# 

# **Trabajo Profesional**

# **HyalCore**

Plataforma de recomendaciones

# 

# **Nicolas Ayala [88276] ncls.ayala@gmail.com**

# **Diego Malchinsky [85373] malcha@gmail.com**

# 

# **Tutora: Adriana Echeverria**

# 

# 

# **HyalCore**

# *Hybrid-Algorithm Collaborative Recommendation*

***Introducción***

¿Qué es un sistema de recomendaciones? Es un sistema que se encarga de mostrar productos o servicios a los usuarios finales. No cualquier producto o servicio, sino los que puedan llegar a ser de interés para el usuario. Esto es un gran beneficio para los usuarios ya que encuentran lo que necesitan de manera más directa dentro de las enormes bases de datos de las tiendas en Internet a la vez que pueden llegar a descubrir nuevos productos o servicios dentro de su interés que si no fuera por este tipo de sistema quizás no podrían llegar a los mismos. Del lado de la tienda que ofrece sus productos por Internet, también es de gran valor ya que le permite obtener más transacciones y conocer un poco mejor a sus usuarios.

Los principales tipos de sistemas de recomendación son:

* Sistemas de recomendación basados en contenido: se basan en la similitud entre objetos. Predicen que para un usuario serán de interés aquellos items muy parecidos en su contenido con aquellos que ya sabemos que han sido de su agrado.
* Sistemas de recomendación colaborativos: las recomendaciones se realizan basándose solamente en los términos de similitud entre los usuarios. Es decir, los sistemas colaborativos recomiendan objetos que son del gusto de otros usuarios de intereses similares.
* Sistemas de recomendación basados en conocimiento: realizan inferencia entre las necesidades y preferencias de cada usuario para sugerir recomendaciones. Necesitan tener un conocimiento general sobre el conjunto de objetos y un conocimiento informal de las necesidades del usuario. Requieren un gran esfuerzo humano para realizar las recomendaciones mediante todo tipo de heurísticas de inferencia.
* Sistemas de recomendación demográficos: tienen como objetivo clasificar al usuario en función de sus características demográficas, realizando a continuación las recomendaciones basándose en clases demográficas.
* Sistemas de recomendación híbridos

Los problemas más importantes a los que los sistemas de recomendación deben enfrentar son:

* Nuevos usuarios: los usuarios que recién ingresan al sistema no tienen ni historial de interacción con items, ni un perfil intereses definido. Generar una buena recomendación es difícil para estos usuarios.
* Nuevos items: de manera análoga un nuevo producto del que no se dispone todavía de suficientes datos de acceso de los usuarios a éste, puede ser complicado de recomendar.
* Escasez: los sistemas de recomendación colaborativos necesitan una gran cantidad de datos, muchos usuarios puntuando muchos ítems para así poder realizar recomendaciones en base a estas puntuaciones.
* Escalabilidad: los sistemas de recomendación colaborativos usan por norma general algoritmos de cálculo de los k vecinos más cercanos (knn, K-nearest neighbors) para obtener la similaridad entre usuarios. Estos algoritmos son computacionalmente costosos y su coste crece linealmente cuanto mayor sea el número de usuarios y de ítems por lo que con bases de datos con millones de elementos, al aumentar el número de datos, el sistema sufrirá graves problemas de escalabilidad.

Estamos convencidos de que un solo algoritmo no puede atacar todos los problemas presentados y nunca va a llegar a ofrecer las mejores recomendaciones. Creemos que el uso de un sistema que se sepa adaptar al contexto del usuario para ofrecer recomendaciones va a obtener mejores resultados. Esto involucra el desarrollo de una infraestructura que permita almacenar y disponer de la información en forma flexible y escalable, para que distintos algoritmos de recomendación determinen el mejor conjunto de items a recomendar.

El objetivo de este trabajo es crear una plataforma que ofrezca recomendaciones para todo tipo negocio, cuya principales características sean:

* Flexibilidad para la carga de información: nuestra plataforma necesita información y no queremos que la carga de la misma sea un problema para nuestro negocios cliente.
* Disponibilidad: siempre se va a poder cargar información, y siempre se van a poder pedir recomendaciones.
* Múltiples algoritmos de recomendación con parámetros ajustables
* Testing A/B de algoritmos de recomendación
* Ecosistema colaborativo
* Visualización de métricas e información provista

# ***Objetivo***

***Desarrollar una plataforma de recomendaciones, la cual posibilite la carga de los ítems a recomendar, sus características, las interacciones que los usuarios tuvieron con los mismos y las interacciones entre los usuarios.***

# 

# ***Alcance***

## *Requerimientos Funcionales*

A continuación, se describen los principales requisitos funcionales del futuro producto.

***Carga de features:***

Los features son las distintas características que poseen los items, por ejemplo pueden ser precio, talle, modelo, marca, colores, region, activo, etc.

El cliente debe poder realizar alta, baja, modificación y consulta de los features.

Estructura:

id, Name, Description, FeatureTypeId, FutureDomainId:

Validaciones:

Al realizar una baja de un feature se debe validar que el mismo no pertenezca a algún item

FeatureType:

El featureType indica el tipo de característica, el cual es usado para segmentar items y determinar la importancia de un feature para una recomendación

Los tipos pueden ser:

Variant: Identifica los features que pueden variar entre 2 ítems equivalentes, por ejemplo el talle, precio y color en el ítem zapatilla.

Determinant: Identifica los features que son determinantes para identificar que dos items no son equivalentes, por ejemplo el modelo, y la marca en un item zapatilla.

CategoryType: Identifica el feature que segmenta fuertemente a un conjunto de ítems

Estructura:

Id, Description

FeatureDomain:

Los featuresDomains son los distintos valores que pueden tener un Feature.

Los featuresDomains pueden ser:

Category; Int; Array; String; Date; Float

Estructura:

Id, Description

Category:

La categoría indica el tipo de un ítem. Estas pueden estar jerarquizadas para tener una mejor descripción de un Item, por ejemplo se Puede Tener la Category TvLed que esta abajo de la Categoría Tv que a su vez esta abajo de Electronics.

El cliente debe poder realizar alta, baja, modificación y consulta de las Categories.

Estructura:

Id, Name, Description, ParentCategoryId, GlobalCategoryId

GlobalCategory:

Las categorías globales son categorías standards comunes a todos los clientes

Se debe proveer una lista de Categorías globales standart, entre las cuales estan por ejemplo Sports, Fashion, Car, Electronics:

Estructura:

Id, Name

FeatureValue:

Los FeatureValues es el par Feature con su valor correspondiente para un item

El cliente debe poder realizar alta, baja, modificación y consulta de los featuresValues.

Estructura:

FeatureId, Value

Validacion:

Al establecer un valor a un future de debe validar que dicho valor pertenezca al featureDomain establecido

***Carga de Items :***

Los ítems son los elementos a recomendar

El cliente debe poder realizar alta, baja, modificación y consulta de los ítems.

Estructura:

Id, Name, ClientId, Description, lista de featureValues

Description: es información general del ítem, usado para extraer palabras claves

Validaciones:

Entre los FeaturesValues al menos debe haber un feature cuyo featureType sea Category.

Entre los FeaturesValues al menos debe haber un feature cuyo featureType sea Determinant.

Si el item tiene un límite de vigencia, ya sea por un indicador o flag ‘activo’, fecha de fin de vigencia o contador de cantidad, este Feature debe estar especificado y su FeatureValue debe actualizarse. De no existir, el item será vigente siempre.

***Carga de usuario***

El cliente debe poder realizar alta, baja, modificación y consulta de los usuarios.

Estructura:

Id, Name, Email, ClientId, SocialNetworkIds

***Carga de tipos de interacciones entre usuarios***

Los tipos de interacciones varían de negocio a negocio, ejemplos son: see, like, share, comment, buy, intention of buying.

Cada tipo de interacción puede tener asociado un peso que indique la relevancia de la interacción.

Las recomendaciones dependen mucho del contexto en el que se sitúa el usuario y un lugar importante para obtener dicha información son las últimas búsquedas que realizó

Estructura:

Id, Name

***Carga de interacciones***

***usuario - item***

Se debe permitir cargar las interacciones entre usuarios y los items

Estructura:

UserId, Interaction, ItemId, Date, IteractionValue

interacción: Like, Share, Comment, Buy, Interest o Pendding to buy. Rate y su valor si es el caso.

Date: fecha de la interacción

InteractionValue: Es opcional, este puede ser usado al valorar un item

***usuario - usuario***

Se debe permitir cargar las interacciones entre usuarios

Estructura:

UserIdFrom, UserIdTo, InteractionId,Date

interacción: Follow, Friend, Invite, Mention

Date: idem usuarios-items

***usuario (unilateral)***

Es una acción unilateral realizada por el usuario, estas pueden ser:

Realizó una búsqueda

Ordenó/ filtro por algún tipo de característica

Ingreso al sistema

Indico su ubicación

Se debe permitir cargar las interacciones de usuarios

Estructura:

UserId, InteractionId, Date, InteractionValue

***usuario - ítem red social***

Interacción realizada con un objeto en una determinada red social, ejemplos son:

Like en una página de Facebook

Estructura:

UserId, InteractionId, Date

***Recomendación de items***

Contar con distintos algoritmos de recomendación que se adapten a la información de cada usuario y los requerimientos de cada cliente. Para esto se va a analizar primero el contexto del usuario, información que se tiene del mismo y en base ello se va a utilizar un algoritmo u otro. Para cada situación se pueden elegir distintos algoritmos para recomendar distintos items para el mismo contexto y usuario (ver Testing A/B)

Se debe recomendar los items, sobre la base de:

Items de características similares a los items que le interesan al usuario

Items que le interesan a los usuarios que estan relacionados con el usuario en cuestión

Items de interés de usuarios con un perfil similar

Para ello se debe implementar los siguientes algoritmos

Algoritmo de recomendación basado en la información del usuario/contexto

Algoritmo de recomendación basado en la interacción del usuario

Algoritmo de recomendación basado en personas de usuarios con los que interaccionó

Algoritmo de recomendación basado en personas de perfil similar

Se debe reconocer el contexto del usuario para utilizar el algoritmo más adecuado.

Los algoritmos deben contemplar los siguientes puntos:

- Falsos positivos

- Ordenamiento de las recomendaciones:

ordenadas según relevancia

ordenamiento random

- Dos pedidos de recomendaciones seguidos (o en determinado rango de tiempo) deberían responder distintas recomendaciones para un mismo usuario

- Agrupamiento de los ítems: no es deseado realizar una recomendación de ítems equivalentes. Osea si a un usuario se le recomiendan 5 zapatillas de correr del mismo modelo pero de diferente color no va a ser una buena experiencia o al menos no mejor que una recomendación de 5 zapatillas de correr de distintas características.

- Fallbacks: siempre recomendar algo, aunque sea un grupo de ítems previamente definido

- La performance de la búsqueda debe ser aceptable (no más de 2 segundos)

***Ecosistema colaborativo***

Una fuente de datos extra son la base de datos de recomendaciones de otros usuarios del sistema, que en forma colaborativa se proveen de información en beneficio de todos. Así la información que se tiene de un usuario dado en un negocio se puede cruzar con la de otro negocio para hacer más efectiva la recomendación.

El ecosistema colaborativo permite construir el perfil del usuario tomando información de varios negocios. Con esta información extra podemos construir un algoritmo más efectivo viéndose beneficiados los clientes cuya configuración permita la colaboración. Es importante destacar que los clientes no acceden a la información de los usuarios en otros negocios, sino que es el algoritmo de recomendación el que se nutre de la colaboración y los perfiles globales generados por el mismo.

***Testing A/B de algoritmos***

Para un mejor resultado se recomienda usar al menos 2 algoritmos distintos, al mismo tiempo, para mostrar las recomendaciones e ir ponderando los algoritmos en base al éxito o fracaso de cada uno. Cuando los usuarios interaccionan con los resultados del algoritmo en cuestión, se lo define como éxito.

***Autenticación del cliente***

Para una buena administración de nuestros recursos y la seguridad de la información provista por el cliente, es necesario que el intercambio de información tenga un nivel de seguridad aceptable. Es por eso que cada cliente contará con un token único para acceder a su información que se renovará periódicamente.

Se debe autenticar al cliente debido a que se manejan datos confidenciales, por los tanto dentro de dicha autentificación se encuentran los siguientes roles, que le permiten al cliente realizar distintas operaciones:

Administrador:

Puede agregar, modificar, eliminar items, tipos de características, tipos de interacciones, interacciones y usuarios (usuarios del negocio y usuarios que consumen nuestra API).

Operador de datos:

Puede agregar, modificar, eliminar items, tipos de características, tipos de interacciones, interacciones y usuarios (sólo usuarios del negocio), recomendaciones certeras

Consumidor de datos

Tiene el permiso para pedir una recomendación, o para pedir un training sets.

Analista

Tiene el permiso para ver los reportes que se obtienen de la aplicacion.

***Visualización de la información provista***

El cliente contará con un panel para visualizar métricas de la información provista:

cantidad de interacciones por tipo

cantidad items

cantidad de items con perfil completo

cantidad de usuarios

cantidad de usuarios con perfil completo

resultados de testing de algoritmos

## *Requerimientos no Funcionales*

### *Hardware*

Dado que el servicio que vamos a proveer va a estar alojado en servidores propios. El cliente no tiene ninguna responsabilidad antes estos apartados.

En cuanto al Hardware inicial necesario para la instalación del servicio, es necesario un Procesador de al menos 2 núcleos, 4 Gb de memoria RAM y con una capacidad para almacenar información de 500 Gb.

### *Software*

El cliente debe tener la capacidad/herramientas para brindar información de sus productos y usuarios.

El cliente debe tener la capacidad para modificar su software y de esta manera utilizar la api brindada.

Nuestro servicio va a correr en sistemas operativos basados en Linux. Ubuntu en su versión 14.04 es la distribución de Linux de preferencia.

Servidor Web:

Nginx (<http://nginx.org/en/>) junto con Unicorn (<http://unicorn.bogomips.org/>)

Bases de Datos:

PostgreSql: log de interacciones, información de items y usuarios, e información general del negocio.

Redis: utilizada para guardar el cache de recomendaciones

Neo4j: base de datos orientada a grafos utilizada para estructurar las recomendaciones

### *Seguridad*

La información almacenada de cada cliente es confidencial, por ello se debe autentificar al usuario y brindarle credenciales para realizar las consultas requeridas.

### *Rendimiento*

Es importante que el intercambio de información con el cliente, sea respondido a la brevedad por nuestro servicio. Entendiéndose a la brevedad como tiempo máximo de 1 seg.

***Concurrencia***

Es importante que se puedan procesar varias consultas al mismo tiempo

***Estándares***

El intercambio de información va a ser realizado utilizando el protocolo HTTP versión 1.1.

Los datos van a estar estructurados en formato JSON.

***Disponibilidad***

Se debe dar disponibilidad 24hs los 7 días de la semana, sin cortes por despliegue (zero-downtime deploys).

Ante la contingencia de no poder acceder a la base de datos primaria, se debe poder acceder a un backup de la misma para responder a las consultas necesarias, mientras que los datos a ser escritos en la base, se deben almacenar temporalmente fuera de la misma, para cuando se vuelva a tener acceso, sean escritos automáticamente.

***Idioma***

La calidad de las recomendaciones no dependerá del idioma, pero para el alcance del trabajo profesional podremos ofrecer recomendaciones de items cuyo lenguaje en el dominio e inglés o español.

# ***Herramientas de Desarrollo***

### *Hardware*

Se necesitará una computadora con un procesador de 2 núcleos, 4 Gb de memoria RAM y con una capacidad para almacenar información de 500 Gb.

### *Software*

Versionado de código fuente:

Git: Sistema de control de versiones de código fuente distribuido. Software de código abierto y libre.

Proveedor: BitBucket (<https://bitbucket.org/>)

Lenguajes de programación:

Ruby: Software de código abierto y libre (<https://www.ruby-lang.org/>)

Javascript

Frameworks:

Ruby on Rails: Software de código abierto y libre (<http://rubyonrails.org/>)

Sinatra: Software de código abierto y libre. Es un DSL para crear aplicaciones web (<http://www.sinatrarb.com/>)

Rendimiento:

Para medir el rendimiento de la aplicación utilizaremos ruby-prof (https://github.com/ruby-prof/ruby-prof).

La efectividad de las recomendaciones queda definida como cantidad de éxitos sobre la cantidad de recomendaciones del algoritmo en cuestión.

# ***Metodología***

[Pendiente]

# ***Cronograma***

[Pendiente]

Se realizaron estimaciones de esfuerzo y horas hombre en el apartado **WBS y estimaciones**

# ***Referencias***

[Pendiente]

# ***Curricululo/s Autor/es***

[Pendiente]

# ***Plan de Cursado***

[Pendiente]

Nota: aparte presentar acta acuerdo entre alumnos y tutores y carta de elevación al departamento de computación. Posibles modelos de estos documentos serían:

*[Pendiente]*

*Buenos Aires, x de M de 200z*

*Sr. Director*

*Del Departamento de Computación*

*De la Facultad de Ingeniería de la Universidad de Buenos Aires*

*Mg. Gustavo López*

*S.D.*

*De mi mayor consideración:*

*Por la presente me dirijo a Usted a fin de elevar a su consideración el acta donde consta mi conformidad y la del/de los estudiante/s AAAAAA (Padrón Nº #####), en la que se acuerda el tema de su Proyecto de Trabajo Profesional en Ingeniería Informática: “Nombre del trabajo”.*

*Se acompañan copias del Proyecto de Trabajo Profesional así como los datos del/de los Sr./Srs. AAAAAAAA.*

*Atentamente*

*Firma de/de los Profesor/es responsables*

*ACTA*

*En Buenos Aires al X día del mes de M del año dos mil X se reúnen en el Departamento de Computación el/la profesor/a Nombre del profesor y el/los estudiante/estudiantes de la carrera de Ingeniería en Informática Sr./Srs. AAAAAAAA (Padrón Nº #####) para acordar el Tema de Trabajo profesional para el Ciclo Superior de la Carrera de /del Sr./Srs. AAAAAAAA.*

*Luego de haber conversado sobre las áreas de interés del/de los estudiante/estudiantes y habiendo propuesto el/la profesor/a una lista posible de temas, se acuerda establecer como Tema de Trabajo Profesional en Ingeniería Informática de/del Sr./Srs. AAAAAAAA: “Nombre del trabajo”.*

*El/La nombre del profesor deja constancia que en su opinión, al haber elegido el/los Sr./Srs. AAAAAAAA las asignaturas de la Orientación, de colocar la orientación de la Carrera de Ingeniería en Informática y sus correspondientes electivas, se puede dar por cumplida la elaboración del Plan de Estudio Personal. Deja constancia que los contenidos de las asignaturas de la orientación abarcan los conocimientos necesarios para desarrollar satisfactoriamente el trabajo profesional.*

*Sin más que tratar, se da por concluida la reunión firmándose tres ejemplares de la presente acta, uno para elevar a la Dirección del Departamento de Computación, otro para el/los estudiante/s y el tercero para el/la profesor/a.*

*Sr./Srs. nombres*

*Firma de profesor responsable*

*Nota: Al momento de presentar la propuesta ante el Departamento de Computación, se deberá entregar un ejemplar de la misma en papel y también enviar toda la documentación, en formato digital y debidamente identificada con los apellidos de los alumnos presentantes, a la siguiente dirección:*

*trabajosytesisfiuba@gmail.com*

**WBS y estimaciónes**

**()** Definicion de pesos de esfuerzo: 1, 2, 3, 5, 8, 13, 21, 34, 55 siendo 1 el menor

**()** Estimación: en hs

**Total esfuerzo: 234**

**Total horas hombre: 806**

**Carga y actualización de items (13) (43)**

CRUD de items

Validacion de FeatureValues

(ver si se agrega carga manual/posteo de csv)

**Carga y actualización de características de items (13) (43)**

CRUD de Features

CRUD de Category

Creacion de FeatureDomains

Creacion FeatureTypes

Creacion de GlobalCategory

Mapeo de categorías de redes sociales con GlobalCategory

(ver si se agrega carga manual/posteo de csv)

**Carga y actualización de tipos de interacciones (1) (10)**

CRUD de InteractionType

(ver si se agrega carga manual)

**Carga y actualización de usuarios (3) (16)**

CRUD de user

Registro de información de redes sociales

(ver si se agrega carga manual/posteo de csv)

**Carga de interacciones usuario - item (1) (10)**

CRUD de Usuario - ítem (Se especifica si la interacción con el item viene o no de una recomendación exitosa)

**Carga de interacciones usuario - red social ítem (5) (17)**

CRUD de interacción

Información de categorías en redes sociales

**Carga de interacciones usuario - usuario (2) (13)**

- CRUD de interacciones de usuario con usuario.

- CRUD de información de relaciones con otros usuario en las redes sociales.

**Carga de interacciones usuario (unilateral) (1) (10)**

- CRUD de interacciones de usuario

**Recomendación de items (5) (17)**

Proveer un la api un método para consultar recomendaciones para un determinado usuario.

La recomendación devuelve el Id del item y un token para corroborar si el éxito de la recomendación es válido.

Recomendación con Testing A/B

**Algoritmos de recomendación (55) (181)**

Algoritmo de recomendación basado en la información del usuario/contexto

Algoritmo de recomendación basado en la interacción del usuario

Algoritmo de recomendación basado en personas de usuarios con los que interaccionó

Algoritmo de recomendación basado en personas de perfil similar

Reconocimiento del contexto del usuario para utilizar el algoritmo más adecuado

**Identificación de perfiles (13)(43)**

Armado de perfiles de los usuarios:

Perfiles colaborativos

Perfiles internos a un cliente (el cual contiene más información)

**Autentificación del cliente (8) (27)**

Creacion de roles

Autentificación

Autorización

Renovacion de credenciales

**Visualización de información (8) (27)**

Se deben mostrar graficos representativos de las principales métricas, pudiendo seleccionar el periodo deseado.

Cantidad de usuarios e ítems que se tienen

Cantidad de usuarios por región

Cantidad de ítems que se tienen por cada categoría

Cantidad de interacciones por recomendación.

Cantidad de recomendaciones por producto.

Cantidad de interacciones por usuario

**Capacitación/Investigación (34)(112)**

Ruby

Neo4J

PostgreSql

Typescript

Teoria de servicio de recomendaciones

Api de facebook, google+ y twitter

**Performance (17)(56)**

Mejora de performance en las recomendaciones

**Set de datos (8)(27)**

Preparación de sets de datos:

Items

Features

FeatureTypes

Category

GlobalCategory

Users

InteractionsTypes

UserInteractions with users

UserInteractions with items

UserInteractions unilateral

Interacciones con recomendaciones

**Administración (25%)(154)**

Manejo de proyecto

Análisis

Preparación de documentos

***Glosario:***

Item: Es un objeto en el modelo de negocio del cliente, compuesto por características, con el cual los usuarios pueden interactuar. Es lo que nuestro servicio quiere recomendar.

Usuario: Rol del lado del modelo de negocio del cliente que identifica a las personas que interactúan con los items.

Característica: Propiedad del item que describe cierto rasgo del mismo.

Items equivalentes: Dos ítems son equivalentes cuando todos los valores de las características de tipo Determinant, tengan los mismo valores.