEAN_DISTANCE
- SIMILARITY_LOGLIKELIHOOD
- SIMILARITY_PEARSON_CORRELATION
- SIMILARITY_TANIMOTO_COEFFICIENT

En aquesta tasca s'ha calculat directament les recomanacions de tots els usuaris alhora, de manera que en fer una petició de recomanació tan sols s'hauria de consultar el fitxer amb les dades distribuïdes en la infraestructura Hadoop.

En executar la tasca amb diferents volums de dades es veuen temps de resposta molt diferents en funció del nombre de registres.

- 100.000 registres: el càlcul triga al voltant d'1 i 2 minuts.
- 1.000.000 registres: el càlcul triga al voltant d'uns 18 i 20 minuts
- 10.000.000 registres: el càlcul triga al voltant d'uns 200 minuts:



5.5.2. Avaluació del recomanador distribuït

Actualment en Apache Mahout no es disposa de cap eina per avaluar el recomanador calculat amb Hadoop.

Tot i així, es podria dissenyar un petit històric de recomanacions, de manera que quan un usuari valora un producte que anteriorment s'havia recomanat, es podrà comprovar quina diferència hi havia entre la recomanació i la valoració real.

5.6. Sistema d'entrada de dades pel recomanador

En primer lloc, abans de dissenyar l'entrada de dades amb Drupal, s'ha creat una simple base de dades amb aproximadament 300 cerveses. La gran majoria d'aquestes dades s'han aconseguit gràcies a la web de OpenBeerDB [www.openbeerdb.com], la qual ofereix milers de cerveses amb molta informació complementària.

Per aquest cas tan sols s'ha tingut en compte 5 camps diferents:

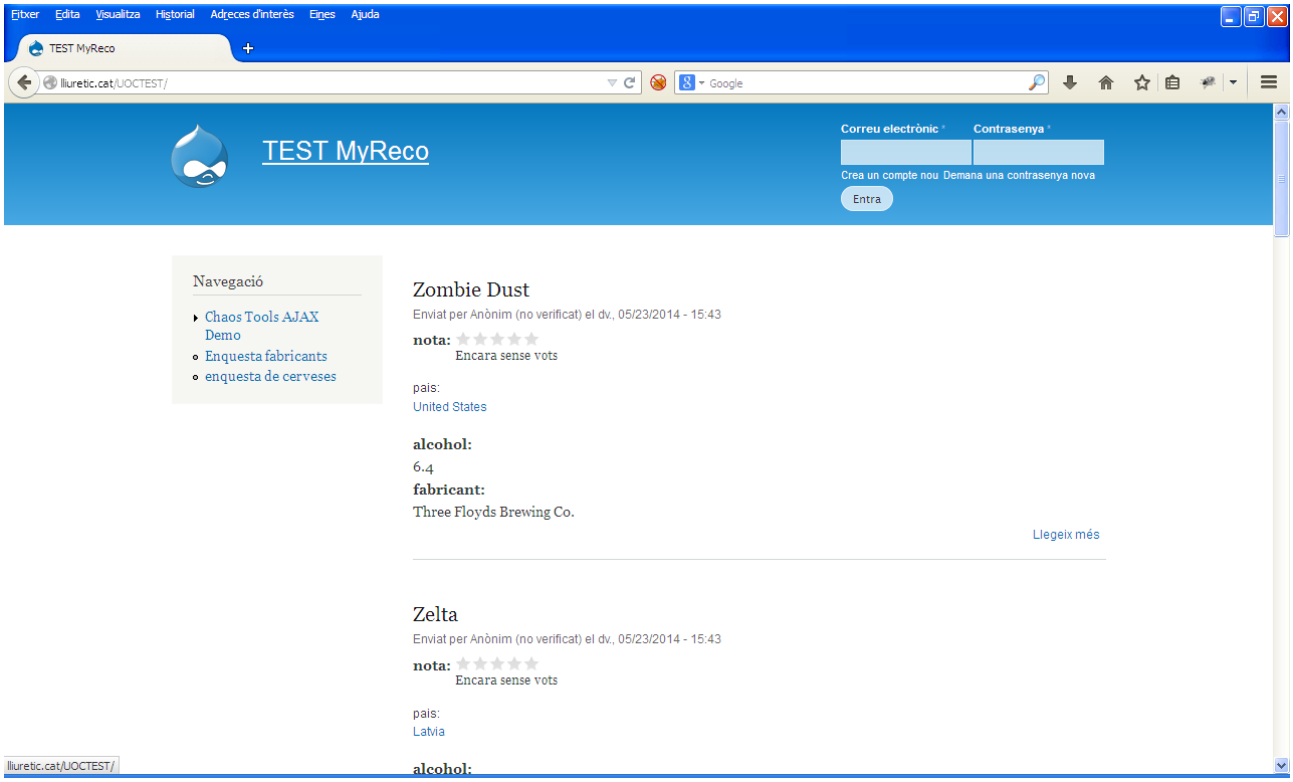
- guid: identificador de producte
- title: nom de la cervesa
- fabricant: empresa fabricant de la cervesa
- alcohol: graduació alcohòlica de la cervesa (pot estar en blanc si no es disposa de la dada)
- país: país on es fabrica la cervesa

Aquestes dades s'han agrupat en un simple fitxer CSV per a facilitar la seva importació de dades a altres bases de dades. Un petit exemple dels primers registres d'aquestes dades es troba a continuació:

```
"guid","title","fabricant","alcohol","pais"
100001,"1784 Anniversary Beer","vyturio Alaus Darykla","5.4","Lithuania"
100002,"1798 Revolution","Dublin Brewing",,"Ireland"
100003,"47 Bryg","Carlsberg Bryggerierne","7.0","Denmark"
100004,"77 Lager","BrewDog Ltd","4.9","Scotland"
100005,"A.K. Damm","Damm","4.8","Spain"
100006,"Abt 12","Brouwerij St. Bernardus","10.0","Belgium"
100007,"Affligem Dubbel","Brouwerij Alken-Maes","6.8","Belgium"
100008,"Aigua de Moritz","Moritz","0.0","Spain"
100009,"Alexander","Brouwerij Rodenbach",,"Belgium"
100010,"Alhambra Especial","San Miguel - Mahou","5.4","Spain"
```

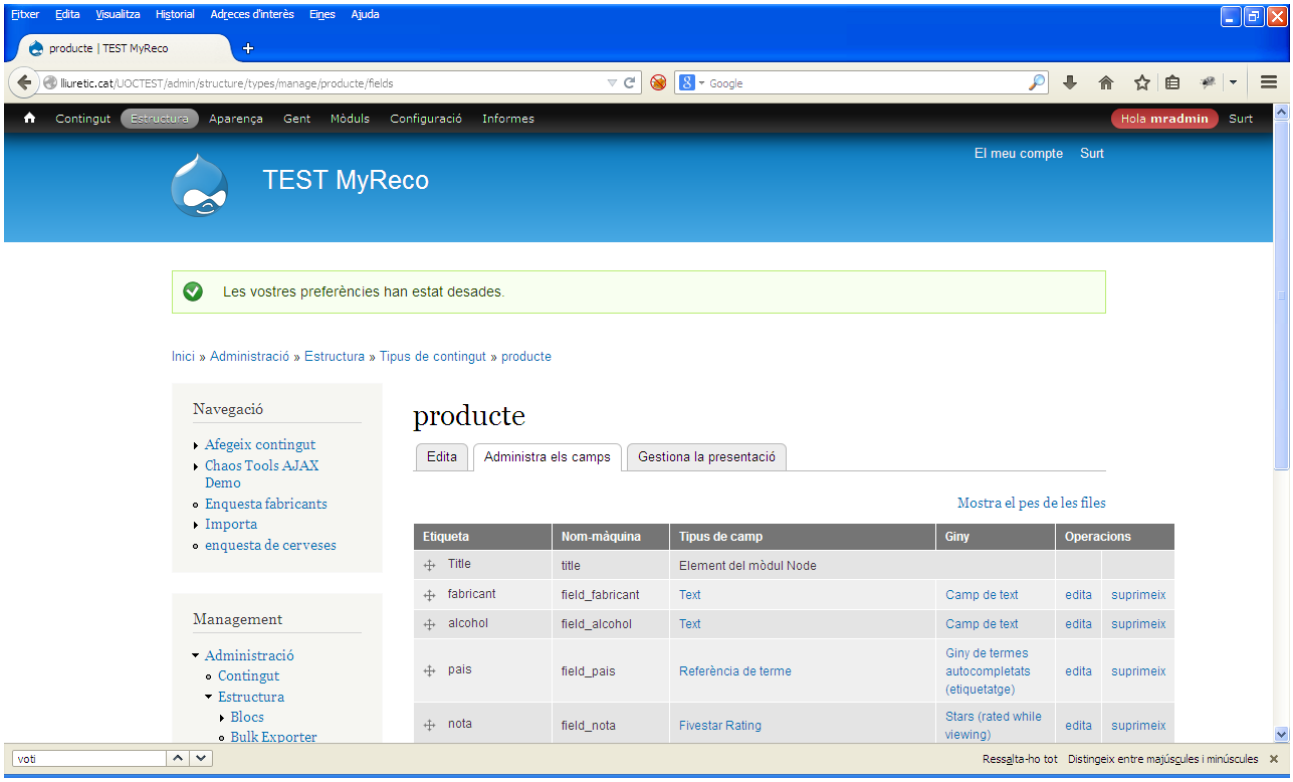
A partir d'aquestes dades inicials, amb el mòdul *Feeds* de Drupal s'han importat aquestes dades de cerveses en la base de dades de Drupal com a un tipus de contingut específic per cerveses.

Unes altres dades necessàries pel sistema d'entrada de dades són els usuaris. Amb Drupal ja va integrada una simple gestió d'usuaris per a que es donin d'alta al sistema tan sols posant un identificador, la seva contrasenya, i un correu electrònic vàlid.

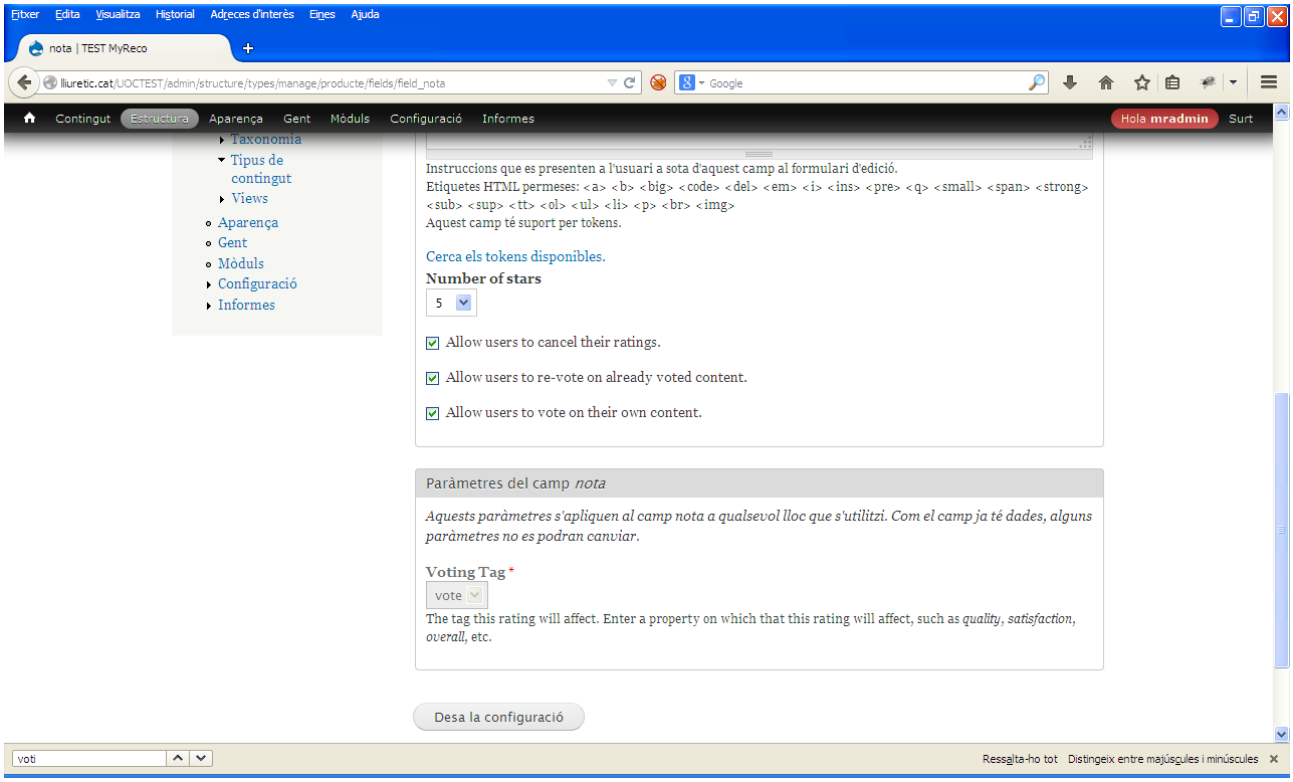


Un cop es tenen diferents productes i alguns usuaris, el següent pas es activar la relació d'aquestes 2 taules per a poder valorar aquests productes. Amb Drupal això es pot fer amb els mòduls *Fivestar* i *Voting API*, ja que permeten registrar les valoracions dels usuaris a cada producte, i es poden configurar con més ens convingui, per exemple, indicant el rang de valoracions pels productes.

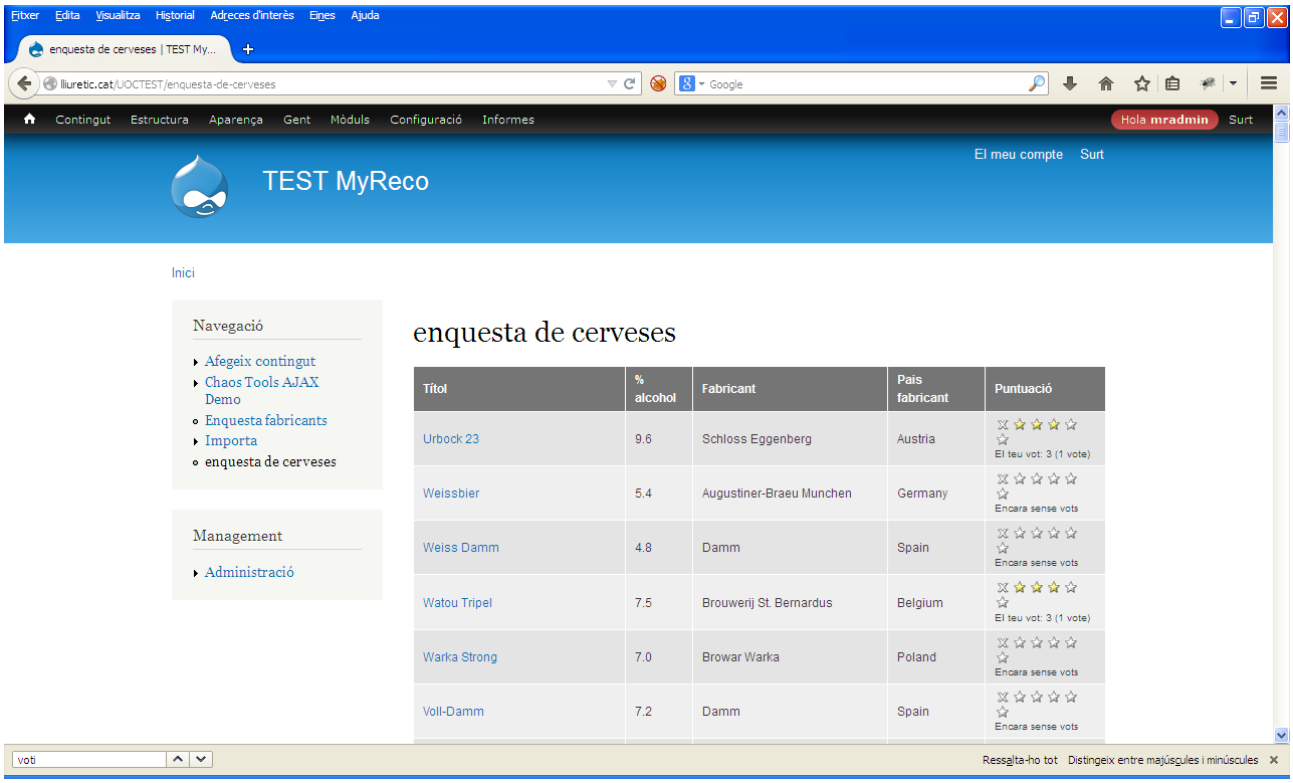
El tipus de contingut de les cerveses estaran composats pels mateixos camps que els definits en el fitxer CSV, a més d'un nou camp per registrar les valoracions.



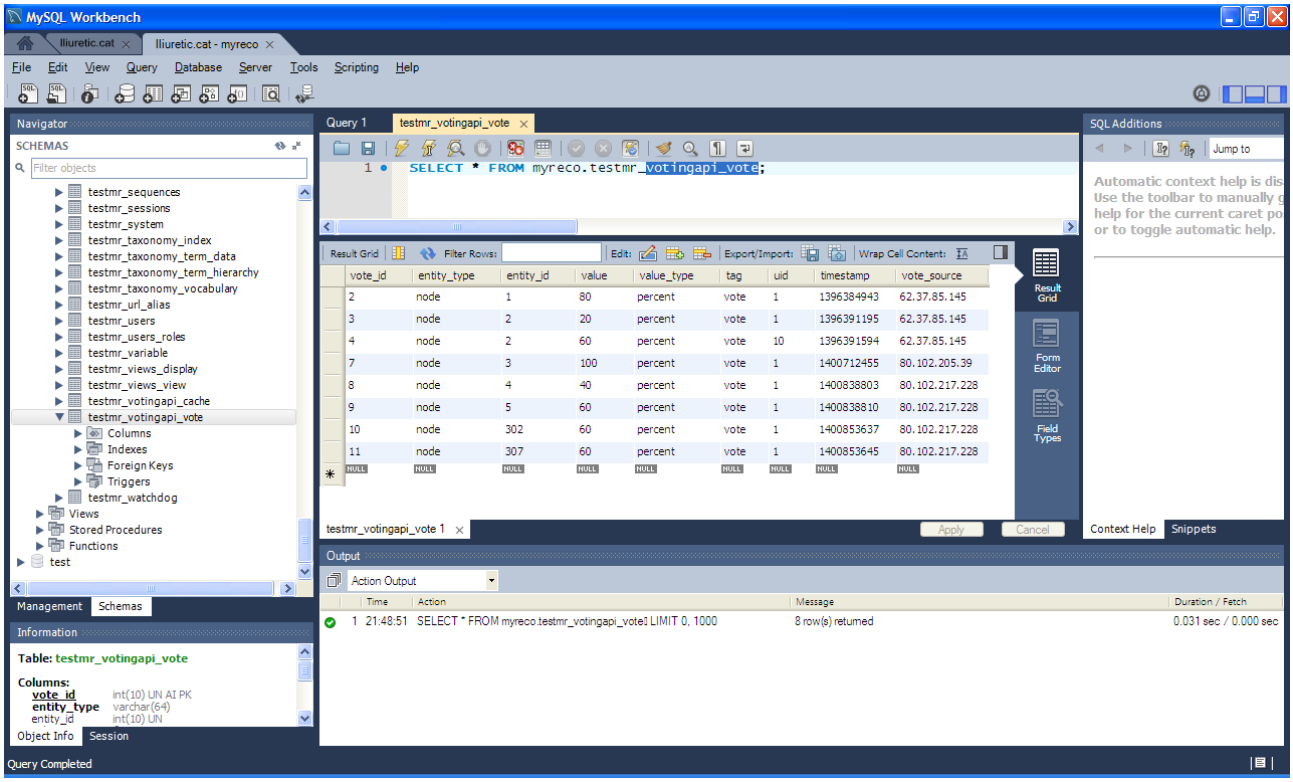
En la configuració d'aquest camp es pot triar el nombre màxim de puntuació que es pot donar, de 1 a 10.



I ja per acabar, tan sols faltarà crear una enquesta per a que els usuaris puguin valorar les cerveses de forma ràpida. Amb el mòdul *Views* es pot parametritzar el format en el que es mostren les dades, quins camps són ordenables, o quins camps es poden filtrar per fer l'enquesta més fàcil.



Aquestes dades queden registrades en la taula votingapi_vote, i es podran exportar en el format CSV necessari per a que es pugui aplicar al recomanador.



Les úniques dades necessàries en aquesta taula, requerides pel recomanador, són les columnes següents:

- entity_id: correspon al identificador de producte
- vote: correspon a la valoració de l'usuari en base 100
- uid: correspon al identificador de l'usuari

A continuació es mostra la versió utilitzada del nucli de Drupal i dels components complementaris utilitzats:

Versió utilitzada	Web de descàrrega
Drupal core 7.28 (maig de 2014)	https://drupal.org/project/drupal
Feeds 7.x-2.0-alpha8 (abril de 2013)	https://drupal.org/project/feeds
Fivestar 7.x-2.1 (març de 2014)	https://drupal.org/project/fivestar
Views 7.x-3.8 (maig de 2014)	https://drupal.org/project/views
Voting API 7.x-2.11 (març de 2013)	https://drupal.org/project/votingapi

5.7. Sortida de dades oberta per altres aplicacions

Aquesta tasca no s'ha pogut desenvolupar per falta de temps, però a continuació s'explicarà l'anàlisi realitzat pels desenvolupaments previstos inicialment.

En primer lloc calia obtenir els resultats del recomanador d'una forma oberta, amb dades estàndards, per a que fos accessible per a altres aplicacions externes. Els 3 formats principals amb els que estava previst treballar era en CSV, XML i JSON.

Per exportar qualsevol dada registrada en la base de dades MySQL, es pot fer de forma senzilla en CSV, ja que la majoria dels clients MySQL ja ofereixen aquesta eina d'exportació.

Un exemple de sortida de dades oberta que s'ha aplicat en aquest TFM han sigut els resultats de les avaluacions dels algoritmes del recomanador, on s'han desat en un fitxer CSV per a poder extreure gràfiques i estadístiques i valorar quina seria la millor configuració del model de dades avaluat.

```
Pearson;Threshold;0.4;0.8041562420950923;4,145 min
Pearson;Threshold;0.4;0.8038634541569608;4,105 min
Pearson;Threshold;0.4;0.8121097792254234;4,124 min
Pearson;NearestN;500-0.2min;0.7964109339066519;4,632 min
Pearson;NearestN;500-0.2min;0.7894135740416373;4,633 min
Pearson;NearestN;500-0.2min;0.7909649626556554;4,692 min
Euclidean;Threshold;0.4;0.7669897747524815;4,942 min
Euclidean;Threshold;0.4;0.7692763007068124;4,919 min
Euclidean;Threshold;0.4;0.7663852472123697;4,955 min
Euclidean;NearestN;500-0.2min;0.7603728125086958;4,887 min
Euclidean;NearestN;500-0.2min;0.7640133338117746;4,921 min
Euclidean;NearestN;500-0.2min;0.7679003953067033;4,902 min
```

El significat de cada columna és el següent:

1. Indica l'algorisme de similitud utilitzat
2. Indica l'algorisme de veïnatge utilitzat
3. Indica els paràmetres utilitzats en el càlcul de veïnatge
4. Mostra el resultat de l'avaluació
5. Mostra el temps que ha trigat en fer els càlculs

6. CONCLUSIONS

Un cop finalitzat el desenvolupament del TFM és l'hora de valorar la feina feta, des de la planificació inicial al tancament del projecte. Durant aquests mesos de feina també s'han produït algunes desviacions sobre la planificació inicial del projecte, ja que algunes tasques han tingut una dificultat més alta de l'esperada i han requerit més temps per resoldre-les.

Personalment em quedo amb la satisfacció d'haver adquirit nous coneixements amb la llibreria «Apache Mahout», l'entorn «Apache Hadoop», i tota la infraestructura que s'ofereix des de AWS. També s'ha pogut aplicar part dels coneixements adquirits durant el Màster Universitari en les assignatures de Intel·ligència Artificial Avançada, Gestió de projectes, Enginyeria del Software, o Sistemes distribuïts a gran escala.

6.1. Fites aconseguides

Al llarg de tot el TFM s'han aconseguit les següents fites:

Fites aconseguides
Planificar el projecte de desenvolupament d'un sistema de recomanació del productes.
Adquirir coneixements sobre el funcionament de la llibreria «Apache Mahout».
Configurar l'entorn de desenvolupament amb «Apache Mahout», Java i Eclipse.
Crear un recomanador de productes d'un tipus de producte concret, a partir de les dades públiques de MovieLens.
Crear un recomanador de productes de diferents tipus, a partir de les mateixes dades de MovieLens, i filtrant pel gènere de les pel·lícules.
Avaluar els resultats del recomanador amb diferents algoritmes de càlcul.
Adquirir coneixements sobre els servidors virtuals de AWS, i els entorns Hadoop per executar instruccions de tipus MapReduce.
Configurar un entorn de servidors virtuals distribuïts amb AWS, per executar el recomanador de productes amb «Apache Mahout» i «Apache Hadoop».
Dissenyar i començar el desenvolupament d'un sistema d'entrada de dades per obtenir noves valoracions d'usuaris i productes.
Adquirir coneixements per implementar un sistema de sortida dades oberta.

6.2. Desviacions en la planificació inicial del projecte

La dificultat principal que ha aparegut en aquest TFM s'ha trobat en la configuració d'un entorn distribuït per executar «Apache Mahout» amb Hadoop, ja que les darreres versions de cada programa utilitzat per preparar la infraestructura no eren compatibles entre elles, i ha dificultat molt el poder solucionar aquests problemes per poder engegar la infraestructura necessària.

Aquest inconvenient ha afectat a la resta de tasques previstes pel projecte, i ha fet que se centressin els esforços en fer funcionar el recomanador de forma distribuïda amb Hadoop, ja que formava parts d'un dels punts importants del TFM. D'aquesta manera, algunes de les tasques que s'havien de desenvolupar després d'aquesta han quedat afectades i s'han realitzat de forma més simplificada.

Tots aquests inconvenients que han anat apareixent s'han informat al consultor del TFM per anar acordant com resoldre'ls.

Segons el pla detallat del projecte inicial, les tasques 5, 6, 7 i 8 han patit canvis respecte a la seva durada i al seu abast. El fet d'allargar la durada de la tasca 5 ha afectat directament en l'abast de les tasques 6, 7, i 8, que s'han pogut fer en la part final del projecte però de forma molt més simplificada.

- **La primera versió del TFM (PAC2)**
  1. Preparació de l'entorn de desenvolupament
  2. Recomanador d'un tipus de producte concret
  3. Recomanador de diferents tipus de productes

- **La segona versió del TFM (PAC3)**
  4. Validació de resultats del recomanador
  5. Recomanador amb dades distribuïdes **(no finalitzat)**
  6. ~~Sistema d'entrada de dades pel recomanador~~ **(no realitzat en aquesta fase)**
  7. ~~Sortida de dades oberta per altres aplicacions~~ **(no realitzat en aquesta fase)**

- **Entrega final**

5. Recomanador amb dades distribuïdes **(finalització)**
6. Sistema d'entrada de dades pel recomanador **(simplificat)**
7. Sortida de dades oberta per altres aplicacions **(simplificat)**
8. Millores generals en totes les tasques **(simplificat)**
9. Validació general de resultats
10. Pròxims passos després del TFM
11. Memòria i presentació

6.3. Propers passos

Un cop finalitzat aquest TFM s'intentarà continuar amb la feina feta fins ara, per conèixer altres possibilitats que ofereix «Apache Mahout», i veure fins a quin punt pot sorgir una idea de negoci.

En primer lloc s'intentarà acabar l'apartat de l'entrada de dades per poder llançar al públic la plataforma per poder valorar diferents tipus de productes. En aquesta enquesta de recollida de dades seria interessant poder fer un estudi d'usabilitat, per aconseguir que els usuaris facin el màxim de valoracions en el menor temps possible.

També seria interessant definir com es faria la interfície d'usuari per a que aquests usuaris poguessin fer peticions de recomanacions via web. Aquesta part es continuaria fent amb un servidor individual, ja que el volum de dades inicial no es preveu que sigui molt gran.

En el moment que es tingui un volum de dades amb el que ja es puguin fer les primeres recomanacions als usuaris, es faran les primeres avaluacions del model de dades per veure quins paràmetres es poden ajustar millor per respondre a les peticions.

L'objectiu d'aquest recomanador és que pugui ser utilitzat pels usuaris de forma gratuïta i anònima. Mentre que les dades globals que es disposin es podrien vendre a les empreses en forma d'estadístiques, dels productes més valorats i menys valorats, i de les tendències de nous productes que es valoren més o menys.

Pel que fa a la part del servidor distribuït, es pot continuar fent proves i valorar altres possibilitats diferents al MapReduce amb Apache Hadoop. Actualment ja s'estan fent proves amb una nova plataforma, «Apache Spark», en el qual s'assegura que es millora fins a 10 vegades el rendiment de les operacions MapReduce en «Apache Hadoop».

També es pot dissenyar un històric de recomanacions, de manera que quan un usuari valora un producte que anteriorment s'havia recomanat, es podrà comprovar quina diferència hi havia entre la recomanació i la valoració real.

I ja per acabar, atès que es disposen de dades d'usuaris, es pot mirar de fer un estudi de com poden afectar les tècniques de datamining mantenint la privacitat dels usuaris.

7. BIBLIOGRAFIA

Informació general de sistemes de recomanació i filtratge col·laboratiu

- Sistemes de recomanació:
  - http://ca.wikipedia.org/wiki/Sistema_de_recomanaci%C3%B3
  - http://en.wikipedia.org/wiki/Recommender_system
  - http://www.cs.carleton.edu/cs_comps/0607/recommend/recommender/index.html

- Filtratge col·laboratiu:
  - http://en.wikipedia.org/wiki/Collaborative_filtering
  - http://recommender-systems.org/collaborative-filtering/

- Opinions generals:
  - http://online-behavior.com/targeting/recommendation-engines
  - http://datacommunitydc.org/blog/2013/05/recommendation-engines-why-you-shouldnt-build-one/

- Active Transfer Learning for Cross-System Recommendation:
  - http://www.cse.ust.hk/~qyang/Docs/2013/Zhao.pdf

- User-based Collaborative Filtering on Cross Domain by Tag Transfer Learning:
  - http://www.niculescu-mizil.org/KDD2012/forms/workshop/CDKD2012/doc/WWQ0626.pdf

- Can Movies and Books Collaborate? Cross-Domain Collaborative Filtering for Sparsity Reduction:
  - http://www.ijcai.org/papers09/Papers/IJCAI09-338.pdf

«Apache Mahout» en servidor individual

- Introducció completa a «Apache Mahout»:
  - https://www.ibm.com/developerworks/java/library/j-mahout/index.html
- Primers passos amb un recomanador amb «Apache Mahout»
  - http://mahout.apache.org/users/recommender/recommender-first-timer-faq.html
- Crear un recomanador basat en usuaris en 5 minuts:
  - https://www.youtube.com/watch?v=63k560Livmg
  - http://mahout.apache.org/users/recommender/userbased-5-minutes.html
- Crear un recomanador basat en productes:
  - https://www.youtube.com/watch?v=yD40rVKUwPI
- Sistema de recomanació distribuïts
  - http://ssc.io/deploying-a-massively-scalable-recommender-system-with-apache-mahout/
- Crear un recomanador creuat, amb filtre de productes:
  - http://stackoverflow.com/questions/8773861/candidate-strategy-for-genericuserbasedrecommender-in-mahout
  - http://www.warski.org/blog/2013/10/creating-an-on-line-recommender-system-with-apache-mahout/
- Avaluació de recomanador:
  - http://kickstarthadoop.blogspot.com.es/2011/05/evaluating-mahout-based-recommender.html
- Algoritmes de càlcul
  - http://www.slideshare.net/vangjee/a-quick-tutorial-on-mahouts-recommendation-engine-v-04
  -
- Collaborative Filtering with Apache Mahout:
  - http://ssc.io/wp-content/uploads/2013/02/cf-mahout.pdf
- Item-Based Collaborative Filtering Recommendation Algorithms:
  - http://files.grouplens.org/papers/www10_sarwar.pdf

«Apache Mahout» amb en servidors distribuïts

- Apache Hadoop
  - http://chimpler.wordpress.com/2013/01/20/deploying-hadoop-on-ec2-with-whirr/
  - http://chimpler.wordpress.com/2013/02/20/playing-with-the-mahout-recommendation-engine-on-a-hadoop-cluster/
  - http://www.xmsxmx.com/apache-whirr-create-hadoop-cluster-automatically/
- Distributed Matrix Factorization with MapReduce using a series of Broadcast-Joins, RecSys'13:
  - http://ssc.io/wp-content/uploads/2011/12/sys024-schelter.pdf
- Scalable Similarity-Based Neighborhood Methods with MapReduce, RecSys'12
  - http://ssc.io/wp-content/uploads/2012/06/rec11-schelter.pdf
- «Apache Spark»
  - http://gigaom.com/2014/03/27/apache-mahout-hadoops-original-machine-learning-project-is-moving-on-from-mapreduce/

Jocs de dades

- Valoracions de pel·lícules per usuaris:
  - http://grouplens.org/datasets/movielens/
- Llistats de productes de tot tipus:
  - http://www.freebase.com
- Llistat de cerveses espanyoles:
  - http://es.scribd.com/doc/141689143/Listado-de-Cervezas-Espanolas
- Llistat de fabricants de cervesa espanyols:
  - http://www.scribd.com/doc/118509654/Grupos-Cerveceros-Espanoles
- Base de dades de cerveses de tot el món:
  - http://openbeerdb.com
- Base de dades de cerveses de tot el món (basat en l'anterior, però més complet):
  - http://openbeer.github.io

8. GLOSARI

Terme	Descripció
Apache hadoop http://hadoop.apache.org/	Llibreria de programari lliure per emmagatzemar i processar dades en servidors distribuïts a gran escala.
Apache jclouds http://jclouds.apache.org/	Eina de programari lliure en Java que permet crear aplicacions portables pel núvol i sota control.
Apache mahout http://mahout.apache.org/	Llibreria de programari lliure desenvolupada en Java que implementa diferents algoritmes d'aprenentatge automàtic de forma escalable
Apache maven http://maven.apache.org/	Programari estàndard de gestió de projectes en Java que permet centralitzar compilacions, informes i documentació des d'un mateix lloc.
Apache spark http://spark.apache.org/	Llibreria de programari lliure de processament de dades a gran escala que millora el rendiment d'altres paradigmes com MapReduce en «Apache Hadoop».
Apache whirr https://whirr.apache.org/	Llibreria de programari lliure per executar serveis al núvol amb «Apache jclouds».
CSV	*Comma separated values*, format simple per emmagatzemar dades en un fitxer de text
Drupal https://drupal.org/	Sistema de gestor de continguts de programari lliure per crear aplicacions web, amb múltiples opcions de configuració.
Eclipse IDE http://www.eclipse.org/	Entorn de desenvolupament integrat (IDE), de programari lliure, i personalitzable per diferents mòduls.
HDFS	*Hadoop File System*. Sistema de fitxers de Hadoop.
JSON	*JavaScript Object Notation*. Format d'intercanvi de dades lleuger.
MapReduce	Paradigma de programació que permet processar grans volums de dades distribuïdes i en paral·lel.
PMBOK	*Project management body of knowledge*. Guia estàndard internacional per a la gestió de projectes.

Terme	Descripció
TFM	Treball final de Màster
XML	*Extensible Markup Language*. Llenguatge extensible d'etiquetes estàndard, amb gran utilitat per intercanviar dades entre aplicacions.
XP	*eXtreme Programming*, Metodologia àgil de desenvolupament

ANNEX 1: ALGORISMES DE CÀLCUL EN «APACHE MAHOUT»

A continuació es detallen els diferents algorismes de càlcul utilitzats en un recomanador de productes en «Apache Mahout». La majoria d'aquests algorismes s'han avaluat en aquest TFM en l'apartat **5.4. Validació de resultats del recomanador.**

A. 1.1. Similitud entre usuaris

En calcular la similitud entre 2 usuaris, els valors obtinguts es troben en el rang de -1.0 a +1.0. El valor +1.0 representa la màxima similitud entre 2 usuaris.

Amb Mahout es disposen de diferents càlculs de similitud:

- **Correlació de Pearson:** Es pot interpretar com el cosinus de l'angle obtingut pel vector de resultats entre 2 usuaris. En aquesta correlació les dades estan centrades, és a dir, que la seva mitjana és 0.

- **Correlació de Pearson no centrada:** És el mateix que la correlació de Pearson, però en aquest cas sense centrar les dades.

- **Correlació de Spearman:** Es calcula com la correlació de Pearson, però en aquest cas, s'ordenen les puntuacions de cada usuari i es dona un rànquing a cada puntuació. La primera del rànquing té la màxima puntuació, i l'última del rànquing té la mínima puntuació. Aquest tipus de càlcul pot tenir sentit quan es disposen de valoracions molt variades, per exemple, amb valors decimals. En el cas de les dades de MovieLens no té gaire sentit.

- **Distància Euclidiana:** El càlcul de la distància euclidiana és el que ofereix menys cost computacional, però també cal tenir en compte que els càlculs de similituds no estan normalitzats, de manera que no es poden comparar similituds amb dades de dominis diferents, com per exemple, quan hi ha diferents rangs de valoracions.

- **Distància Manhattan (o també City block):** Correspon a la suma dels valors absoluts de la diferència de cada direcció. El resultat final queda normalitzat entre els valors 0 i 1.

- **LogLikelihood:** Mètode pensat per puntuacions booleanes (tipus m'agrada / no m'agrada) en el que es tenen en compte paràmetres de sorpresa i de coincidència.

  [http://tdunning.blogspot.com.tr/2008/03/surprise-and-coincidence.html]

  [http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.14.5962]

- **Coeficient de Tanimoto:** Pensat per dades amb puntuacions booleanes (tipus m'agrada / no m'agrada).

  [http://en.wikipedia.org/wiki/Jaccard_index#Tanimoto_coefficient_.28extended_Jaccard_coefficient.29]

A. 1.2. Similitud entre productes

Apart dels càlculs que s'utilitzen per la similitud entre usuaris, també hi ha altres algoritmes que es fan servir per calcular la similitud entre productes. Amb «Apache Mahout» hi ha el següent:

- **Cooccurrència:** Es considera que 2 productes són molt similars si acostumen a aparèixer junts en les valoracions dels usuaris.

A. 1.3. VEÏNS DE CADA USUARI

Per calcular quins són els veïns de cada usuari es poden fer servir 2 mètodes diferents.

- **Basat en llindar (Threshold Neighborhood):** Permet calcular els veïns d'un usuari en funció de la seva similitud, utilitzant un valor llindar:

  - Un llindar gran (proper a 1) dona lloc a tenir un petit nombre de veïns, però molt semblants.

  - Un llindar petit (proper a 0) dona lloc a tenir un major nombre de veïns, però molt variats.

  Amb aquest càlcul es pot donar la situació que alguns usuaris tinguin molts veïns, perquè són molt similars entre ells, i d'altres usuaris que tinguin pocs veïns perquè n'hi ha pocs com ell. Els usuaris amb molts veïns podrien tenir unes recomanacions més acurades, i els usuaris amb pocs veïns tindrien unes recomanacions menys acurades.

- **N veïns propers (Nearest N Neighborhood):** Permet indicar el nombre N de veïns propers per a cada usuari. D'aquesta manera tots els usuaris tindran el mateix nombre de veïns.

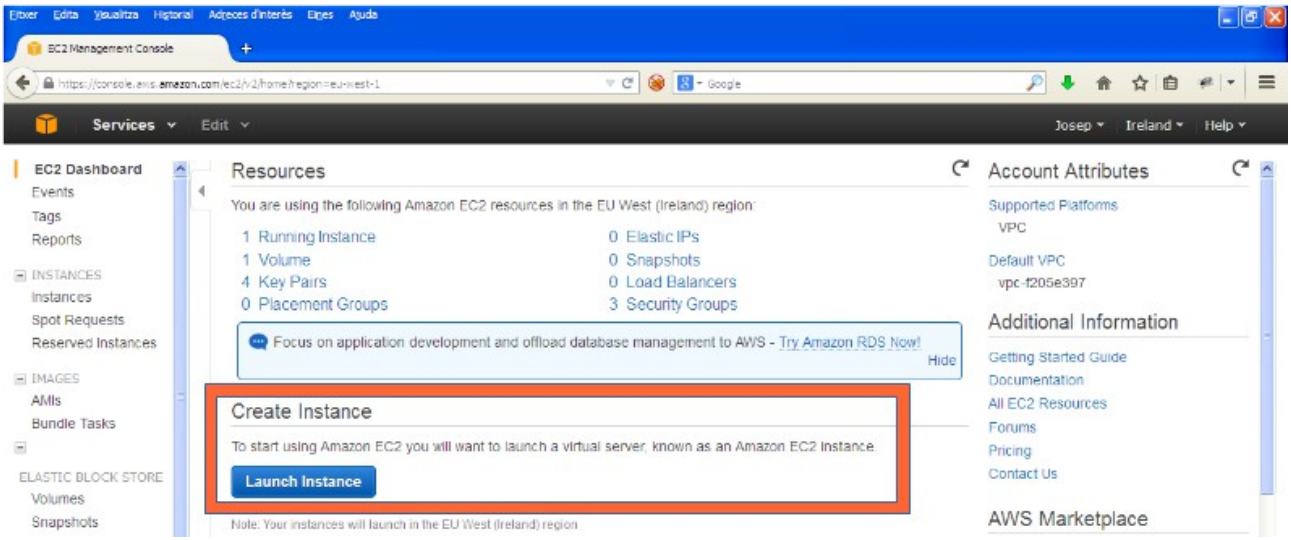
  Tot i així també es pot utilitzar aquesta funció indicant un mínim de similitud, de manera que si algun dels seus N veïns està molt llunyà, tampoc es tindria en compte.

  Cal anar amb compte amb aquest valor mínim, ja que si s'utilitza de forma molt restrictiva pot donar que hi hagi usuaris amb pocs veïns.

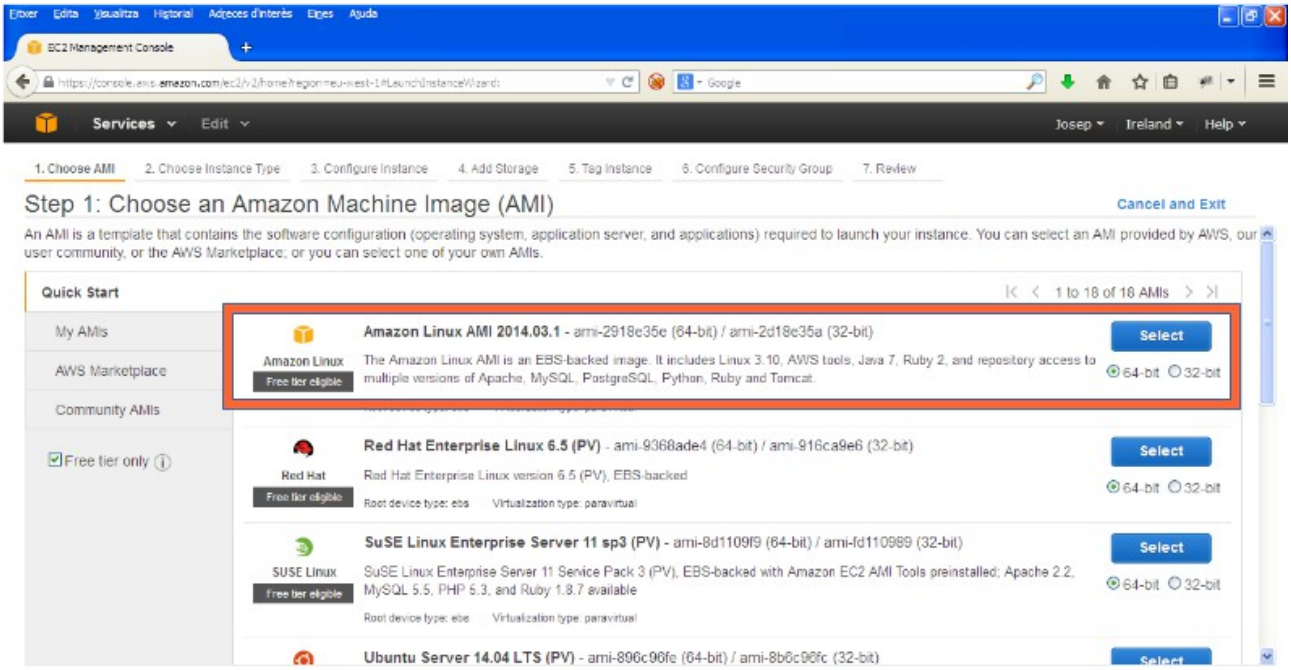
ANNEX 2: INSTAL·LACIÓ I CONFIGURACIÓ D'ENTORN DISTRIBUÏT AMB «AMAZON WEB SERVICES» (AWS)

A continuació es detalla com s'ha fet la instal·lació i configuració de l'entorn distribuït amb «Amazon Web Services» (AWS). Amb aquesta instal·lació es podrà executar «Apache Mahout» per calcular recomanacions de productes de forma distribuïda mitjançant el paradigma «MapReduce», tal i com s'explica en aquest TFM en l'apartat **5.5. Recomanador amb dades distribuïdes**.

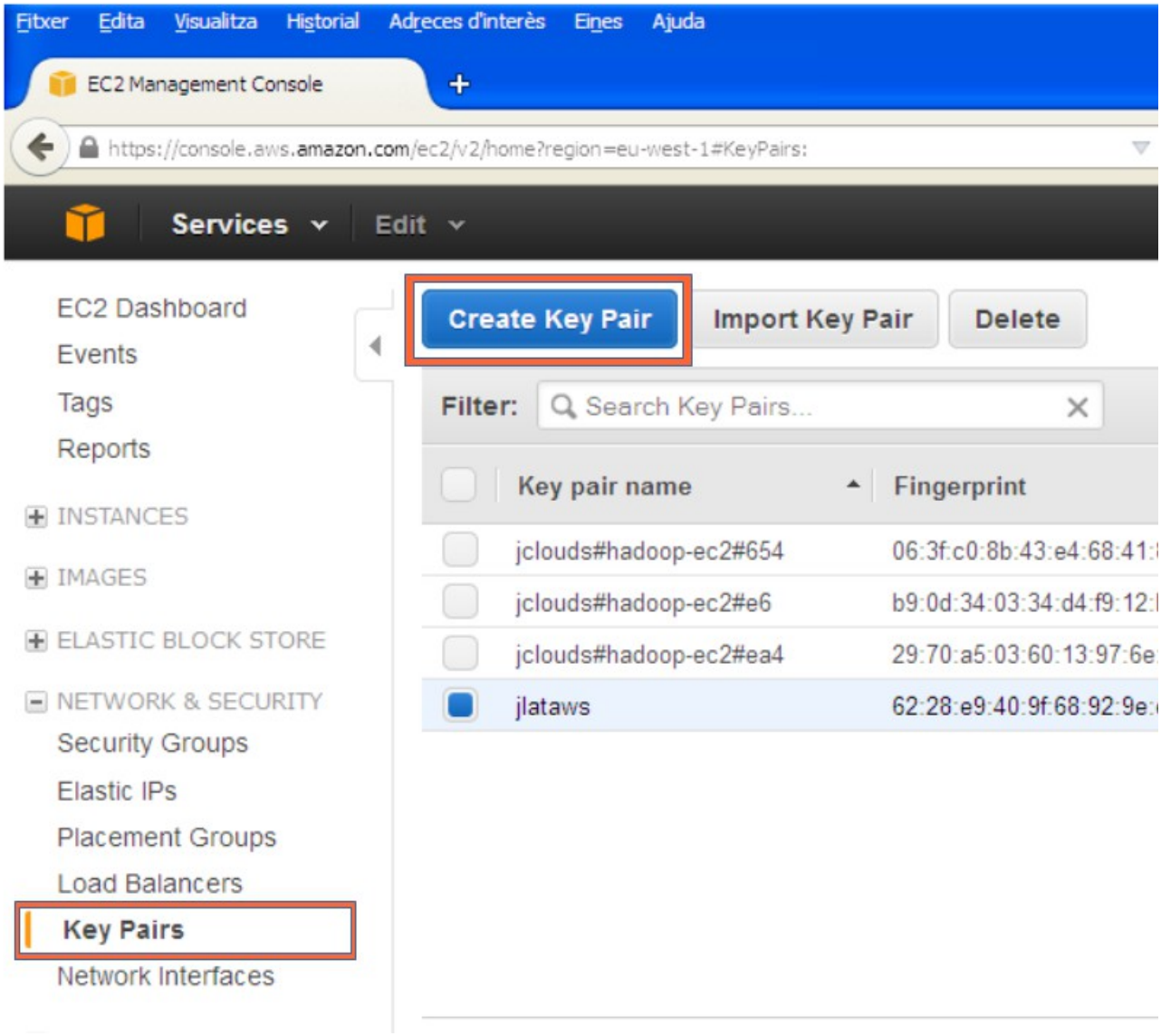
En primer lloc cal crear una instància d'un servidor virtual al núvol d'Amazon, mitjançant el servei «Elastic Cloud Computing» (EC2).



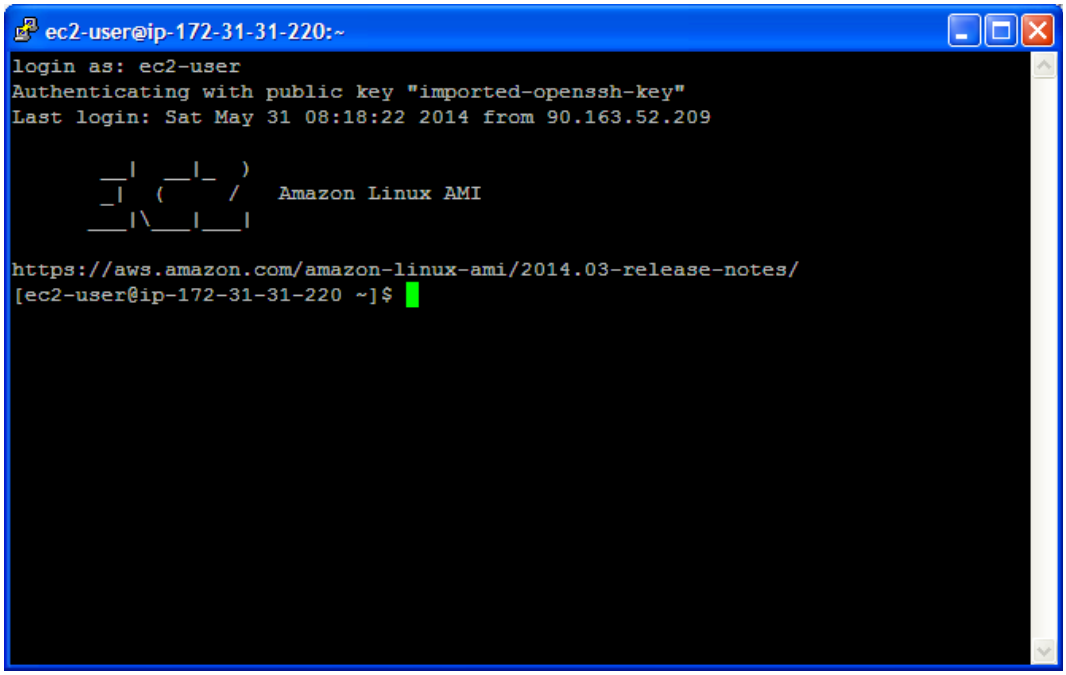
A continuació cal triar el tipus de servidor, i configurar-lo amb els paràmetres desitjats. En el aquest cas s'ha seleccionat una distribució Linux del propi Amazon, el qual ja incorpora la gran majoria del programari desitjat.



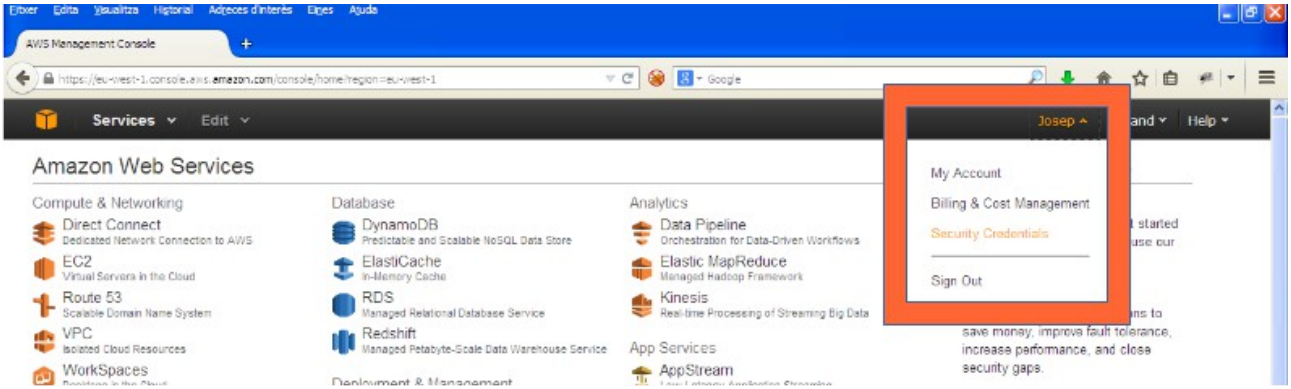
Per accedir a aquesta instància cal configurar-se una parella de claus d'accés segur.



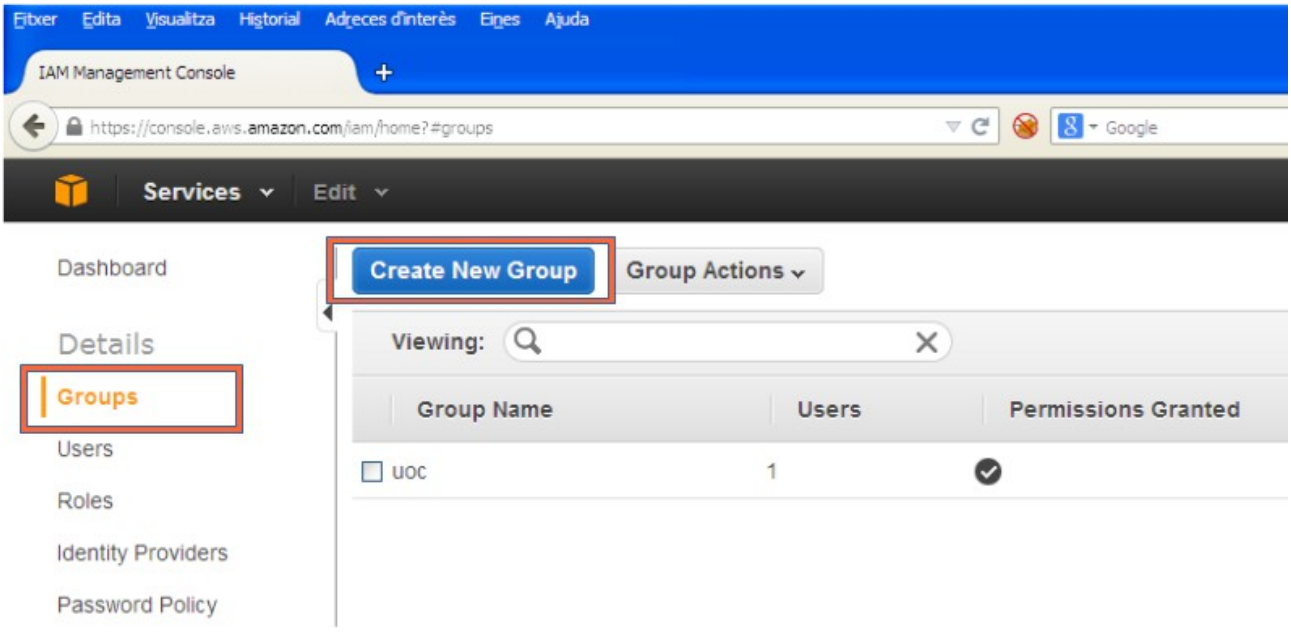
Aquesta parella de claus es podrà fer servir amb una connexió segura en mode consola amb l'aplicació PuTTY, o en mode SFTP amb l'aplicació WinSCP.



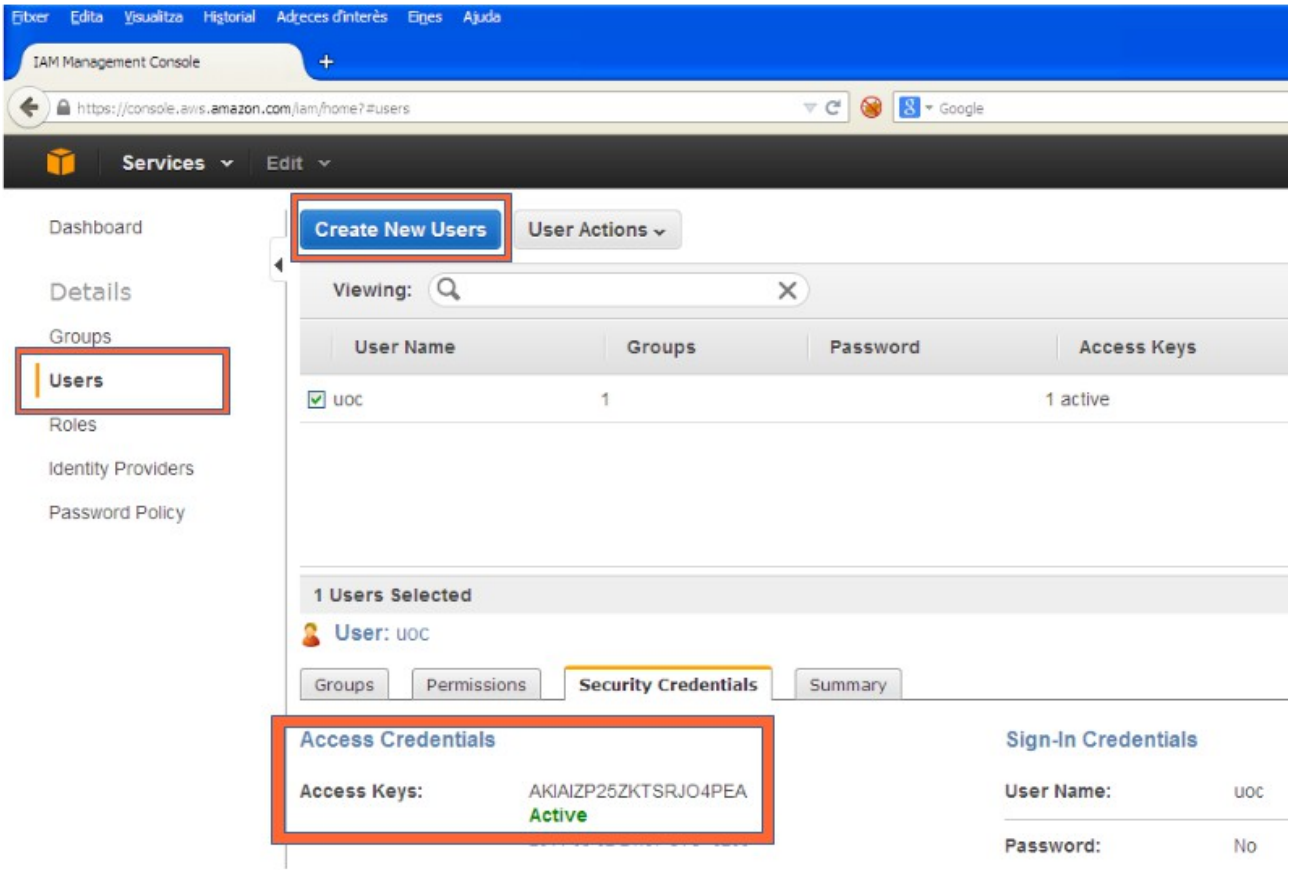
Un cop creada la instància, cal configurar l'accés segur de noves aplicacions. Des de la consola principal del nostre AWS s'accedeix a la gestió de credencials de seguretat (Security credentials):



En aquest apartat, cal crear un grup i un usuari amb les seves claus d'accés.



En aquest exemple, s'ha creat un grup «uoc» i un usuari «uoc» amb les seves corresponents claus d'accés als serveis del servidor.



El següent pas és instal·lar el programari necessari en les seves versions compatibles corresponents.

Instal·lar Apache Whirr 0.82, Apache Hadoop 1.2.1, i Java Developement Kit 1.7.0_45:

```
~$ tar xvzf whirr-0.8.2.tar.gz
~$ tar xvzf hadoop-1.2.1.tar.gz
~$ tar xvzf jdk-7u45-linux-x64.tar.gz
```

Per configurar l'Apache Whirr, en primer lloc cal crear una parella de claus, i autoritzar l'accés al propi servidor:

```
~$ ssh-keygen -t rsa -P ''
~$ cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
```

I a continuació cal crear el fitxer de configuració de Whirr per AWS amb Hadoop. En el nostre cas li donarem el nom de «*hadoop-ec2.properties*»:

```
whirr.cluster-name=hadoop-ec2
whirr.cluster-user=${sys:user.name}
whirr.instance-templates=1 hadoop-namenode+hadoop-jobtracker,2 hadoop-datanode+hadoop-tasktracker
whirr.hadoop.version=1.2.1
whirr.provider=aws-ec2
whirr.identity=AKIAIZP25ZKTSRJO4PEA
whirr.credential=<CLAU PRIVADA>
whirr.private-key-file=${sys:user.home}/.ssh/id_rsa
whirr.public-key-file=${whirr.private-key-file}.pub
whirr.hardware-id=t1.micro
whirr.image-id=eu-west-1/ami-2918e35e
whirr.location-id=eu-west-1b
whirr.java.install-function=install_oab_java
```

En aquest fitxer de configuració s'haurien d'ajustar els diferents paràmetres.

- Les claus d'accés: en els paràmetres «whirr.identity» i «whirr.credential»
- El proveïdor del servei: en el paràmetre «whirr.provider»
    - Pel nostre cas seria aws-ec2. Hi ha altres proveïdors compatibles amb whirr que es poden trobar en la seva web

- El nom de la instància del servidor: en els paràmetres «whirr.image-id» i «whirr.location-id»
- El nombre d'instàncies a desplegar: en el paràmetre «whirr.instance-templates».
    - Per exemple, amb la configuració actual s'executa 1 instància amb hadoop-namenode i hadoop-jobtracker, i 2 instàncies amb hadoop-datanode i hadoop-tasktracker.

```
1 hadoop-namenode+hadoop-jobtracker, 2 hadoop-datanode+hadoop-tasktracker
```

    - En el cas que es volgués desplegar 4 hadoop-datanode, s'hauria de configurar de la següent manera:

```
1 hadoop-namenode+hadoop-jobtracker, 4 hadoop-datanode+hadoop-tasktracker
```

Es poden trobar altres configuracions diferents en la mateixa guia online de whirr:

http://whirr.apache.org/docs/0.8.2/configuration-guide.html

Un cop s'ha configurat l'entorn que es vol crear ja es pot iniciar el seu desplegament. En primer lloc s'ajusten les rutes de cada aplicació:

```
~$ export HADOOP_PREFIX=~/hadoop-1.2.1
~$ export WHIRR_HOME=~/whirr-0.8.2
~$ export JAVA_HOME=~/jdk1.7.0_45/
~$ export PATH=$JAVA_HOME/bin:$HADOOP_PREFIX/bin:$WHIRR_HOME/bin:$PATH
```

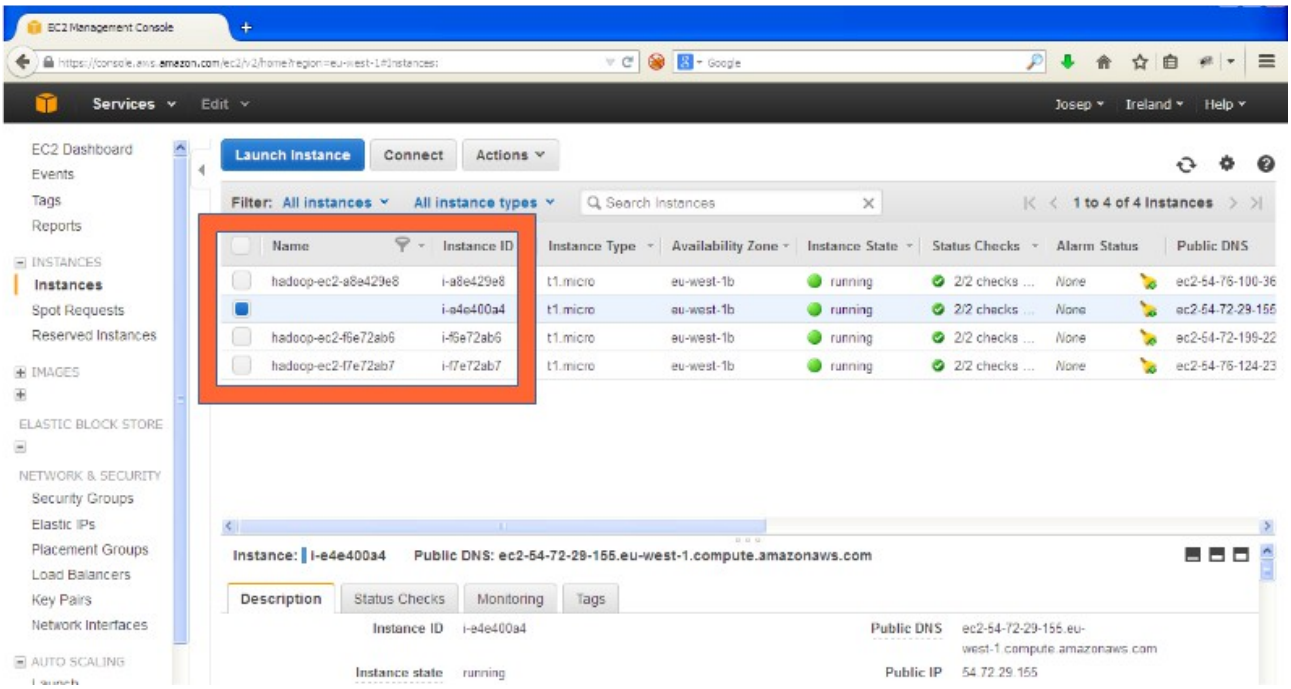
I a continuació ja es pot crear l'entorn distribuït amb Hadoop segons la configuració del fitxer *hadoop-ec2.properties*:

```
~$ whirr launch-cluster --config ~/hadoop-ec2.properties
```

En el moment que es vulgui eliminar aquesta infraestructura tan sols cal destruir-la amb una simple instrucció, utilitzant el mateix fitxer de configuració *hadoop-ec2.properties*:

```
~$ whirr destroy-cluster --config ~/hadoop-ec2.properties
```

En revisar les instàncies creades en el panell de control de AWS, apareixerà la instància creada inicialment, i les 3 noves instàncies creades des de Whirr.



Per continuar amb la posada en marxa de l'entorn distribuït amb Hadoop cal executar el seu proxy de la següent manera:

```
~$ export HADOOP_CONF_DIR=~/.whirr/hadoop-ec2/
~$ sh $HADOOP_CONF_DIR/hadoop-proxy.sh
```

A continuació fer una nova connexió cap a la mateixa instància original creada amb AWS. Es tornen a ajustar les rutes de cada aplicació:

```
~$ export HADOOP_PREFIX=~/hadoop-1.2.1
~$ export PATH=$HADOOP_PREFIX/bin:$PATH
```

També s'actualitza la ruta de configuració de Hadoop, per a que utilitzi els generats anteriorment pel Whirr, i li canviem el nom del fitxer de configuració per evitar un avís de fitxer obsolet.

```
~$ export HADOOP_CONF_DIR=~/.whirr/hadoop-ec2/
~$ mv ~/.whirr/hadoop-ec2/hadoop-site.xml core-site.xml
```

Per comprovar que Hadoop s'executa de forma correcta, es pot fer una senzilla prova d'executar una tasca de tipus MapReduce. En primer lloc ens posem en la carpeta d'instal·lació de Hadoop, i afegir el fitxer LICENSE.txt al sistema de fitxers de Hadoop «HDFS».

```
~$ cd $HADOOP_PREFIX
~$ hadoop fs -put LICENSE.txt LICENSE.txt
```

Executem la tasca de comptar paraules del fitxer, i mostrem els resultats obtinguts per comprovar que tot s'ha engegat correctament.

```
~$ hadoop jar hadoop*examples*.jar wordcount LICENSE.txt output
~$ hadoop fs -cat output/part-r-00000
```