



UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática



**TFG del Grado en Ingeniería
Informática**

**Sistema de Recomendación
basado en Aprendizaje
Profundo
Documentación Técnica**



Presentado por Raúl Negro Carpintero
en Universidad de Burgos — 1 de julio
de 2019

Tutor: Bruno Baruque Zanón

Índice general

Índice general	I
Índice de figuras	III
Índice de tablas	IV
Apéndice A Plan de Proyecto Software	1
A.1. Introducción	1
A.2. Planificación temporal	2
A.3. Estudio de viabilidad	4
Apéndice B Especificación de Requisitos	5
B.1. Introducción	5
B.2. Objetivos generales	5
B.3. Catalogo de requisitos	5
B.4. Especificación de requisitos	7
Apéndice C Especificación de diseño	21
C.1. Introducción	21
C.2. Diseño de datos	21
C.3. Diseño procedimental	22
C.4. Diseño arquitectónico	24
C.5. Diseño de la interfaz	31
Apéndice D Documentación técnica de programación	33
D.1. Introducción	33
D.2. Estructura de directorios	33

D.3. Manual del programador	34
D.4. Compilación, instalación y ejecución del proyecto	35
D.5. Pruebas del sistema	38
Apéndice E Documentación de usuario	39
E.1. Introducción	39
E.2. Requisitos de usuarios	39
E.3. Instalación	39
E.4. Manual del usuario	39
Bibliografía	41

Índice de figuras

B.1. Diagrama en caso de obtener un nuevo modelo	7
B.2. Diagrama en caso de obtener los resultados y las métricas . . .	8
B.3. Diagrama en caso de querer añadir valoraciones	8
B.4. Diagrama en caso de mostrar la ayuda	9
C.1. Diagrama de secuencia para obtener un nuevo modelo	23
C.2. Diagrama de secuencia para obtener métricas y predicciones . .	23
C.3. Diagrama de secuencia para añadir valoraciones	24
C.4. Esquema del patrón MVC [2]	25
C.5. Diagrama UML del proyecto	26
C.6. Diagrama UML del paquete <i>vista</i>	27
C.7. Diagrama UML del paquete <i>controlador</i>	28
C.8. Diagrama UML del paquete <i>modelo</i>	29
C.9. Interfaz web	32
D.1. Descargar proyecto	36
D.2. Navegación hasta <i>src</i>	37
D.3. Ejecución del proyecto	37
D.4. Interfaz web de la aplicación	38

Índice de tablas

B.1. CU-01 Gestión de los datos intermedios	10
B.2. CU-02 Guardar matrices	10
B.3. CU-3 Cargar matrices	11
B.4. CU-4 Gestión de los modelos	11
B.5. CU-5 Mostrar modelos	12
B.6. CU-6 Seleccionar modelos	12
B.7. CU-7 Guardar modelos	13
B.8. CU-8 Cargar modelos	13
B.9. CU-9 Gestión de los resultados	14
B.10.CU-10 Guardar resultados	14
B.11.CU-11 Mostrar resultados	15
B.12.CU-12 Mostrar predicciones	15
B.13.CU-13 Gestión de los usuarios	16
B.14.CU-14 Añadir valoraciones	17
B.15.CU-15 Gestión de los conjuntos de datos	17
B.16.CU-16 Mostrar los conjuntos de datos	18
B.17.CU-17 Seleccionar el conjunto de datos	18
B.18.CU-18 Añadir conjunto de datos	19
B.19.CU-19 Ayuda de la aplicación	19
B.20.CU-20 Mostrar información sobre los modelos	20
B.21.CU-21 Mostrar información sobre los conjuntos de datos	20

Apéndice A

Plan de Proyecto Software

A.1. Introducción

La planificación es un punto fundamental en cualquier proyecto software. En ella, se tiene que estimar qué cosas hay que hacer y cuánto tiempo y dinero va a llevar terminarlás. De esta manera, se podrá comprobar si se llega a tiempo a los distintos plazos y así evitar retrasos, lo que supondría pérdidas de dinero y tiempo.

La planificación se puede dividir en dos partes clave:

- **Planificación temporal:** consiste en dividir el proyecto en etapas (*sprints*) con una duración de x días. En estas etapas hay que dejar claro qué cosas se quieren conseguir y cuánto tiempo o esfuerzo se va a dedicar a cada una de ellas. Los *sprints* no tienen por qué durar siempre lo mismo, se pueden ajustar a las necesidades de cada momento.
- **Estudio de viabilidad:** con este estudio se podrá ver si el proyecto puede seguir adelante (es viable) o si no, con lo que habría que tomar medidas para cambiar la situación. Se divide en:
 - **Viabilidad económica:** según la cual se deberá calcular por cuánto tiene que venderse el servicio para compensar el dinero, el tiempo y el esfuerzo puestos en el proyecto. Se incluye salario del personal y la compra de software de terceros necesario para la realización del proyecto.
 - **Viabilidad legal:** se estudia el licenciamiento del software que se va a utilizar para proceder dentro de la legalidad.

A.2. Planificación temporal

Para el desarrollo del proyecto se decidió seguir la metodología ágil *Scrum*. Al ser un proyecto académico con una sola persona trabajando en él, no se siguen a raja tabla todos los rasgos característicos de *Scrum*. Las características que se han seguido son:

- Desarrollo incremental en cada *sprint*.
- Cada *sprint* tiene una duración aproximada de dos semanas.
- Se realizan reuniones al final de cada *sprint* con el objetivo de repasar lo hecho y pensar en los objetivos del siguiente *sprint*.

A continuación se desarrolla el contenido de cada *sprint*.

Sprint 1 (- 13/12/2018)

Este *sprint* se corresponde con el milestone **Búsqueda de información inicial**. En él se estudia el capítulo dedicado a los sistemas de recomendación del libro *Mining of Massive Datasets* [7].

También se realiza el curso de *fast.ai* [1]. Además, se escoge la librería de *LightFM* y se obtiene un modelo inicial.

Sprint 2 (13/12/2018 - 18/01/2019)

Este *sprint* se corresponde con el milestone **Explorar modelo con LightFM**. En él se intenta comprender el funcionamiento de *LightFM* y se utiliza el conjunto de datos de *Movielens* con el modelo.

Sprint 3 (18/01/2019 - 16/02/2019)

Este *sprint* se corresponde con el milestone **Recomendación híbrida y por contenido con LightFM**. En él se obtienen las primeras versiones del modelo híbrido y por contenido de *LightFM*.

También se obtiene un modelo inicial con *PyTorch* y se empieza a pensar en las métricas que se usarán para evaluar los modelos.

Sprint 4 (16/02/2019 - 03/03/2019)

Este *sprint* se corresponde con el milestone **Trabajar en la documentación**. En él se empieza a ampliar los apartados de la documentación relativos a los conjuntos de datos utilizados.

Sprint 5 (03/03/2019 - 13/03/2019)

Este *sprint* se corresponde con el milestone **División de los datos en LightFM**. En él se dividen los conjuntos de datos en *train* y *test* y se obtienen de nuevo los modelos.

Sprint 6 (13/03/2019 - 28/03/2019)

Este *sprint* se corresponde con el milestone **Medidas de calidad LightFM**. En él se obtienen las métricas finales para los modelos de *LightFM*.

Además, se guardan las matrices y los modelos en archivos *pickle* como medida de persistencia.

Sprint 7 (28/03/2019 - 05/04/2019)

Este *sprint* se corresponde con el milestone **Últimos pasos con LightFM**. En él se intenta acabar todo lo relacionado con los modelos clásicos (tanto código como documentación).

Sprint 8 (05/04/2019 - 24/04/2019)

Este *sprint* se corresponde con el milestone **Ampliar documentación**. En él se amplía la documentación relativa a *LightFM* y se empiezan a obtener los primeros diagramas UML con la estructura del proyecto hasta el momento.

Sprint 9 (24/04/2019 - 08/05/2019)

Este *sprint* se corresponde con el milestone **Cambios en el diseño, Flask y DL**. En él se empieza a obtener la interfaz web con *Flask* y se escoge *Spotlight* como la librería de aprendizaje profundo.

También se modifica la documentación para ir acorde a los cambios del código.

Sprint 10 (08/05/2019 - 22/05/2019)

Este *sprint* se corresponde con el milestone **Primeros pasos con Spotlight**. En él se obtiene un primer modelo de *Spotlight* y se unifican los archivos de Entrada y Persistencia para que solo haya uno de cada.

Se continúa trabajando en la documentación.

Sprint 11 (22/05/2019 - 05/06/2019)

Este *sprint* se corresponde con el milestone **Completar modelos de Spotlight**. En él se obtienen el resto de modelos de *Spotlight* y se generalizan los métodos de lectura de datos y obtención de matrices.

Se continúa trabajando en la documentación.

Sprint 12 (05/06/2019 - 12/06/2019)

Este *sprint* se corresponde con el milestone **Diseño GUI**. En él se piensa qué forma va a tener la interfaz web obtenida a través de *Flask*.

Sprint 13 (12/06/2019 - 20/06/2019)

Este *sprint* se corresponde con el milestone **Obtención GUI**. En él se termina de crear la interfaz web y se empiezan a emplear los métodos obtenidos anteriormente para que la aplicación sea funcional.

Sprint 14 (20/06/2019 - 03/07/2019)

Este *sprint* se corresponde con el milestone **Último milestone**. En él se termina todo lo que queda por hacer en el proyecto. La interfaz es totalmente funcional y la documentación está terminada.

A.3. Estudio de viabilidad

Viabilidad económica

Viabilidad legal

Apéndice B

Especificación de Requisitos

B.1. Introducción

Este anexo recoge los objetivos generales y la especificación de requisitos del proyecto.

B.2. Objetivos generales

El proyecto persigue los siguientes objetivos generales:

- Comprender los sistemas de recomendación tanto clásicos como basados en aprendizaje profundo.
- Recoger y evaluar los resultados obtenidos por los dos modelos sobre diferentes conjuntos de datos.
- Comparar los resultados.

B.3. Catalogo de requisitos

Los requisitos derivados de los objetivos del proyecto son los siguientes:

Requisitos funcionales

- **RF-1 Gestión de los datos intermedios:** el programa tiene que ser capaz de gestionar los datos intermedios:

- **RF-1.1 Guardar matrices:** el programa tiene que ser capaz de guardar las matrices generadas por los modelos para ahorrar tiempo en siguientes ejecuciones.
- **RF-1.2 Cargar matrices:** el programa tiene que ser capaz de cargar las matrices que el usuario quiera.
- **RF-2 Gestión de los modelos:** el programa tiene que ser capaz de gestionar los modelos:
 - **RF-2.1 Mostrar modelos:** el programa tiene que ser capaz de mostrar todos los tipos de modelos disponibles.
 - **RF-2.2 Seleccionar modelos:** el programa tiene que ser capaz de dejar al usuario seleccionar el tipo de modelo que quiera.
 - **RF-2.3 Guardar modelos:** el programa tiene que ser capaz de guardar los modelos obtenidos para ahorrar tiempo en futuras ejecuciones.
 - **RF-2.4 Cargar modelos:** el programa tiene que ser capaz de cargar los modelos previamente guardados.
- **RF-3 Gestión de los resultados:** el programa tiene que ser capaz de gestionar los resultados:
 - **RF-3.1 Guardar resultados:** el programa tiene que ser capaz de guardar los resultados generados por los modelos.
 - **RF-3.2 Mostrar resultados:** el programa tiene que ser capaz de mostrar los resultados obtenidos por los distintos modelos.
 - **RF-3.3 Mostrar predicciones:** el programa tiene que ser capaz de mostrar las predicciones obtenidas por los distintos modelos.
- **RF-4 Gestión de los usuarios:** el programa tiene que ser capaz de introducir las valoraciones de nuevos usuarios:
 - **RF-4.1 Añadir valoraciones:** el programa tiene que ser capaz de simular la entrada de nuevos usuarios permitiendo el añadido de nuevas valoraciones.
- **RF-5 Gestión de los conjuntos de datos:** el programa tiene que:
 - **RF-5.1 Mostrar los conjuntos de datos:** el programa tiene que ser capaz de mostrar los distintos conjuntos de datos de prueba.
 - **RF-5.2 Seleccionar el conjunto de datos:** el programa tiene que ser capaz de mostrar dejar que el usuario seleccione uno de los conjuntos de datos de prueba.
 - **RF-5.3 Añadir conjunto de datos:** el programa tiene que ser capaz de dejar que el usuario añada un conjunto de datos propio.

- **RF-6 Ayuda de la aplicación:** el programa tiene que ofrecer información al usuario:
 - **RF-6.1 Mostrar información sobre modelos:** el programa tiene que ser capaz de mostrar información sobre los distintos modelos que se pueden escoger.
 - **RF-6.2 Mostrar información sobre conjuntos de datos:** el programa tiene que ser capaz de mostrar información sobre los distintos conjuntos de datos de prueba.

Requisitos no funcionales

- **RNF-1 Usabilidad:** la interfaz gráfica tiene que ser intuitiva y fácil de usar.
- **RNF-2 Soporte:** el programa tiene que dar soporte a versiones iguales o mayores a Python 3.
- **RNF-3 Localización:** el programa tiene que estar preparado para soportar varios idiomas.

B.4. Especificación de requisitos

En esta sección se mostrará el diagrama de casos de uso y se desarrollará cada uno de ellos.

Diagrama de casos de uso

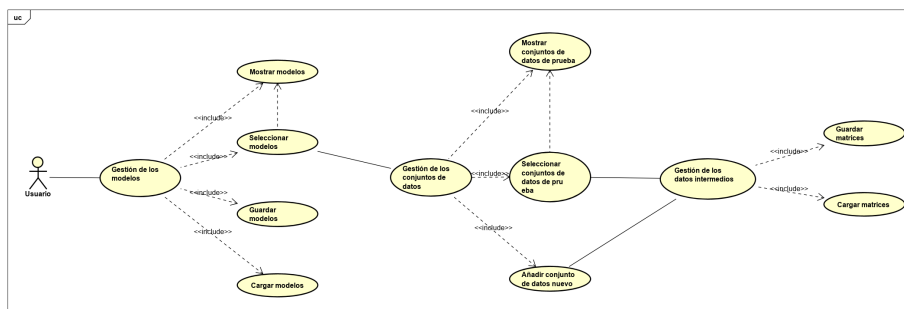


Figura B.1: Diagrama en caso de obtener un nuevo modelo

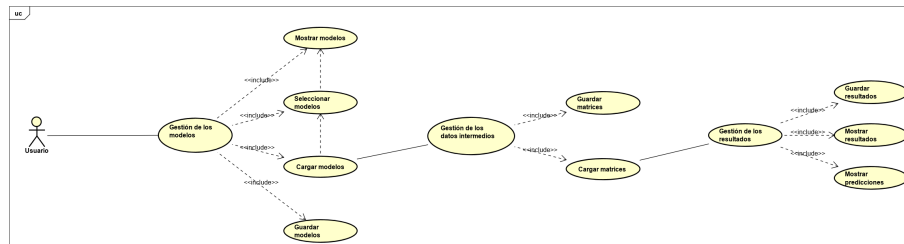


Figura B.2: Diagrama en caso de obtener los resultados y las métricas

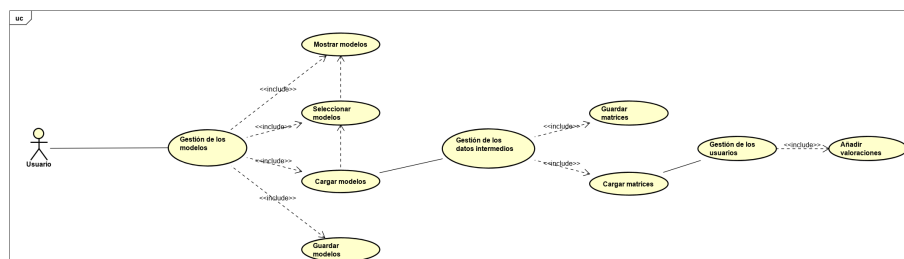


Figura B.3: Diagrama en caso de querer añadir valoraciones

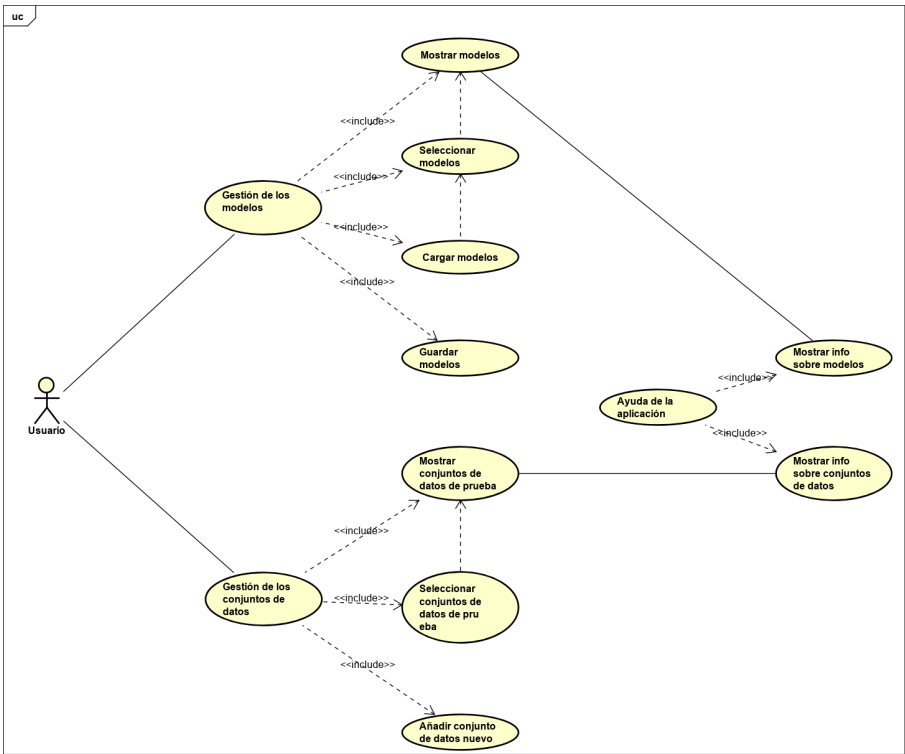


Figura B.4: Diagrama en caso de mostrar la ayuda

Actores

Con la aplicación solo interactuará un actor, el usuario que esté probando la aplicación en un momento determinado.

Casos de uso

A continuación, se desarrollará cada caso de uso:

CU-01	Gestión de los datos intermedios
Versión	1.0

continúa en la página siguiente

continúa desde la página anterior

CU-01	Gestión de los datos intermedios
Autor	Raúl Negro Carpintero
Requisitos asociados	RF-1 & RF-1.1 & RF-1.2
Descripción	Gestión de los datos intermedios.
Precondiciones	Los datos intermedios tienen que existir.
Acciones	El usuario escoge los datos que quiere cargar o guardar.
Postcondiciones	Los datos intermedios se cargan o se guardan.
Excepciones	Los datos intermedios no existen.

Tabla B.1: CU-01 Gestión de los datos intermedios

CU-02	Guardar matrices
Versión	1.0
Autor	Raúl Negro Carpintero
Requisitos asociados	RF-1 & RF-1.1
Descripción	Guardar las matrices de datos.
Precondiciones	Tienen que existir las matrices.
Acciones	El usuario guarda las matrices generadas por el sistema.
Postcondiciones	Las matrices se guardan en archivos <i>.pickle</i>
Excepciones	Los matrices no existen.

Tabla B.2: CU-02 Guardar matrices

CU-03	Cargar matrices
Versión	1.0
Autor	Raúl Negro Carpintero
Requisitos asociados	RF-1 & RF-1.2

continúa en la página siguiente

continúa desde la página anterior

CU-03	Cargar matrices
Descripción	Cargar las matrices de datos.
Precondiciones	Tienen que existir los archivos <i>.pickle</i> con las matrices.
Acciones	El usuario carga las matrices.
Postcondiciones	Las matrices se cargan y se vinculan al sistema.
Excepciones	No hay archivos <i>.pickle</i> con las matrices.

Tabla B.3: CU-3 Cargar matrices

CU-04	Gestión de los modelos
Versión	1.0
Autor	Raúl Negro Carpintero
Requisitos asociados	RF-2 & RF-2.1 & RF-2.2 & RF-2.3 & RF-2.4
Descripción	Gestión de los modelos.
Precondiciones	Se tienen que haber generado los modelos.
Acciones	El usuario escoge el tipo de modelo que quiere obtener, guardar o cargar.
Postcondiciones	El modelo escogido se crea, guarda o carga.
Excepciones	No hay modelos generados.

Tabla B.4: CU-4 Gestión de los modelos

CU-05	Mostrar modelos
Versión	1.0
Autor	Raúl Negro Carpintero
Requisitos asociados	RF-2 & RF-2.1
Descripción	Se listan todos los tipos de modelos que se pueden obtener.

continúa en la página siguiente

continúa desde la página anterior

CU-05	Mostrar modelos
Precondiciones	-
Acciones	El usuario piensa que tipo de modelo utilizar.
Postcondiciones	-
Excepciones	No hay tipos de modelos que mostrar.

Tabla B.5: CU-5 Mostrar modelos

CU-06	Seleccionar modelos
Versión	1.0
Autor	Raúl Negro Carpintero
Requisitos asociados	RF-2 & RF-2.2
Descripción	Se selecciona un modelo.
Precondiciones	Se han mostrado todos los tipos de modelos.
Acciones	El usuario escoge el tipo de modelo que quiere crear, guardar o cargar.
Postcondiciones	El tipo de modelo que se quiere utilizar se queda seleccionado para trabajar con él.
Excepciones	No existe el modelo seleccionado.

Tabla B.6: CU-6 Seleccionar modelos

CU-07	Guardar modelos
Versión	1.0
Autor	Raúl Negro Carpintero
Requisitos asociados	RF-2 & RF-2.3
Descripción	Se guarda un modelo.
Precondiciones	El modelo que se quiere guardar debe existir.

continúa en la página siguiente

continúa desde la página anterior

CU-07	Guardar modelos
Acciones	Se guarda el modelo en un archivo <i>.pickle</i> .
Postcondiciones	Se obtiene un archivo <i>.pickle</i> con el modelo.
Excepciones	No hay ningún modelo que guardar.

Tabla B.7: CU-7 Guardar modelos

CU-08	Cargar modelos
Versión	1.0
Autor	Raúl Negro Carpintero
Requisitos asociados	RF-2 & RF-2.4
Descripción	Se carga un modelo.
Precondiciones	El archivo <i>.pickle</i> con el modelo que se quiere cargar debe existir.
Acciones	Se carga el modelo desde un archivo <i>.pickle</i> .
Postcondiciones	Se obtiene el modelo.
Excepciones	No hay ningún archivo <i>.pickle</i> con el modelo deseado.

Tabla B.8: CU-8 Cargar modelos

CU-09	Gestión de los resultados
Versión	1.0
Autor	Raúl Negro Carpintero
Requisitos asociados	RF-3 & RF-3.1 & RF-3.2 & RF-3.3
Descripción	Gestión de los resultados.
Precondiciones	El modelo tiene que estar creado. Los resultados tienen que haber sido calculados.

continúa en la página siguiente

continúa desde la página anterior

CU-09	Gestión de los resultados
Acciones	Tienen que existir los resultados. Las predicciones tiene que haber sido calculadas. Guardar los resultados. Mostrar los resultados.
Postcondiciones	Mostrar las predicciones. Se cargan los resultados. Se muestran los resultados.
Excepciones	Se muestran las predicciones. Los resultados no se han calculado. Las predicciones no se han calculado. No hay ningún modelo.

Tabla B.9: CU-9 Gestión de los resultados

CU-10	Guardar resultados
Versión	1.0
Autor	Raúl Negro Carpintero
Requisitos asociados	RF-3 & RF-3.1
Descripción	Guardar resultados.
Precondiciones	Los resultados tienen que haber sido calculados. El modelo tiene que estar creado.
Acciones	Guardar los resultados.
Postcondiciones	Se guardan los resultados.
Excepciones	Los resultados no se han calculado. No hay ningún modelo.

Tabla B.10: CU-10 Guardar resultados

CU-11	Mostrar resultados
-------	--------------------

continúa en la página siguiente

continúa desde la página anterior

CU-11	Mostrar resultados
Versión	1.0
Autor	Raúl Negro Carpintero
Requisitos asociados	RF-3 & RF-3.2
Descripción	Mostrar los resultados.
Precondiciones	Los resultados tienen que haber sido calculados. El modelo tiene que estar creado.
Acciones	Mostrar los resultados.
Postcondiciones	Se muestran los resultados.
Excepciones	Los resultados no se han calculado. No hay ningún modelo.

Tabla B.11: CU-11 Mostrar resultados

CU-12	Mostrar predicciones
Versión	1.0
Autor	Raúl Negro Carpintero
Requisitos asociados	RF-3 & RF-3.3
Descripción	Mostrar las predicciones.
Precondiciones	El modelo tiene que estar creado. Las predicciones tiene que haber sido calculadas.
Acciones	Mostrar las predicciones.
Postcondiciones	Se muestran las predicciones.
Excepciones	Las predicciones no se han calculado. No hay ningún modelo.

Tabla B.12: CU-12 Mostrar predicciones

CU-13	Gestión de los usuarios
-------	-------------------------

continúa en la página siguiente

continúa desde la página anterior

CU-13		Gestión de los usuarios
Versión		1.0
Autor		Raúl Negro Carpintero
Requisitos asociados	asocia-	RF-4 & RF-4.1
Descripción		Gestión de los usuarios.
Precondiciones		Deben existir las matrices de datos. Debe existir el usuario con el que se quieren añadir valoraciones.
Acciones		Añadir valoraciones.
Postcondiciones		Se añaden valoraciones y/o usuarios a un conjunto de datos.
Excepciones		No existen las matrices de datos. No existe el usuario al que se le quiere añadir valoraciones.

Tabla B.13: CU-13 Gestión de los usuarios

CU-14		Añadir valoraciones
Versión		1.0
Autor		Raúl Negro Carpintero
Requisitos asociados	asocia-	RF-4 & RF-4.1
Descripción		Añadir valoraciones de un usuario existente o nuevo.
Precondiciones		Deben existir las matrices de datos. Debe existir el usuario con el que se quieren añadir valoraciones.
Acciones		Añadir valoraciones sobre un usuario nuevo o uno ya existente.
Postcondiciones		Se añaden valoraciones y/o usuarios a un conjunto de datos.

continúa en la página siguiente

continúa desde la página anterior

CU-14	Añadir valoraciones
Excepciones	No existen las matrices de datos. No existe el usuario al que se le quiere añadir valoraciones.

Tabla B.14: CU-14 Añadir valoraciones

CU-15	Gestión de los conjuntos de datos
Versión	1.0
Autor	Raúl Negro Carpintero
Requisitos asociados	RF-5 & RF-5.1 & RF-5.2 & RF-5.3
Descripción	Gestión de los conjuntos de datos.
Precondiciones	Tienen que existir los conjuntos de datos de prueba. Tiene que existir un conjunto de datos para usar.
Acciones	Listar los conjuntos de datos de prueba. Seleccionar un conjunto de datos de prueba. Añadir conjunto de datos nuevo.
Postcondiciones	Se carga el conjunto de datos de prueba seleccionado. Se añade el conjunto de datos nuevo.
Excepciones	No existen conjuntos de datos de prueba. No existe ningún conjunto de datos que añadir.

Tabla B.15: CU-15 Gestión de los conjuntos de datos

CU-16	Mostrar los conjuntos de datos
Versión	1.0
Autor	Raúl Negro Carpintero
Requisitos asociados	RF-5 & RF-5.1

continúa en la página siguiente

continúa desde la página anterior

CU-16	Mostrar los conjuntos de datos
Descripción	Listar los conjuntos de datos de prueba.
Precondiciones	Tienen que existir los conjuntos de datos de prueba.
Acciones	Listar los conjuntos de datos de prueba.
Postcondiciones	Se selecciona un conjunto de datos de prueba.
Excepciones	No existen conjuntos de datos de prueba.

Tabla B.16: CU-16 Mostrar los conjuntos de datos

CU-17	Seleccionar el conjunto de datos
Versión	1.0
Autor	Raúl Negro Carpintero
Requisitos asociados	RF-5 & RF-5.2
Descripción	Seleccionar un conjunto de datos de prueba.
Precondiciones	Tienen que existir los conjuntos de datos de prueba.
Acciones	Seleccionar un conjunto de datos de prueba.
Postcondiciones	Se carga el conjunto de datos de prueba seleccionado.
Excepciones	No existen conjuntos de datos de prueba. No hay ningún conjunto de datos de prueba seleccionado.

Tabla B.17: CU-17 Seleccionar el conjunto de datos

CU-18	Añadir conjunto de datos
Versión	1.0
Autor	Raúl Negro Carpintero
Requisitos asociados	RF-5 & RF-5.3
Descripción	Añadir nuevo conjunto de datos.
Precondiciones	-

continúa en la página siguiente

continúa desde la página anterior

CU-18	Añadir conjunto de datos
Acciones	Añadir conjunto de datos nuevo.
Postcondiciones	Se añade el nuevo conjunto de datos.
Excepciones	No existe ningún conjunto de datos que añadir.

Tabla B.18: CU-18 Añadir conjunto de datos

CU-19	Ayuda de la aplicación
Versión	1.0
Autor	Raúl Negro Carpintero
Requisitos asociados	RF-6 & RF-6.1 & RF-6.2
Descripción	Ayuda de la aplicación.
Precondiciones	Tiene que haber información sobre los modelos. Tiene que haber información sobre los conjuntos de datos de prueba.
Acciones	Mostrar información sobre los modelos. Mostrar información sobre los conjuntos de datos de prueba.
Postcondiciones	Se muestra la información sobre los modelos. Se muestra la información sobre los conjuntos de datos de prueba.
Excepciones	No existe información sobre los modelos. No existe información sobre los conjuntos de datos de prueba.

Tabla B.19: CU-19 Ayuda de la aplicación

CU-20	Mostrar información sobre los modelos
Versión	1.0

continúa en la página siguiente

continúa desde la página anterior

CU-20		Mostrar información sobre los modelos
Autor		Raúl Negro Carpintero
Requisitos asociados		RF-6 & RF-6.1
Descripción		Mostrar información sobre los modelos.
Precondiciones		Tiene que haber información sobre los modelos.
Acciones		Mostrar información sobre los modelos.
Postcondiciones		Se muestra la información sobre los modelos.
Excepciones		No existe información sobre los modelos.

Tabla B.20: CU-20 Mostrar información sobre los modelos

CU-21		Mostrar información sobre los conjuntos de datos
Versión		1.0
Autor		Raúl Negro Carpintero
Requisitos asociados		RF-6 & RF-6.2
Descripción		Mostrar información sobre los conjuntos de datos de prueba.
Precondiciones		Tiene que haber información sobre los conjuntos de datos de prueba.
Acciones		Mostrar información sobre los conjuntos de datos de prueba.
Postcondiciones		Se muestra la información sobre los conjuntos de datos de prueba.
Excepciones		No existe información sobre los conjuntos de datos de prueba.

Tabla B.21: CU-21 Mostrar información sobre los conjuntos de datos

Apéndice C

Especificación de diseño

C.1. Introducción

En este apéndice se explica cómo están conformados los datos que utilizan las librerías usadas en el proyecto, así como la forma en la que está estructurado el mismo.

C.2. Diseño de datos

Todos los datos que se han utilizado a lo largo del proyecto están en formato .csv. Lo más normal es que para cada conjunto de datos se tengan los siguientes archivos:

- *ratings.csv*
- *users.csv*
- *items.csv*

La estructura de estos archivos suele ser: *idUser*, *idItem*, *rating*, *timestamp* para *ratings.csv*, *idUser*, *name*, *feature1*, ..., *featureN* para *users.csv* y *idItem*, *name*, *feature1*, ..., *featureN* para *items.csv*.

Para poder trabajar con los datos primero se pasan a *DataFrames* de *pandas* [8].

Datos con LightFM

Una vez obtenidos los *DataFrames* para cada *.csv*, es necesario convertirlos a *Dataset* de *LightFM* [3] para poder trabajar con ellos.

Esta clase se encarga de convertir los datos almacenados en los *DataFrames* en *matrices COO* y *matrices CSR*.

Datos con Spotlight

A diferencia de *LightFM*, en *Spotlight* no se trabaja con matrices dispersas, si no con arrays de *NumPy* (aunque se ofrece la posibilidad de transformar los arrays en *matrices COO* y *matrices CSR*).

Spotlight utiliza su propia clase, *Interactions* [4], para convertir los *DataFrames* y así poder utilizar los datos.

Otro aspecto a tener en cuenta en *Spotlight* son las secuencias, utilizadas en el modelo de secuencia implícito [6]. Las recomendaciones pueden verse como secuencias; dados los ítems con los que ha interactuado un usuario, ¿cuáles serán los próximos ítems con los que interactuará? De esta manera, una vez obtenidas las interacciones "normales", hay que pasarlas a interacciones de secuencia con el método `to_sequence()` [5].

Persistencia

Para la realización del proyecto se necesita guardar de alguna manera tanto los datos intermedios (matrices de interacción) como los propios modelos y los resultados obtenidos al evaluarlos.

Esto se consigue gracias a *pickle* [9].

C.3. Diseño procedimental

En este apartado se utilizan diagramas de secuencia para explicar el funcionamiento de la aplicación.

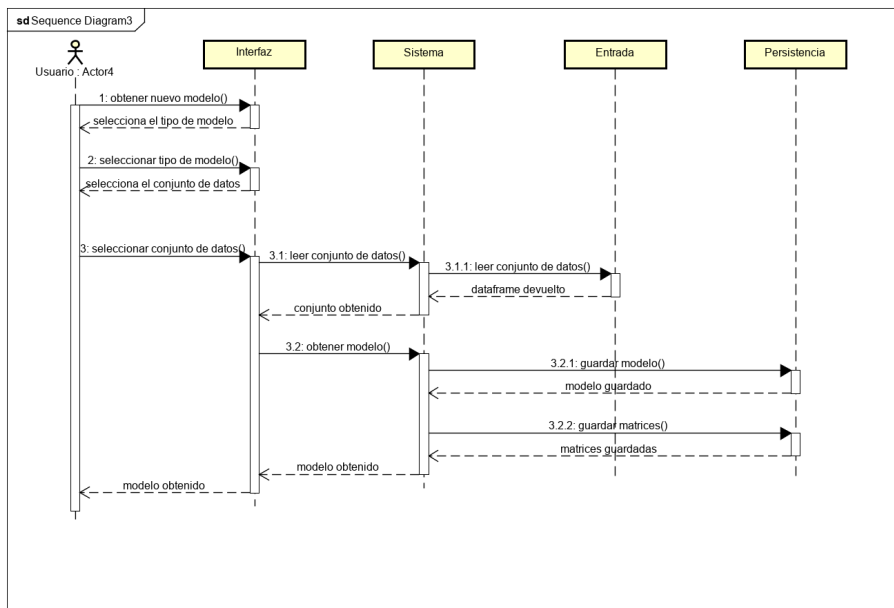


Figura C.1: Diagrama de secuencia para obtener un nuevo modelo

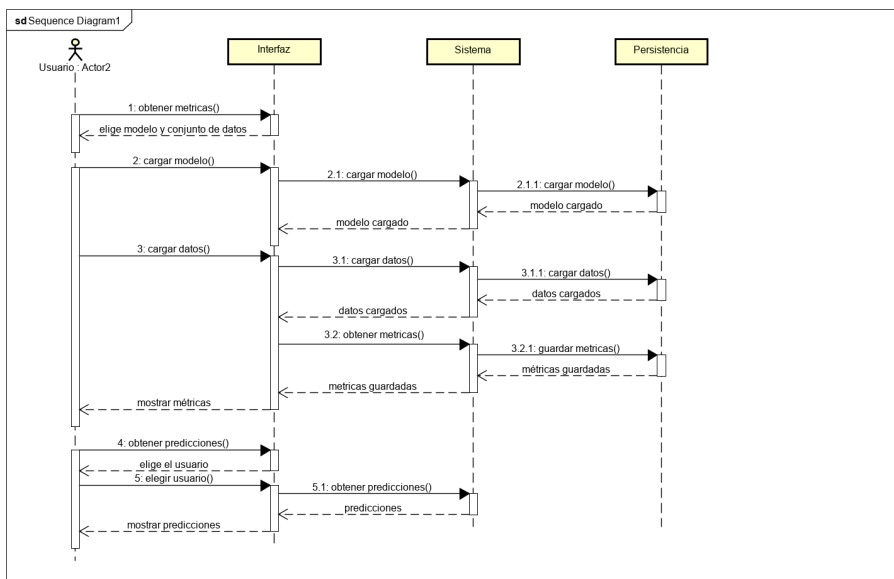


Figura C.2: Diagrama de secuencia para obtener métricas y predicciones

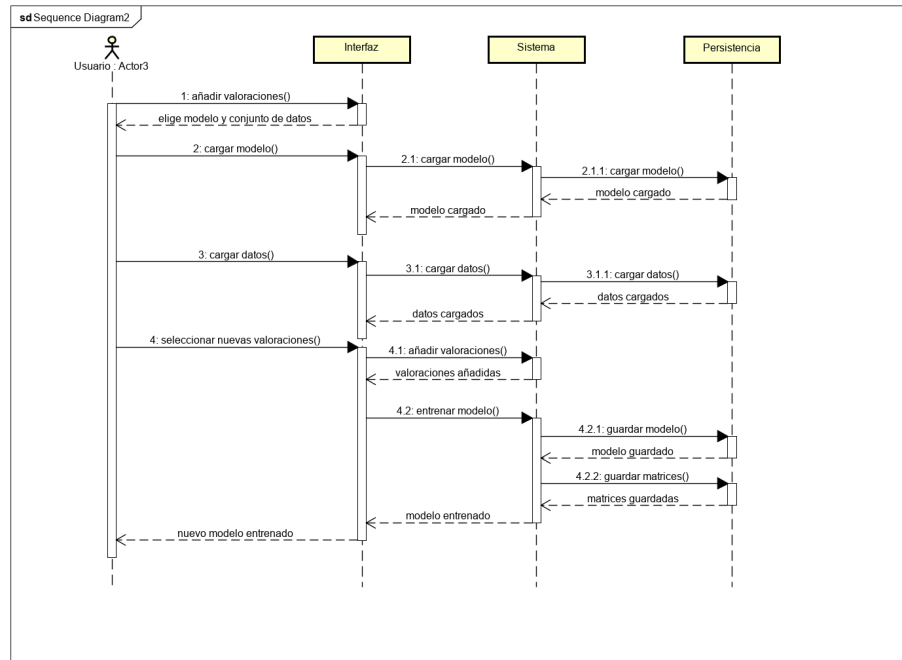


Figura C.3: Diagrama de secuencia para añadir valoraciones

C.4. Diseño arquitectónico

Para la realización de este proyecto se ha seguido el patrón arquitectónico MVC (*Modelo Vista Controlador*). El objetivo de este patrón es dividir el código en función de su propósito. Sus partes son:

- *Modelo*: el acceso a los datos. Se corresponde con las clases de Entrada, Salida y Persistencia; que leen los datos para dárselo al sistema de recomendación y guardan los resultados.
- *Vista*: la visualización de los datos. Se corresponde con las clases de Interfaz, Flask y Forms, que muestran la información solicitada.
- *Controlador*: la manipulación de los datos. Se corresponde con los clases de Sistema, que crean los sistemas de recomendación.

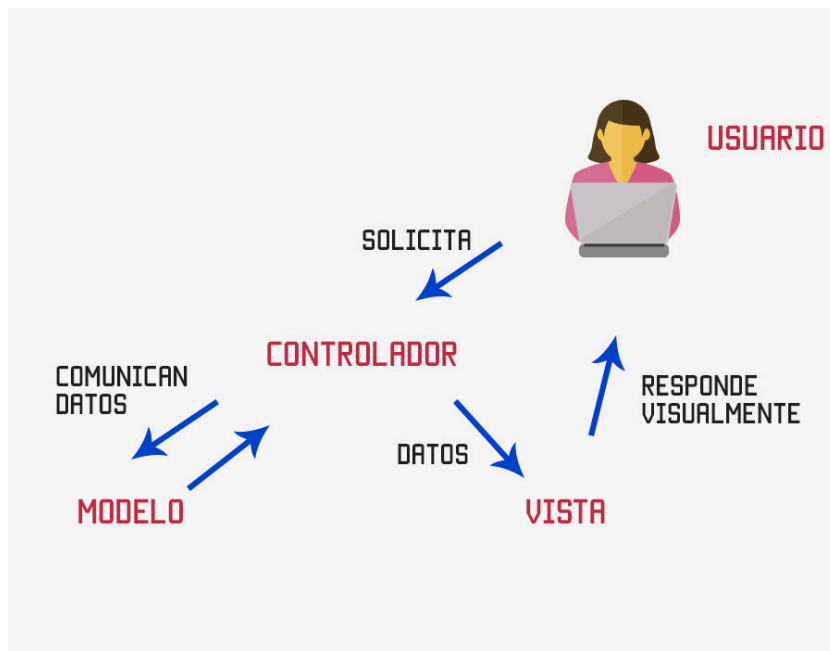


Figura C.4: Esquema del patrón MVC [2]

La estructura del proyecto siguiendo este patrón quedaría de la siguiente forma:

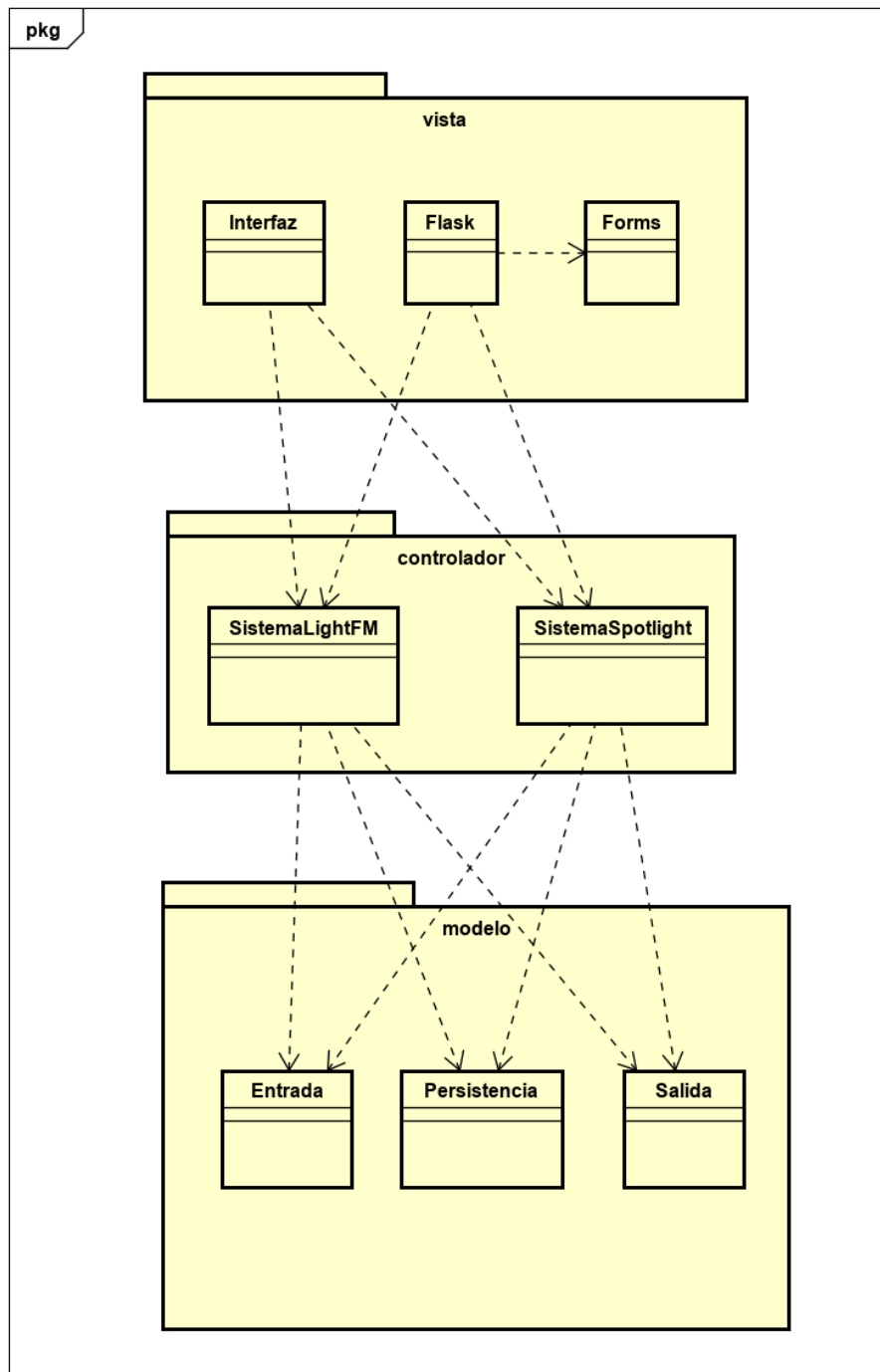


Figura C.5: Diagrama UML del proyecto

Por separado, los paquetes contienen: NO ESTÁN ACTUALIZADOS!!

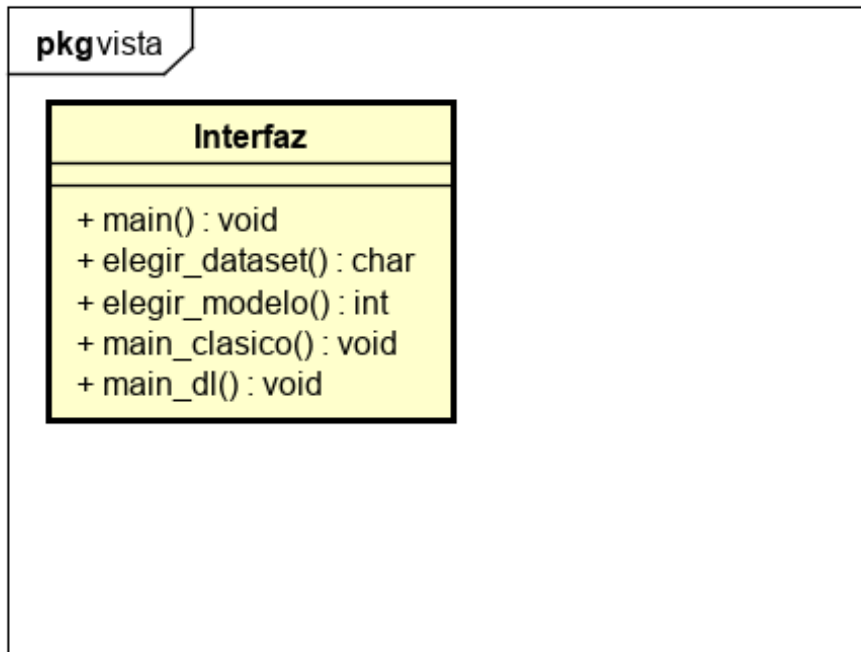
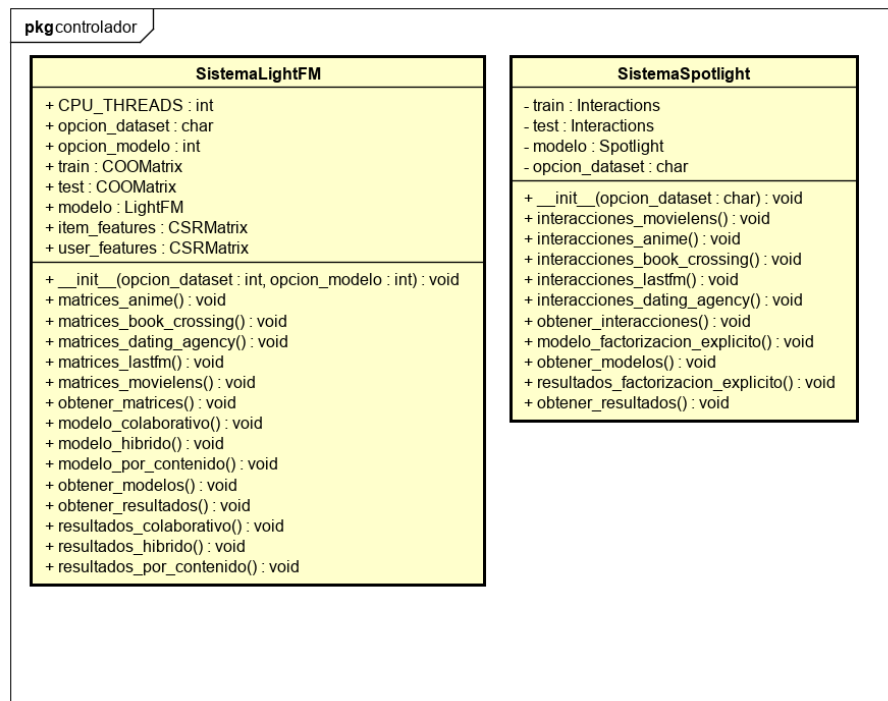


Figura C.6: Diagrama UML del paquete *vista*

Figura C.7: Diagrama UML del paquete *controlador*

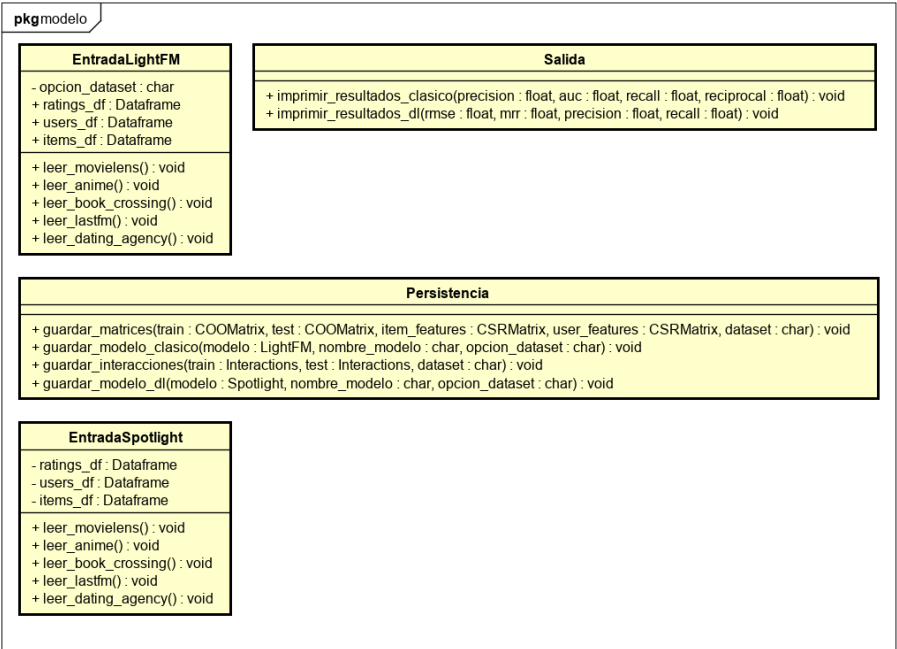


Figura C.8: Diagrama UML del paquete *modelo*

Diseño con LightFM

NO ESTÁ ACTUALIZADO!! Para la parte del modelo clásico, se tienen los siguientes archivos: En la clase `EntradaLightFM` se tiene:

- `leer_x`: estos métodos recogen los datos de los *.csv* para cada conjunto de datos. Estos métodos son utilizados por los métodos de obtención de matrices de la clase `SistemaLightFM`.

En el archivo `Salida` se tiene:

- `imprimir_resultados_clasico`: este método imprime las métricas del modelo clásico escogido.

En el archivo `Persistencia` se tiene:

- `guardar_matrices`: este método guarda en un archivo *pickle* las matrices que *LightFM* necesita para obtener los modelos.
- `guardar_modelo`: este método guarda el modelo obtenido por *LightFM* en un archivo *pickle*.

En el archivo `Interfaz` se tiene:

- `elegir_dataset`: este método muestra un menú mediante el cual elegimos un conjunto de datos que utilizar.
- `elegir_modelo`: este método muestra un menú mediante el cual elegimos un modelo concreto a crear.
- `main_clasico`: programa principal si el modelo escogido es *LightFM*.

En la clase `SistemaLightFM` tenemos:

- `matrices_x`: estos métodos crean las matrices necesarias para cada conjunto de datos.
- `modelo_x`: estos métodos crean los distintos modelos de recomendación.
- `resultados_x`: estos métodos obtienen los resultados para cada modelo de recomendación.

Diseño con Spotlight

NO ESTÁ ACTUALIZADO!! Para la parte del modelo basado en aprendizaje profundo, se tienen los siguientes archivos: En la clase `EntradaSpotlight` se tiene:

- `leer_x`: estos métodos recogen los datos de los *.csv* para cada conjunto de datos. Estos métodos son utilizados por los métodos de obtención de interacciones de la clase `SistemaSpotlight`.

En el archivo `Salida` se tiene:

- `imprimir_resultados_dl`: este método imprime las métricas del modelo basado en aprendizaje profundo escogido.

En el archivo `Persistencia` se tiene:

- `guardar_interacciones`: este método guarda en un archivo *pickle* las interacciones que *Spotlight* necesita para obtener los modelos.
- `guardar_modelo_dl`: este método guarda el modelo obtenido por *Spotlight*.

En el archivo `Interfaz` se tiene:

- `main_dl`: programa principal si el modelo escogido es *Spotlight*.

En la clase `SistemaSpotlight` tenemos:

- `interacciones_x`: estos métodos crean las interacciones necesarias para cada conjunto de datos.
- `modelo_x`: estos métodos crean los distintos modelos de recomendación.
- `resultados_x`: estos métodos obtienen los resultados para cada modelo de recomendación.

C.5. Diseño de la interfaz

A continuación se muestra el diseño que tendrá la interfaz web (es posible que la versión final cambie).

Apéndice *D*

Documentación técnica de programación

D.1. Introducción

En este apéndice se explica cómo está estructurado el proyecto en cuanto a los directorios, así como los pasos para compilar, instalar y ejecutar el programa.

D.2. Estructura de directorios

El proyecto está dividido en los siguientes directorios:

- **/:** carpeta raíz. Contiene la carpeta *docs*, la carpeta *src* y los ficheros *.gitignore* y *README.md*.
- **/docs:** contiene todos los archivos relacionados con la documentación tanto en \LaTeX como en *pdf*. También contiene los archivos de bibliografía y las imágenes que se utilizan a lo largo de la documentación.
- **/docs/img:** carpeta que contiene las imágenes necesarias para apoyar la documentación.
- **/docs/tex:** carpeta que contiene los distintos apartados de la memoria y los anexos en \LaTeX .
- **/src:** carpeta que contiene todo el código de la aplicación.
- **/src/controlador:** contiene los ficheros *.py* con los sistemas de *LightFM* y *Spotlight*.
- **/src/modelo:** contiene los ficheros *.py* correspondientes a la *Entrada*, *Salida* y *Persistencia* de los datos.

APÉNDICE D. DOCUMENTACIÓN TÉCNICA DE PROGRAMACIÓN

- **/src/vista:** contiene los ficheros *.py* correspondientes a la *Interfaz* (de texto), *Flask* (web) y *Forms* (formularios de los que se compone la interfaz web). Además, contiene los *.html* de la interfaz.
- **/src/vista/templates:** contiene las páginas *.html* que componen la interfaz web de la aplicación.
- **/src/uploads:** carpeta donde se guardan los modelos y las matrices obtenidas.
- **/src/notebooks:** esta carpeta contiene archivos que en su día se utilizaron de prueba. Se podrían borrar, pero se prefiere dejarlos para que haya constancia de que se trabajó en ellos.

D.3. Manual del programador

Las herramientas necesarias para la realización del proyecto han sido:

- Python 3.6
- Spyder o Sublime Text
- LightFM
- Spotlight
- Flask
- WTForms

A continuación se detalla como instalar estas herramientas:

Python

Lenguaje de programación escogido para llevar a cabo el proyecto. Se puede descargar la última versión desde el siguiente enlace: [Descargar Python](#).

IDE

Se puede utilizar el IDE que más guste. En este caso, se ha utilizado tanto *Spyder* como *Sublime Text*, además de los notebooks de *Jupyter*.

Spyder y *Jupyter* vienen dentro del paquete de *Anaconda* (que también incluye Python). [Guía para instalar Anaconda en Windows](#). En este otro enlace se descarga *Anaconda* directamente para cualquier sistema operativo: [Descargar Anaconda](#).

Sublime Text se puede obtener desde el siguiente enlace: [Descargar Sublime Text](#).

LightFM

Librería utilizada para la obtención de los modelos de recomendación clásicos. Para instalar la librería es recomendable seguir los pasos descritos en el siguiente enlace: [Instalación de LightFM](#).

Spotlight

Librería utilizada para la obtención de los modelos de recomendación basados en aprendizaje profundo. Para instalar *Spotlight* basta con ejecutar el siguiente comando: **conda install -c maciejkula -c pytorch spotlight=0.1.5** [?]. Como se puede observar, con este comando también instalamos *PyTorch*.

Flask

Con *Flask* se obtiene la interfaz web. Tan solo hay que ejecutar el comando **pip install Flask** para instalarlo [?].

WTForms

Esta librería se utiliza para poder generar todos los formularios que componen la interfaz web. Para instalarla vale con ejecutar **pip install WTForms** [?].

D.4. Compilación, instalación y ejecución del proyecto

Obtención del código

El código se encuentra disponible en el siguiente repositorio de GitHub: [Proyecto](#). Una vez dentro del repositorio, basta con descargar el proyecto desde el botón de descarga o clonación.

APÉNDICE D. DOCUMENTACIÓN TÉCNICA DE PROGRAMACIÓN

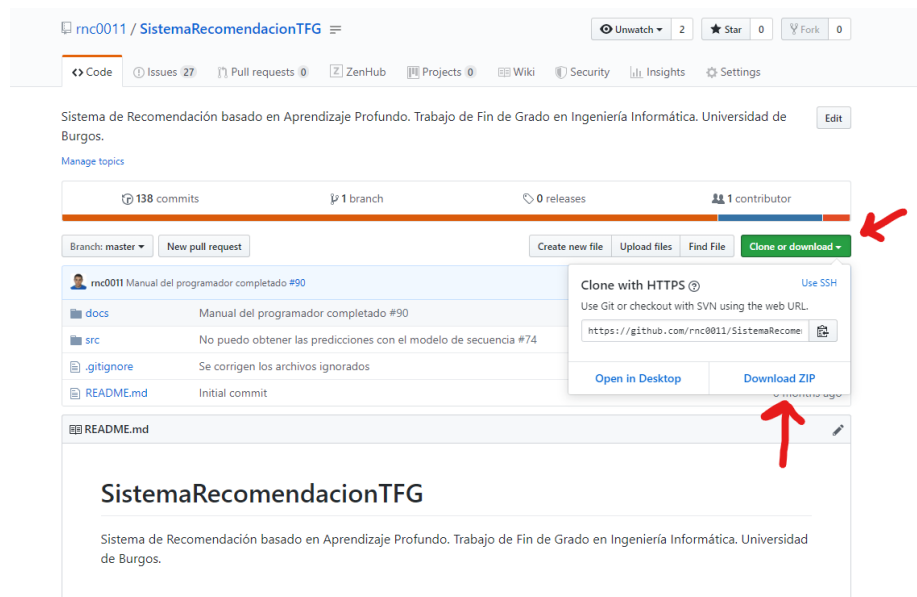


Figura D.1: Descargar proyecto

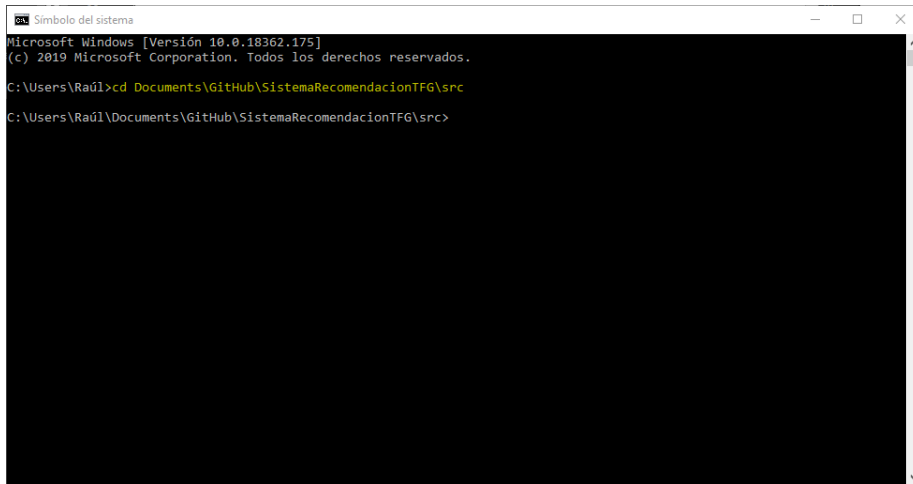
Instalación

Para poder compilar y ejecutar el proyecto es necesario tener instaladas todas las herramientas listadas anteriormente.

Compilación y ejecución

Una vez descargado el repositorio e instaladas todas las herramientas necesarias, se puede proceder a ejecutar la aplicación. Para ello, navegamos desde la consola hasta la carpeta *src* del proyecto:

D.4. COMPILACIÓN, INSTALACIÓN Y EJECUCIÓN DEL PROYECTO



```
Simbolo del sistema
Microsoft Windows [Versión 10.0.18362.175]
(c) 2019 Microsoft Corporation. Todos los derechos reservados.

C:\Users\Raúl>cd Documents\GitHub\SistemaRecomendacionTFG\src
C:\Users\Raúl\Documents\GitHub\SistemaRecomendacionTFG\src>
```

Figura D.2: Navegación hasta *src*

Para ejecutar el proyecto basta con ejecutar el archivo `__main__.py` que se encuentra dentro de *src*:

Figura D.3: Ejecución del proyecto

Metiendo la opción *2* ejecutamos la interfaz web obtenida gracias a *Flask*. Se tiene que abrir el navegador e ir a la dirección: **`http://localhost:5000/home`**.

APÉNDICE D. DOCUMENTACIÓN TÉCNICA DE PROGRAMACIÓN

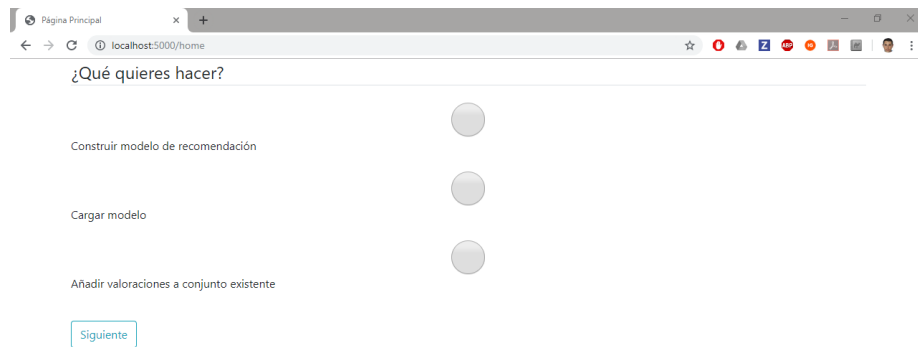


Figura D.4: Interfaz web de la aplicación

D.5. Pruebas del sistema

-

Apéndice E

Documentación de usuario

- E.1. Introducción
- E.2. Requisitos de usuarios
- E.3. Instalación
- E.4. Manual del usuario

Bibliografía

- [1] fast ai. *Practical Deep Learning for Coders, v3*.
- [2] Uriel Hernandez. Mvc (model, view, controller) explicado.
- [3] Maciej Kula. *Dataset construction*.
- [4] Maciej Kula. *Interactions*.
- [5] Maciej Kula. *Interactions - to_sequenc_e()*.
- [6] Maciej Kula. *Sequential models*.
- [7] Jure Leskovec, Anand Rajaraman, and Jeffrey D. Ullman. *Mining of Massive Datasets*, chapter 9, page 513. Universidad de Stanford, 2010.
- [8] Pandas. *DataFrame*.
- [9] Python Software Foundation. *pickle - Python object serialization*.