Deusto
Facultad de Ingeniería
Ingeniaritza Fakultatea

Grado en Ingeniería Informática

Informatikako Ingeniaritzako gradua

Proyecto fin de grado

Gradu amaierako proiektua

Estudio sobre la privacidad en el aprendizaje federado mediante el desarrollo de un sistema de recomendación online

Ibai Guilén Pacho

Director: Diego Casado Mansilla

Bilbao, mayo de 2021

Índice general

1. Objetivos y Alcance **1**
1.1. Obejetivos 1
1.1.1. Generales 1
1.1.2. Específicos 1
1.2. Alcance 2
1.2.1. Dentro del alcance 3
1.2.2. Fuera del alcance 3

2. Sistema de recomendación **5**
2.1. LightFM 5
2.1.1. Creación del modelo 5
2.1.2. Ajuste 7
2.1.3. Predicción 7
2.2. Evaluación 7
2.2.1. Métricas 7
2.2.2. Técnicas 7

Índice de figuras

Índice de tablas

Capítulo 1

Objetivos y Alcance

1.1. Obejetivos

Los objetivos de este proyecto pueden clasificarse en varios grupos. Por un lado están los objetivos generales, objetivos que a simple vista no tienen hitos específicos para en qué estado se encuentran y si se han cumplido o no.

Por otro lado se encuentran los objetivos específicos, estos, al contrario que los generales, son hitos específicos fácilmente identificables que permiten saber si la tarea se ha completado, y en caso contrario saber en qué estado se encuentra la tarea. Dentro de este grupo se encuentran los objetivos específicos de desarrollo, apartado donde se agrupan los objetivos que permiten saber en qué estado de desarrollo se encuentra el proyecto y que funcionalidades incorpora. Pero además, también se encuentran los objetivos de estudio, donde se incluyen los objetivos que tienen que ver con el estudio de los diferentes experimentos que se realizarán.

1.1.1. Generales

El objetivo general de este proyecto es el de desarrollar un sistema de recomendaciones online basado en federated learning. Este sistema se desarrollará sobre un sistema de recomendaciones ya existente basado en Machine Learning y deberá adaptarlo para su correcto funcionamiento con información descentralizada.

Para respetar la privacidad y derechos de los usuarios el proyecto deberá incluir la privacidad como patrón de diseño, cumpliendo tanto con las normativas vigentes en Europa y España,como con las posibles nuevas medidas que entren en vigor en un futuro.

Por último, el sistema deberá asegurar que la información se quede en el dispositivo y no se comparta ninguna información que no sea la del propio modelo desarrollado por el participante de la red.

1.1.2. Específicos

Los objetivos específicos son fácilmente reconocibles y concretos, lo que permite saber con exactitud cuando se han cumplido los objetivos y cuando no.

1.1.2.1. De desarrollo

Los objetivos de desarrollo tienen que ver con el correcto desarrollo del sistema de recomendación basado en federated learning.

Formación: El alumno deberá formarse en conocimientos sobre Inteligencia Artificial para poder comprender los conceptos. También deberá familiarizarse con el código del sistema de recomendación para realizar las modificaciones pertinentes.

Parametrización y configuración: Se tendrán que configurar las plataformas de desarrollo para poder ejecutar el sistema de recomendación centralizado con mayor rapidez y agilizar el desarrollo del proyecto. También se tendrán que configurar las Raspberries y la Jetson Nano, aprovisionarlas del software necesario para ejecutar en ellas el sistema de recomendación y ejecutarlo sin problemas.

Sistema de recomendación: Se deberá hallar la forma de combinar modelos de IA de cada Raspberry y se realizarán las modificaciones pertinentes en el sistema de recomendación para cumplir los siguientes puntos:

- Que sea capaz de obtener las recomendaciones para un único usuario en concreto.
- Que sea capaz de guardar y cargar los modelos entrenados de LightFM.
- Que el modelo de LightFM sea capaz de admitir aprendizaje incremental.

1.1.2.2. De estudio

Los objetivos de estudio tienen que ver con el análisis de los resultados del sistema de recomendación basado desarrollado sobre federated learning.

Análisis: El estudio del sistema deberá analizar la diferencia de rendimiento entre el sistema de recomendación con información centralizada y distribuida. Y analizar cómo afectan los distintos modelos y su combinación al rendimiento del sistema de recomendación.

Privacidad: Hay que valorar y analizar la privacidad y seguridad de los datos personales de los usuarios tanto en el modelo central, como en el distribuido. Se deberán evitar los problemas de seguridad y privacidad implícitos de la implementación de un sistema de recomendación basado en federated learning

1.2. Alcance

En el alcance se centra en concretar qué objetivos entran en el desarrollo del proyecto y cuáles no. Para ello se ha dividido este apartado en dos grupos, los objetivos y tareas que entran en el alcance del proyecto y los que no. La segregación en estos grupos nos permite concretar con más precisión los límites del proyecto, evitando que el proyecto se amplíe más allá de sus límites.

1.2.1. Dentro del alcance

Entran dentro del alcance los objetivos y tareas derivadas del desarrollo de un sistema de recomendación basado en federated learning, tanto las tareas de parametrización y configuración del hardware como el desarrollo de software. Esto incluye todos los objetivos específicos de desarrollo mencionados en el apartado anterior.

También se incluye el estudio y análisis de los resultados del sistema de recomendación, así como su precisión y sus inconvenientes, es decir, todo lo comentado en los objetivos específicos de estudio.

Para finalizar se incluirá un estudio de la estabilidad legal del sistema en el tiempo. Esto incluye la corroboración del cumplimiento de las actuales medidas de protección de datos y el estudio de las posibles futuras medidas, legislaciones y derechos que limiten el uso de información en Europa.

Todo esto quedará recogido en la memoria técnica del proyecto que será entregada como proyecto fin de grado.

1.2.2. Fuera del alcance

Fuera del alcance queda cualquier estudio de mercado sobre si la solución sería viable económicamente además de cualquier estudio de alternativas a federated learning.

No se incluirá ni se recogerá ninguna legislación fuera del marco jurídico Español y del Ordenamiento jurídico de la Unión Europea. Lo que deja fuera del alcance cualquier derecho, legislación o restricción de cualquier estado u organización ajeno a los comentados.

Queda fuera del alcance también la encriptación de las comunicaciones entre los diferentes dispositivos.

Capítulo 2

Sistema de recomendación

2.1. LightFM

LightFM es una implementación en python de varios algotitmos populares de recomendación.

2.1.1. Creación del modelo

El paso de creación del modelo es un paso complejo puesto que conlleva la correcta gestión de muchos datos e información. Asimismo, estos datos han de estar correctamente estructurados y construidos, puesto que LightFM requiere que la información este convertida a matrices dispersas y no tolera elementos vacíos en ellas. Para ello LightFM proveé de herramientas de creación de datasets que facilitan la creación de las matrices, de forma que sea más fácil incluirlas en el modelo.

Entre estas erramientas se encuentran:

- *build_user_features(...)* Permite crear la matriz CSR de usuarios y atributos de usuarios.
- *build_item_features(...)* Permite crear la matriz CSR de items y atributos de items.
- *build_interactions(...)* Permite crear las matriz COO de interacciones y las matriz COO de sus correspondientes pesos.

Hay que tener en cuenta que las matrices CSR (Compressed Sparse Row) hacen referencia a las matrices que admiten operaciones matriciales y acceso eficientemente; y que las matrices COO (Coordinate list) hacen referencia a las matrices que soportan modificaciones eficientemente, son generalmente utilizadas para construir matrices.

2.1.1.1. Atributos de los usuario

Para crear el modelo es necesario, entre otras cosas, disponer de las características de los usuarios. En este caso contamos con multitud de atributos sobre cada usuario: edad, género, nivel educativo, país, cultura de trabajo, perfil PST (Pinball, Shortcut, Thought-ful), barreras, intenciones y confianza.

Para convertir toda esa información del usuario a la matriz CSR que exige LightFM habrá que

llamar a *build_user_features(...)* con los IDs de usuario y y sus atributos, formando una lista que tenga listas de los IDs de los usuarios con sus listas de atributos, es decir:

$$\Big[\Big[userId_1 \quad \Big[feature_{11} \quad \cdots \quad feature_{1w}\Big]_1\Big] \quad \ldots \quad \Big[userId_v \quad \Big[feature_{v1} \quad \cdots \quad feature_{vw}\Big]_v\Big]\Big]$$

Teniendo en cuenta que:

$$\begin{aligned} v &\leftarrow \textit{Cantidad de IDs de usuario} \\ w &\leftarrow \textit{Contidad de atributos por usuario} \\ userId_v &\leftarrow \textit{ID del usuario } v \\ feature_{vw} &\leftarrow \textit{Atributo } w \textit{ del usuario } v \end{aligned}$$

Una vez creada la lista y pasado como parámetro al método, obtendremos la matriz CSR de atributos de usuario.

2.1.1.2. Atributos de los elementos

Además de los usuarios y de sus atributos, también se ha de disponer de los elementos a ordenar en el ranking, llamados items, y de sus características. En este caso estos items representan las estrategias de persuasión por las que se preguntó a los usuarios en el cuestionario. Sin embargo, al contrario que en el punto anterior, no contamos con tantos atributos sobre estos items, sino que cada estrategia de persuasión cuenta con dos atributos llamados dimensiones. Estos atributos no se encuentran presente en todas las estrategias, lo que da como resultado que haya algunas que tengan dos, una o ninguna dimension.

Para convertir toda esa información de los items a la matriz CSR que exige LightFM habrá que llamar a *build_item_features(...)* con los IDs de las estrategias y y sus atributos, formando una lista que tenga listas de los IDs de las estrategias con sus listas de atributos, es decir:

$$\Big[\Big[itemId_1 \quad \Big[feature_{11} \quad \cdots \quad feature_{1w}\Big]_1\Big] \quad \ldots \quad \Big[itemId_v \quad \Big[feature_{v1} \quad \cdots \quad feature_{vw}\Big]_v\Big]\Big]$$

Teniendo en cuenta que:

$$\begin{aligned} v &\leftarrow \textit{Cantidad de IDs de estrategias} \\ w &\leftarrow \textit{Contidad de dimensiones por estrategia} \\ itemId_v &\leftarrow \textit{ID de la estrategia } v \\ feature_{vw} &\leftarrow \textit{Atributo } w \textit{ de la estrategia } v \end{aligned}$$

Una vez creada la lista y pasado como parámetro al método, obtendremos la matriz CSR de atributos de usuario.

2.1.1.3. Interacciones

Para crear un modelo de LightFM se necesita
user_features, item_features
model = LightFM()

2.1.2. Ajuste

2.1.3. Predicción

2.2. Evaluación

La evaluación de los resultados obtenidos de las predicciones de los modelos es un apartado fundamental que permite analizar la eficacia de estas predicciones. Además, la utilización de métricas y técnicas permitirá poder comparar los resultados entre sí para poder demostrar si la predicción es correcta, aceptable o erronea.

2.2.1. Métricas

Antes de realizar el ranking hay que definir una métrica que permita evaluar la precisión de este, y por ende, la precisión del modelo. Para esta labor se ha utilizado la métrica NDPM, medida de rendimiento normalizada basada en la distancia. Esta métrica se utiliza para sistemas de recomendación basados en ranking y mide la eficacia del modelo para predecir el correcto orden de los n elementos del ranking.

NDPM	Medida de rendimiento normalizada basada en la distancia (Normalized Distance-based Performance Measure)

Cuanto menor sea el NDPM, mayor será la similitud entre el ranking predicho por el modelo y entre el declarado por el usuario. En el caso ideal donde la predicción hubiera acertado por completo el ranking del usuario, el NDPM sería 0. En caso contrario, es decir, que hubiera predicho un orden totalmente erroneo, el NDPM sería 1.

2.2.2. Técnicas

La lista de elementos a ser ordenados por los modelos LightFM son los siguientes:

$$strats = \begin{bmatrix} 'v2' & 'v5' & 'v6' & 'v7' & 'v10' & 'v11' & 'v15' & 'v17' & 'v19' & 'v20' \end{bmatrix}$$

Lista de estrategias a clasificar del 1 a n

En esta lista solo aparecen los elementos más relevantes de las encuestas, ya que (nidea)

i Número de modelos LightFM

n Número de elementos a incluir en el ranking

m Número de epochs sobre cada dataset

a_{mn} Índice del elemento n de la lista de strats en el ranking calculado en la epoch m

$$matriz_{i[m,n]} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix}$$

Matriz numpy de los resultados de la predicción del modelo i

$$prediction_{m[1,n]} = \begin{bmatrix} a_1 & a_2 & \cdots & a_n \end{bmatrix}$$

Lista de los índices de los n strat para el ranking del epoch m

$$Predictions_{i[m,n]} = \begin{bmatrix} \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \end{bmatrix} \\ \begin{bmatrix} a_{21} & a_{22} & \cdots & a_{2n} \end{bmatrix} \\ \vdots \\ \begin{bmatrix} a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \end{bmatrix}$$

Lista de predicciones del modelo i por cada epoch m

$$ndpms_{i[1,m]} = \begin{bmatrix} ndpm_1 & ndpm_2 & \cdots & ndpm_m \end{bmatrix}$$

Lista de NDPMs calculados sobre las predicciones de cada epoch m del modelo i LightFM

Si se desagrupan las predicciones realizadas por cada epoch m y se agrupan en función de cada strat n tendremos la lista de valores que el modelo i da a cada uno de los elementos n de la lista de los strat. Esto quiere decir que si se agrupan en función del elemento que se ordena en el ranking en vez de en por predicción se obtendrá la lista de todas las predicciones de posición para cada uno de los elementos a ordenar.

$$Predictions_{i[m,n]} = \begin{bmatrix} \begin{bmatrix} a_{11} \\ a_{21} \\ \vdots \\ a_{m1} \end{bmatrix} & \begin{bmatrix} a_{12} \\ a_{22} \\ \vdots \\ a_{m2} \end{bmatrix} & \cdots & \begin{bmatrix} a_{1n} \\ a_{2n} \\ \vdots \\ a_{mn} \end{bmatrix} \end{bmatrix}$$

Una vez teniendo la lista de los valores para cada elemento se pueden aplicar métodos estadísticos como la media($\bar{x}$), mediana($\tilde{x}$) o moda ($\hat{x}$) para crear una única predicción consensuada para el modelo i al que pertenecen estas predicciones. Esta predicción consensuada permitirá combinar los rankings generados durante cada epoch para poder obtener la máxima precisión posible y la

mínima desviación típica.

$\bar{x}$Media

$\tilde{x}$Mediana

$\hat{x}$Moda

Consenso por media de las m predicciones para el modelo i:

$$\overline{consensous_{i[1,m]}} = \left[\overline{a_{11}} \quad \overline{a_{21}} \quad \cdots \quad \overline{a_{m1}}\right]$$

Consenso por mediana de las m predicciones para el modelo i:

$$\widetilde{consensous}_{i[1,m]} = \left[\widetilde{a_{11}} \quad \widetilde{a_{21}} \quad \cdots \quad \widetilde{a_{m1}}\right]$$

Consenso por moda de las m predicciones para el modelo i:

$$\widehat{consensous}_{i[1,m]} = \left[\widehat{a_{11}} \quad \widehat{a_{21}} \quad \cdots \quad \widehat{a_{m1}}\right]$$

Algorithm 1 Entrenamiento del modelo

procedure EXPERIMENTO
    // Lista de estrategias a clasificar del 1 a n:
    $strats \leftarrow \begin{bmatrix} 'v2' & 'v5' & 'v6' & 'v7' & 'v10' & 'v11' & 'v15' & 'v17' & 'v19' & 'v20' \end{bmatrix}$

    $i \leftarrow$ *Cantidad de modelos a crear.*
    $m \leftarrow$ *Cantidad de epochs sobre cada modelo i.*

    // Lista de listas de NDPMs, de cada predicción de la epoch m, de cada modelo i:
    // [[$ndpm_1$*,* $ndpm_2$*, ...,* $ndpm_m$*]*$_1$*, ..., [*$ndpm_1$*,* $ndpm_2$*, ...,* $ndpm_m$*]*$_i$*]*
    $ndpms \leftarrow [[float_m]_i]$

    // Lista de NDPMs, de cada predicción consensuada, de cada modelo i:
    // [$ndpm_1$*,* $ndpm_2$*, ...,* $ndpm_i$*]*
    $commonNdpms \leftarrow [float_i]$

    // Lista de los índices de los n strat para el ranking del epoch m
    // $\begin{bmatrix} float_n \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & \cdots & a_n \end{bmatrix}$
    // Lista de los índices de los n strat, de cada epoch m
    // $\begin{bmatrix} \begin{bmatrix} float_n \end{bmatrix}_m \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} a_1 & a_2 & \dots & a_n \end{bmatrix}_1 & \dots & \begin{bmatrix} a_1 & a_2 & \dots & a_n \end{bmatrix}_m \end{bmatrix}$
    // Lista de los índices de los n strat, de cada epoch m, de cada modelo i

$$// \left[\left[\begin{bmatrix} float_n \end{bmatrix}_m\right]_i\right] = \left[\begin{bmatrix}\begin{bmatrix} a_1 & a_2 & \cdots & a_n \end{bmatrix}_1 \\ \vdots \\ \begin{bmatrix} a_1 & a_2 & \cdots & a_n \end{bmatrix}_m\end{bmatrix}_1 \quad \cdots \quad \begin{bmatrix}\begin{bmatrix} a_1 & a_2 & \cdots & a_n \end{bmatrix}_1 \\ \vdots \\ \begin{bmatrix} a_1 & a_2 & \cdots & a_n \end{bmatrix}_m\end{bmatrix}_i\right]$$

    $predictions \leftarrow \left[\left[\begin{bmatrix} float_n \end{bmatrix}_m\right]_i\right]$
loop:
    for *int*$(j) \leftarrow 0$ **to** *range*(i) **do**
        $userFeatures, itemFeatures \leftarrow features()$ *//Obtiene todas las características de los usuarios y de los items*

        $trainInteractions \leftarrow$ *Interacciones para entrenar el modelo.*
        $testInteractions \leftarrow$ *Interacciones para probar el modelo.*

        $trainWeights \leftarrow$ *Peso de las interacciones de entrenamiento.*

        $trainUerid \leftarrow$ *IDs de los usuarios que se utilizan para entrenar.*
        $testUserid \leftarrow$ *IDs de los usuarios que se utilizan para probar.*
loop:
        for *int*$(k) \leftarrow 0$ **to** *range*(m) **do**
            $modelo \leftarrow LightFM()$ *//Se crea el modelo LightFM*

            //Ajustar modelo a la información extraida:
            $modelo.fit(trainInteractions, userFeatures, trainWeights)$

            //Predicción del modelo para las interacciones de prueba:
            $predictRank \leftarrow modelo.predict(testInteractions, userFeatures)$

            $ndpms[i].append(ndpm(ranks_test))$ *//Añadir el valor NDPM del ranking creado por el modelo i, en la epoch m, a la lista de NDPMs del modelo i*
            close;
        $i \leftarrow i - 1.$
        goto *loop.*
        close;

```
50: top:
51: if i > stringlen then return false
52: j ← patlen
53: loop:
54: if string(i) = path(j) then
55:     j ← j − 1.
56:     i ← i − 1.
57:     goto loop.
58:     close;
59: i ← i + máx(delta1(string(i)), delta2(j)).
60: goto top.
```

Algorithm 2 Algoritmo de consenso

```
1: procedure CONSENSUAR(matriz)
2:     a
3:     stringlen ← length of string
4:     i ← patlen
5: top:
6:     if i > stringlen then return false
7:     j ← patlen
8: loop:
9:     if string(i) = path(j) then
10:        j ← j − 1.
11:        i ← i − 1.
12:        goto loop.
13:        close;
14:    i ← i + máx(delta1(string(i)), delta2(j)).
15:    goto top.
```

Bibliografía