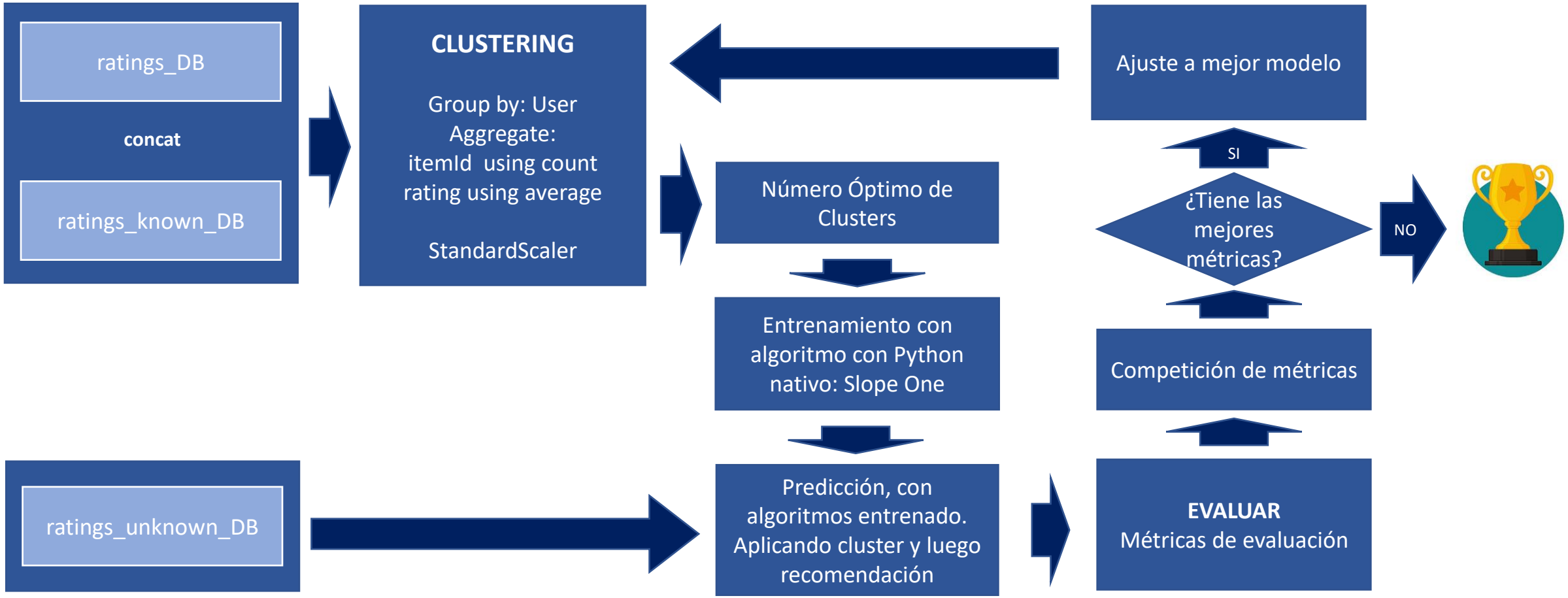
Model-based Collaborative Filtering

(User Clustering + Slope One)

DISEÑO DEL SISTEMA DE RECOMENDACIÓN

Arquitectura del Modelo

Diseño del Sistema de Recomendación



IMPLEMENTACIÓN DE ALGORITMO SLOPE ONE

Lenguaje PYTHON en JUPYTER NOTEBOOK

Algoritmo ONE SLOPE

Método empleado para el Filtrado Colaborativo fácil de implementar de manera eficiente y con una precisión similar a algoritmos con mayor complejidad, de difícil implementación y costosos al entrenar.

- Entrada: Un DATAFRAME con 3 columnas (USERID, ITEMID y RATING)
- Proceso:
  - La entrada se transforma en una matriz de ITEMS y USUARIOS con los RATINGS como valores.
  - Se itera los ítems de forma secuencial y anidada para hallar la diferencia promedio entre cada ITEM; para esto se busca a los usuarios que hayan valorado ambos ITEMS.
  - Se genera una matriz simétrica de las diferencias promedio (DESVIACIONES), donde el lado inferior de la diagonal representa la diferencia promedio negativa debido al cambio en el orden de los ITEMS comparados.
- Salida: Dos DATAFRAMES.
  - El primero es una matriz de ITEMS (columnas) y USUARIOS (filas) con los RATINGS como valores.
  - El segundo es una matriz de ITEMS (columnas y filas) con las DESVIACIONES como valores.

```
def one_slope(dataframe):
    rating_matrix = dataframe.pivot_table(index='userId’,
                                          columns='itemId’,
                                          values='rating')
    def get_dev_val(dataframe, i, j):
        dataframe = dataframe[
            (dataframe[i].notnull())
            & (dataframe[j].notnull())
        ].copy()
        users = len(dataframe)
        if users > 0:
              dev_val = (rating_matrix[i]-rating_matrix[j]).sum() / users
        else:
              dev_val = 0
        return dev_val

    deviations = {}
    for i in rating_matrix.columns:
        for j in rating_matrix.columns:
            if i == j:
                continue
            deviation = get_dev_val(rating_matrix, i, j)
            deviations[i, j] = deviation
            deviations[j, i] = (-1) * deviation

    desviation_matrix = get_matrix(deviations)
    return rating_matrix, desviation_matrix
```

Aplicación del Filtrado Colaborativo

Cuando se identifican las agrupaciones de usuarios según la caracterización de sus variables, se itera cada una de las agrupaciones para filtrar a los usuarios de esa agrupación y obtener sus matrices de calificaciones y desviaciones; Este proceso se realiza para cada agrupación. Durante el proceso se almacenan los resultados obtenidos y la estructura de datos empleada para esta actividad es un diccionario.

```
matrix_per_cluster = {}
for cluster in df_user_clustered['cluster'].unique():
    matrix = {}
    users_clustered = df_user_clustered[
                df_user_clustered['cluster'] == cluster
    ]['userId'].tolist()
    df_ratings_clustered = df_ratings_all[
                df_ratings_all['userId'].isin(users_clustered)
    ]
    matrixes = [one_slope(df_ratings_clustered)]
    matrix['rating_matrix’] = matrixes[0]
    matrix['desviation_matrix’] = matrixes[1]
    matrix_per_cluster[cluster] = matrix
```

DICCIONARIO DE AGRUPACIONES Y MATRICES

```
{
    ("uc_0", "user cluster", int) : {
        ("rating_matrix", "ratings", str) :
                ("rm_uc_0", "ratings matrix of items and users", DataFrame),
        ("desviation_matrix", "desviations", str) :
                ("dm_uc_0", "desviations matrix of items", DataFrame)
    },
    ("uc_1", "user cluster", int) : {
        ("rating_matrix", "ratings", str) :
                ("rm_uc_1", "ratings matrix of items and users", DataFrame),
        ("desviation_matrix", "desviations", str) :
                ("dm_uc_1", "desviations matrix of items", DataFrame)
    },
    ...
    ("uc_n", "user cluster", int) : {
        ("rating_matrix", "ratings", str) :
                ("rm_uc_n", "ratings matrix of items and users", DataFrame),
        ("desviation_matrix", "desviations", str) :
                ("dm_uc_n", "desviations matrix of items", DataFrame)
    }
}
```

Predecir Calificaciones de las Películas

Se recomendará ITEMS (películas) a 100 usuarios, para esto se itera cada uno de los usuarios. Durante este proceso:

- Por cada usuario, se utiliza el algoritmo ONE SLOPE para calcular los posibles RATINGS que el usuario evaluado le asignará a ITEMS que no ha calificado (se asume que al no calificarlo, no lo ha visto).
- Para este calculo se utiliza la matriz de DESVIACIONES y los elementos ya calificados. Por cada elemento evaluado, se le asigna un valor en base a los ya calificados con la desviación correspondiente. Luego se promedian estos valores para asignar una única posible calificación a la película.
- Las películas que el usuario ya ha calificado no se recomiendan. El rating asignado para dichos elementos es NULO.
- El resultado es una matriz de RATINGS CALCULADOS.

```
user_to_predict = df_ratings_known['userId'].unique().tolist()
predictions = {}

for u in user_to_predict:

    user_cluster = df_user_clustered[df_user_clustered['userId'] == u]['cluster'].values[0]

    user_matrix = matrix_per_cluster[user_cluster]

    user_rating_matrix = user_matrix['rating_matrix']

    user_desviation_matrix = user_matrix['desviation_matrix']

    user_row = user_rating_matrix.loc[u, :]

    user_ranked_items = user_row[user_row.notnull()].index

    for i in user_desviation_matrix.columns:

        if i not in user_ranked_items:

            predictions[u, i] = (np.sum(user_rating_matrix.loc[u, user_ranked_items] +
user_desviation_matrix.loc[i, user_ranked_items])) / len(user_ranked_items)

predictions_matrix = get_matrix(predictions)
```

	2	3	5	6	11	16	17	19	21	25	...	8961	30707	40815	44191	45722	46578	48780	58559	68954	74458
2378	1.918205	1.155556	1.475439	2.406978	2.226059	2.483056	2.649987	1.29926	2.100389	2.313801	...	2.5229	2.789054	2.310497	2.561765	2.402098	2.401357	2.620192	2.952936	2.473558	2.664773
7238	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2.85091	NaN	NaN	...	4.106459	4.244212	3.637281	3.981794	3.533441	4.093955	4.227227	4.418837	4.125573	3.97253
10218	3.263352	2.981511	3.140519	3.978437	3.77126	4.098286	4.252811	NaN	3.634626	3.806267	...	NaN	4.050245	NaN	3.887468	NaN	3.976181	4.124794	4.255948	4.022812	3.946956
11418	2.638025	NaN	2.264049	3.068709	3.042308	3.279478	3.427632	2.170601	2.858212	2.902021	...	3.299451	3.565054	2.638319	3.058284	2.469792	3.243697	3.461364	3.564621	3.13499	3.208482
15238	NaN	2.089617	2.272088	3.125048	2.907296	3.213722	3.401451	NaN	2.811446	2.937821	...	3.15377	3.258437	2.797441	3.061509	2.68518	3.120482	3.265163	3.428675	3.243926	3.078242
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
243538	3.202597	2.794086	2.967	3.639656	3.595341	3.613809	4.017387	2.478757	3.649233	3.713673	...	3.966034	3.676698	3.577981	3.745827	3.316135	3.679894	3.983704	4.057039	3.498677	3.747501
247588	3.022682	2.744751	2.789417	3.57845	3.47584	3.559704	3.828527	2.338973	NaN	3.666455	...	3.925507	3.780654	3.545199	3.800565	3.315285	3.766484	4.049717	4.196021	3.685985	3.773872
249158	3.08057	2.748848	2.754401	3.665788	3.545579	3.766277	3.78344	2.38568	3.408097	3.724258	...	3.891363	3.812901	3.573125	3.873681	3.411298	3.787321	4.045549	4.169128	3.698272	3.829108
259448	3.200281	3.025593	2.934469	3.732673	3.632756	3.742685	3.939885	2.54997	NaN	3.75178	...	4.04298	3.923655	3.658706	3.969849	3.521478	3.879067	4.174426	4.298909	3.843953	3.955718
269158	2.865338	2.578945	2.496918	3.379593	3.291347	3.542963	3.442889	2.196802	3.153971	3.334714	...	3.65508	3.576502	3.307413	3.591625	3.192426	3.501805	3.847571	3.933071	3.450131	3.561263

100 rows × 250 columns

Recomendación de Películas

Se escoge las mejores predicciones en base al RATING predicho de los ITEMS (películas) sin calificar para los 100 usuarios; para esto:

- Se iteran los usuarios de la matriz de RATINGS predichos.
- Se ordenan los ITEMS (películas) de mayor a menor.
- Se escogen las 10 películas con mayor puntaje.
- Se almacenan en un diccionario, donde cada llave es un usuario y cada valor es una lista de 10 tuplas (ITEM, RATING) con mayor puntaje de calificación.

```
best_predictions = {}

for idx, row in predictions_matrix.iterrows():
    top_10 = row.sort_values(ascending=False).head(10)
    best_predictions[idx] = list(zip(top_10.index, top_10.values))

best_predictions
```

DICCIONARIO DE PELICULAS RECOMENDADAS Y SUS POSIBLES RATINGS

```
{
    (“user_0", "user", str) : [
        ((“item_0", “item", str), (“rating_0", “rating", float)),
        ((“item_1", “item", str), (“rating_1", “rating", float)),
        ((“item_2", “item", str), (“rating_2", “rating", float)),
        ...
        ((“item_9", “item", str), (“rating_9", “rating", float))
    ],
    (“user_1", "user", str) : [
        ((“item_0", “item", str), (“rating_0", “rating", float)),
        ((“item_1", “item", str), (“rating_1", “rating", float)),
        ((“item_2", “item", str), (“rating_2", “rating", float)),
        ...
        ((“item_9", “item", str), (“rating_9", “rating", float))
    ],     ...
    (“user_n", "user", str) : [
        ((“item_0", “item", str), (“rating_0", “rating", float)),
        ((“item_1", “item", str), (“rating_1", “rating", float)),
        ((“item_2", “item", str), (“rating_2", “rating", float)),
        ...
        ((“item_9", “item", str), (“rating_9", “rating", float))
    ]
}
```

Evaluación con Métricas Cualitativas

Se emplearán métricas cualitativas para medir la eficacia del algoritmo empleado. Estás métricas son:

- DCG: El valor de relevancia graduada se reduce logarítmicamente proporcional a la posición del resultado.

$$\mathrm{DCG_p} = \sum_{i=1}^{p} \frac{rel_i}{\log_2(i+1)}$$

- IDCG: El valor de relevancia graduada se reduce logarítmicamente proporcional a la posición relativa del valor asignado por usuario.

$$\mathrm{IDCG_p} = \sum_{i=1}^{|REL_p|} \frac{rel_i}{\log_2(i+1)}$$

- NDCG: Normalización de la métrica en base a la división del DCG y el IDCG.

$$\mathrm{nDCG_p} = \frac{DCG_p}{IDCG_p}$$

```python
def discounted_cumulative_gain_v2(recommended_list, ground_truth_list):
    dcg_value = 0
    asserts = recommended_list[
                recommended_list['itemId'].isin(ground_truth_list['itemId’])
    ]['itemId'].tolist()
    for assert_item in asserts:
        gt_item_position = ground_truth_list[ground_truth_list['itemId'] == assert_item].index[0]
        r_item_position = recommended_list[recommended_list['itemId'] == assert_item].index[0]
        gt_item_rating = ground_truth_list.loc[gt_item_position, 'rating']
        dcg_value += gt_item_rating / np.log2(r_item_position + 2)
    return dcg_value
```

```python
def ideal_discounted_cumulative_gain_v2(recommended_list, ground_truth_list):
    idcg_value = 0
    asserts = recommended_list[
                recommended_list['itemId'].isin(ground_truth_list['itemId’])
    ]['itemId'].tolist()
    for assert_item in asserts:
        gt_item_position = ground_truth_list[ground_truth_list['itemId'] == assert_item].index[0]
        r_item_position = recommended_list[recommended_list['itemId'] == assert_item].index[0]
        gt_item_rating = ground_truth_list.loc[gt_item_position, 'rating']
        idcg_value += gt_item_rating / np.log2(gt_item_position + 2)
    return idcg_value
```

Evaluación del Modelo

Se almacenan todas las métricas en un DATAFRAME, donde los índices son los USERID y las columnas son: ASSERTS (número de aciertos en las recomendaciones), DCG, IDCG y NDCG. Con este DATAFRAME podremos analizar el modelo de recomendación, a través de sus aciertos y el NDCG promedio del modelo.

```
df_qualitive_measures = pd.DataFrame(
                            data=qualitive_measures.values(),
                            columns=['asserts', 'dcg', 'idcg', 'ndcg'],
                            index=qualitive_measures.keys()
                        )
df_qualitive_measures
```

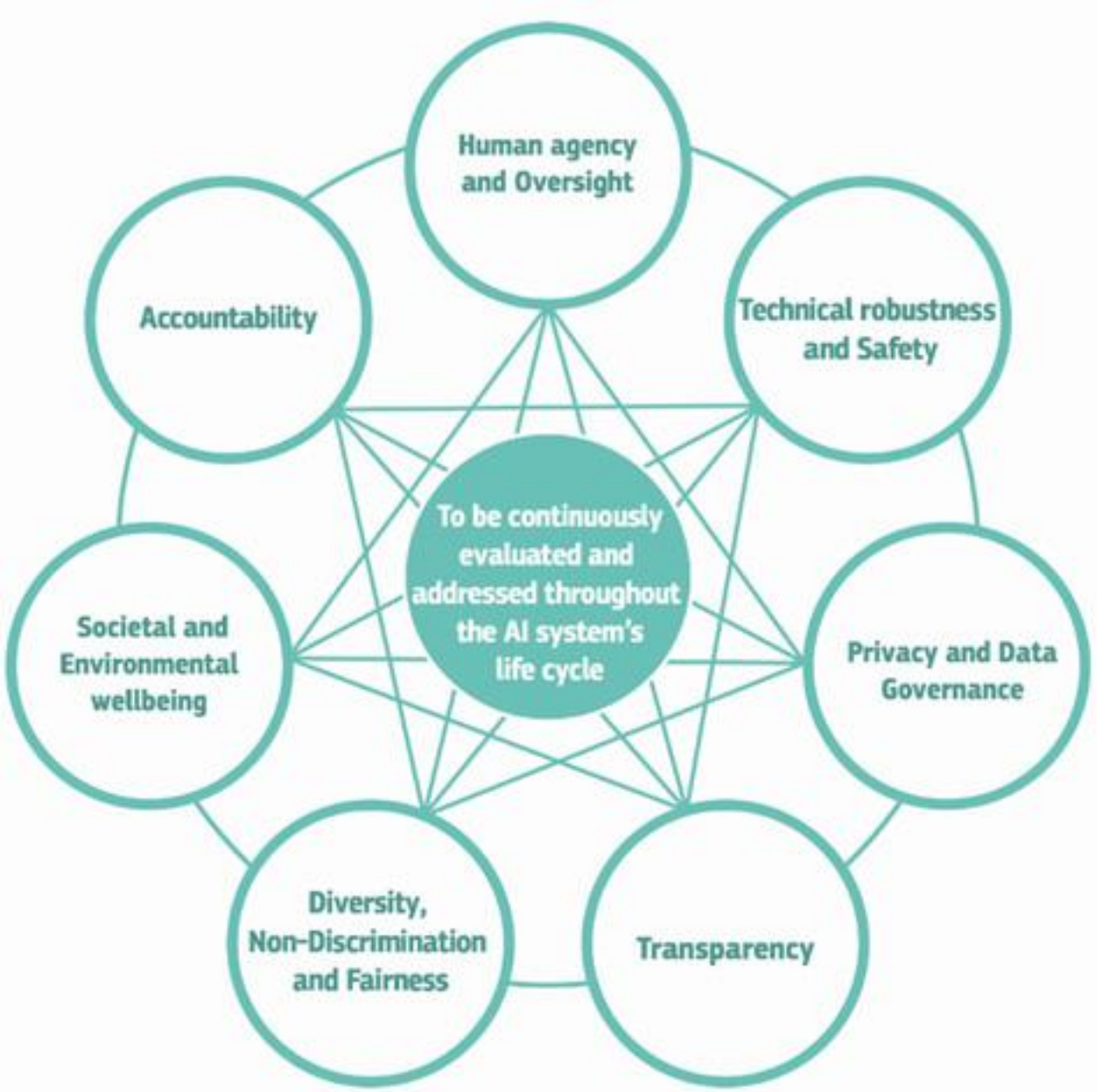
	asserts	dcg	idcg	ndcg
125328	0	0.000000	0.000000	0.000000
79268	1	4.500000	1.938045	2.321928
152978	0	0.000000	0.000000	0.000000
171648	0	0.000000	0.000000	0.000000
227648	1	2.250000	2.839184	0.792481
...	...	...	...	...
166408	0	0.000000	0.000000	0.000000
125668	1	1.934264	5.000000	0.386853
21608	1	1.750000	2.208254	0.792481
7238	0	0.000000	0.000000	0.000000
201128	2	6.781036	5.654649	1.199197

100 rows × 4 columns

Hallazgos del modelo:

- Se agrupan los usuarios en 10 CLUSTERS
- Usuarios con una recomendación mínima de una película acertada 54 de 100
- Porcentaje de usuarios con una recomendación mínima de una película acertada 54.0%
- Películas recomendadas asertivamente 86 de 1000
- Porcentaje de películas recomendadas asertivamente 8.6%
- Promedio de Normalized Discount Cumulative Gain (nDCG) 0.5483345042593719

CONSIDERACIONES ÉTICAS Y REGULATORIAS



Human agency and Oversight

- Para nuestro modelo no utilizamos ninguna información personal sobre los usuarios y las recomendaciones se basan en una calificación colaborativa por grupos de usuarios los cuales son agrupados en base a sus gustos, por lo que no perjudicamos los derechos fundamentales.
- El modelo no afecta la autonomia individual, pero si brinda un apoyo en la selección de items; no usamos procesos subconcientes para fomentar que se ecoja un item particular sino se utiliza una concienca colectiva para recomendar ciertos items.
- Se establecerá la capacidad de supervisar la actividad general del sistema y sus diferentes impactos, tales como económico, social, legal y ético.
- Se establecerá un mecanismo de retroalimentación para que los usuarios puedan informar sobre cualquier problema o duda relacionado con la privacidad y el uso de sus datos.

Technical robustness and Safety

- Para nuestro modelo si se esta considerando la posibilidad de tener los siguientes puntos de mejora para tener robustes y seguridad:
  - Envenenamiento de Datos
  - Evitar fuga de Modelo trabajando en Cloud
  - Una arquitectura de sistema escalable
  - Recomendaciones precisas y relevantes(se tendría que trabajar con mas tipo de datos para lograrlo)
  - Otro tipo de ataques (hacking y el malware)
  - Políticas de privacidad claras para los usuarios
- Para mitigar estos problemas a parte de implementar metodos de seguridad y monitoreo humano utilizaremos las siguientes metricas para verificar que el modelo no haya sido envenenado e incrementar la confiabilidad:
  - MAE
  - RMSE
- El modelo podra ser repoducible para su investigación

Privacy and Data Governance

- Nuestro modelo no maneja información personal de los usuarios, esto se explicará en las políticas de privacidad.
- Nuestro modelo realiza un perfilamiento de los usuarios de manera colaborativa basada en clusters, esto significa que basamos las recomendaciones en base a ciertos grupos de usuario con preferencias similares.
- Nuestro modelo se clusteriza de manera constante por lo cual las preferencias de un usuario pueden cambiar y se puede mover de grupo.
- Nuesto modelo debera restringir el acceso a los datos para evitar cualquier tipo de envenenamiento o extracción de datos.
- El sistema de recomendación debe permitir a los usuarios controlar su información personal y elegir qué datos compartir.

Nota : Pero de igual manera se requiere el monitoreo humano puesto que el modelo puede clusterizar usuarios en base contenidos de las películas que puedan influir en orientación sexual, edad, género, opiniones religiosas o políticas, y se necesita velar por que estos grupos nos se usen para discriminarlos de manera ilegal o injusta.

Transparency

- Se documentara el pre-procesamiento de datos y los algoritmos usados con el fin de que se pueda realizar una trazabilidad.
- Nuestro modelo no se presentara como humano ante los usuarios
- El sistema tendrá una metodología clara y comprensible para generar recomendaciones, que pueda ser explicada a los usuarios si lo solicitan.
- El sistema de recomendación debe tener políticas claras y transparentes sobre cómo se manejan y protegen los datos de los usuarios.
- El sistema de recomendación no cuenta con una retroalimentación en tiempo real que permita a los usuarios ajustar sus preferencias y criterios de recomendación en cualquier momento, pero se implementara según vaya creciendo el modelo.

Diversity Non-Discrimination and Fairness

- La información y el modelo tanto de los rating de los usuarios como de las peliculas pasaran por un proceso de identificación de sesgos historicos, modelos imcompleto, y mala gobernanza. Esto con el fin de evitar:
  - Explotación intencional de sesgos
  - Marginación de usuarios / items

- Para este modelo no aplicamos el diseño universal o si lo aplicaremos realizaremos una segmentación de los usuarios y items en base a edad y la clasificación de los items. Esto con el objetivo de evitar que el sistema de recomendación recomiende peliculas no aptas para todo publico a menores de edad.

- Para este modelo se solicitara retroalimentación por el lado de usuarios y de los proveedores de items con tal de asegurar un trato justo y equitativo.

Societal and Environmental wellbeing

- Para manejar el modelo de manera amable con la sociedad y el medio ambiente tomaremos las siguientes precauciones:
  - Rastreo de huella de carbono con el objetivo de reducir la contamincación al entrenar y operativilizar el modelo
  - Evaluación y monitorización de impacto social en usuarios con el objetivo de asegurarnos que el modelo no deteriore o afecte las habilidades sociales de los usuarios, velando por el vienestar fisico y mental.
  - Evaluación y monitorización de impacto en la sociedad, evaluando que no interfiramos con procesos democraticos, manifestaciones etc.

Accountability

- El modelo expondra su algoritmo utilizado con el fin de ser auditable, por motivos de propiedad intelectual y protección de usuarios no se podra compartir la data utilizada.
- El modelo Identificará, evaluará, informará y minimizará los posibles impactos negativos del modelo de recomendación de manera constante.
- Si el modelo genera un conflicto, se habilitara las negociaciones y las compensaciones respectivas.
- Si se produce un impacto adverso injusto se habilitaran medidas rápidas de reparación