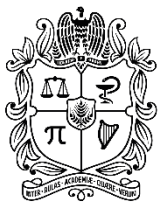


UNIVERSIDAD
NACIONAL
DE COLOMBIA

Modelo de Sistema de Recomendación para visitas guiadas, basado en computación ubicua y sensible al contexto

Juan Carlos Gil Vera

Universidad Nacional de Colombia

Facultad de Minas, Departamento de Ciencias de la Computación y de la Decisión

Medellín, Colombia

2023

Modelo de Sistema de Recomendación en computación ubicua y sensible al contexto

Juan Carlos Gil Vera

Tesis de investigación presentada como requisito parcial para optar al título de:
Magister en Ingeniería de Sistema

Director (a):
Ph.D. Demetrio Arturo Ovalle Carranza

Línea de Investigación:
Inteligencia Artificial

Universidad Nacional de Colombia
Facultad de Minas, Departamento de Ciencias de la Computación y de la Decisión
Medellín, Colombia
2023

A mi madre quien siempre me ha enseñado el valor del trabajo honesto y responsable.

Declaración de obra original

Yo declaro lo siguiente:

He leído el Acuerdo 035 de 2003 del Consejo Académico de la Universidad Nacional. «Reglamento sobre propiedad intelectual» y la Normatividad Nacional relacionada al respeto de los derechos de autor. Esta disertación representa mi trabajo original, excepto donde he reconocido las ideas, las palabras, o materiales de otros autores.

Cuando se han presentado ideas o palabras de otros autores en esta disertación, he realizado su respectivo reconocimiento aplicando correctamente los esquemas de citas y referencias bibliográficas en el estilo requerido.

He obtenido el permiso del autor o editor para incluir cualquier material con derechos de autor (por ejemplo, tablas, figuras, instrumentos de encuesta o grandes porciones de texto).

Por último, he sometido esta disertación a la herramienta de integridad académica, definida por la universidad.

JUAN CARLOS GIL VERA

Nombre

04/11/2023

Fecha

Agradecimientos

Agradezco a mi director de tesis Demetrio Ovalle, por darme la oportunidad de ser mi tutor y de orientarme para trabajar en esta maestría a pesar de mis problemas personales y quebrantos de salud, profesor que siempre he admirado desde el pregrado y que en los momentos de dificultad siempre me demostró su paciencia y comprensión.

Resumen

Modelo de Sistema de Recomendación para visitas guiadas, basado en computación ubicua y sensible al contexto

El modelo propuesto pensado para funcionar en un contexto de visitas guiadas, se basa en el desarrollo de la ontología tourist en *python* usando la librería *Owlready*, y describe las entidades del modelo para visitas guiadas. La ontología permite aplicar los conceptos de ubicuidad y permite representar la sensibilidad al contexto en tres formas, con el contexto geográfico, temporal y ambiental. Para la visita guiada se considera el perfil del usuario, sus preferencias, el estado emocional y las evaluaciones de los lugares visitados, así mismo, el perfil, el itinerario y las características del sitio, las preferencias de transporte del usuario y las características de transporte del sitio. Se utilizó un lenguaje de ontologías que modela los conceptos y características del sistema de visitas guiadas que permite realizar inferencias con reglas usando el lenguaje *SWRL* con el razonador Pellet. Para el modelo de recomendación, se han desarrollado modelos de filtrado colaborativo, centrados en el usuario usando la media y la media ponderada de los puntajes de los sitios, y la información demográfica del usuario. Se han elaborado dos modelos de recomendación de filtrado colaborativo basado en clustering y usando filtrado con descomposición de valores singulares. Y un modelo de recomendación híbrido con una técnica de validación cruzada quíntuple. Todos los modelos fueron evaluados usando la métrica *RMSE* y para evaluar las predicciones se han usado las métricas de precisión, recall y F1 score. Finalmente, como aporte adicional a la tesis, se utilizó la técnica de análisis de sentimientos de Machine Learning para determinar el nivel de percepción del sitio de interés y así validar la utilidad del modelo para visitas guiadas.

Palabras clave: modelo de recomendación, análisis de sentimientos, visita guiada, sensibilidad al contexto, ubicuidad.

Abstract

Recommendation System Model for guided tours, based on ubiquitous and context-sensitive computing

The proposed model, designed to work in a guided tour context, is based on the development of the tourist ontology in python using the Owlready library, and describes the entities of the guided tour model. The ontology allows the application of the concepts of ubiquity and allows the representation of context sensitivity in three ways, with the geographic, temporal and environmental context. For the guided tour, the user profile, preferences, emotional state and evaluations of the places visited are considered, as well as the profile, itinerary and characteristics of the site, the user's transportation preferences and the transportation characteristics of the site. An ontology language was used that models the concepts and characteristics of the guided tour system that allows inferences to be made with rules using the SWRL language with the Pellet reasoner. For the recommendation model, collaborative filtering models have been developed, centered on the user using the mean and weighted mean of the scores of the sites, and the demographic information of the user. Two collaborative filtering recommendation models based on clustering and using filtering with singular value decomposition have been developed, as well as a hybrid recommendation model with a quintuple cross-validation technique. All models were evaluated using the RMSE metric and the precision, recall and F1 score metrics were used to evaluate the predictions. Finally, as an additional contribution to the thesis, the Machine Learning sentiment analysis technique was used to determine the level of perception of the site of interest and thus validate the usefulness of the model for guided tours.

Keywords: recommendation model, sentiment analysis, guided tour, context sensitivity, ubiquity.

Contenido

Pág.

Resumen ..... **IX**

Lista de figuras ..... **XVI**

Lista de tablas ..... **XVIII**

Lista de abreviaturas ..... **XIX**

1. Introducción **21**
1.1 Contextualización 21
1.2 Justificación 22
1.3 Hipótesis 23
1.4 Problema de Investigación 23
1.5 Objetivo general 24
1.6 Objetivos específicos 25
1.7 Alcance 25
1.8 Metodología 26
1.9 Contribuciones 30
1.10 Organización del documento 31

2. Marco teórico **32**
2.1 Computación sensible al contexto 32
2.2 Sistemas de recomendación 34
2.3 Computación ubicua 35
2.4 Modelo del usuario y su personalización 37
2.5 Visitas guiadas contextualizadas 39
2.6 Ontologías 39
2.7 Análisis de sentimientos 41
2.8 Conclusiones del capítulo 43

3. Estado del arte **44**
3.1 Trabajos relacionados con la computación sensible al contexto 44
3.1.1 Predicción de la calidad del aire sensible al contexto habilitada por inteligencia artificial en ciudades inteligentes 44
3.1.2 Reconocimiento de actividad sensible al contexto y adaptable dinámicamente con relojes inteligentes: un estudio de caso sobre el tabaquismo 48
3.1.3 Sistemas de recomendación consientes del contexto para la salud del sueño: en forma de revisión narrativa 50
3.2 Trabajos relacionados con la computación ubicua 52

3.2.1 Un esquema teórico para diseñar entornos de aprendizaje inteligente y ubicuo para el patrimonio cultural al aire libre 52
3.2.2 Opciones de ruta de los pasajeros de autobús después de consultar información ubicua en tiempo real 55
3.2.3 Asistencia ubicua para la evacuación al aire libre basada en GIS: una solución eficaz respuesta a desastres sísmicos 57
3.3 Trabajos relacionados con los sistemas de recomendación 60
3.3.1 Un novedoso sistema de recomendación turística en el contexto del comercio social 61
3.3.2 Navegando a través del complejo sistema de transporte: un enfoque heurístico para la recomendación de turismo urbano 62
3.3.3 Sistema de recomendación descentralizado para inteligencia ambiental de juegos serios de destinos turísticos utilizando un enfoque de calificación conocido y desconocido 64
3.4 Trabajos relacionados con el modelado de visitas guiadas contextualizadas 66
3.4.1 Sistemas de recomendación basados en filtrado colaborativo para Negocios, Atracciones y Destinos Turísticos 67
3.4.2 Sistema de recomendación para turismo basado en agrupamiento semántico y análisis de sentimientos 68
3.5 Trabajos relacionados con el modelado del usuario y su personalización 70
3.5.1 Recomendaciones de puntos de interés basadas en preferencias inmediatas del usuario e influencias contextuales 70
3.5.2 Una ontología para modelar perfiles y actividades de usuarios en la educación gamificada 72
3.6 Tabla comparativa de los trabajos relacionados 74
3.7 Conclusiones del capitulo 77

4. Modelo Propuesto 80
4.1 Caracterización de los conceptos asociados al sistema de recomendación para visitas guiadas 81
4.1.1 Modelo del sistema de recomendación para visitas guiadas 81
4.1.2 Modelo del usuario y su personalización 83
4.1.3 Modelo del lugar y su adaptación 86
4.1.4 Modelo del contexto aplicado al modelo de visitas guiadas contextualizadas 88
4.2 Ontología tourist del modelo de recomendación para visitas guiadas, basado en computación ubicua y sensible al contexto 89
4.2.1 Conceptualización de la ontología 89
4.2.2 Formalización e implementación de la ontología 106
4.3 Sistemas de recomendación aplicados al modelo de recomendación para visitas guiadas, basado en computación ubicua y sensible al contexto 107
4.3.1 Modelo de filtrado colaborativo 107
4.3.2 Modelo de filtrado colaborativo basado en la calificación del usuario 112
4.3.3 Modelo de filtrado colaborativo usando datos demográficos del usuario 113
4.3.4 Modelo de filtrado colaborativo basado en clustering 114
4.3.5 Modelo de filtrado colaborativo usando descomposición en valores singulares 115
4.3.6 Modelo de filtrado colaborativo basado en sistemas de recomendación híbridos 116
4.4 Análisis de sentimientos aplicado al modelo de recomendación para visitas guiadas, basado en computación ubicua y sensible al contexto 117

4.5 Diseño de la arquitectura....119
4.6 Conclusiones del capitulo....120

5. Evaluación de los modelos propuestos a través de métricas 122
5.1 Filtrado colaborativo....124
5.2 Filtrado colaborativo basado en el usuario 125
5.3 Filtrado colaborativo usando datos demográficos del usuario 125
5.4 Filtrado colaborativo basado en clustering 126
5.5 Filtrado colaborativo basado en reducción de dimensionalidad usando descomposición en valores singulares....127
5.6 Filtrado colaborativo basado en sistemas de recomendación híbridos 128
5.7 Análisis de sentimientos aplicado al modelo propuesto....129
5.8 Conclusiones del capitulo....132

6. Conclusiones y trabajo futuro....134
6.1 Conclusiones y aportes de la tesis 134
6.2 Trabajo Futuro....138

A. Anexo: Código implementado en Python para los sistemas de recomendación aplicados al modelo 139

B. Resultados de la recomendación usando el sistema de recomendación híbrido para los casos de estudio....146

C. Código para construir el modelo de análisis de sentimientos....149

D. Definición de las instancias de la ontología tourist....153

E. Formalización e implementación de la ontología tourist....159

Bibliografía 171

Lista de figuras

Pág.

Figura 2-2-2: Categorización contextual de la información del contexto. 33
Figura 2-2-3: Proceso del modelado del Usuario. 37
Figura 2-4: Arquitectura del sistema OiUTIS 40
Figura 2-5: Arquitectura del sistema SmartSenti. 42
Figura 3-3-3: Bloques de construcción del sistema de predicción QA sensible al contexto (MyAQI) 45
Figura 3-3-4: Arquitectura del modelo de predicción del sistema MyAQI y el flujo del funcionamiento general de la predicción 46
Figura 3-3-5: Arquitectura de los sistemas CASHRS. 51
Figura 3-3-6: Arquitectura de SmartC. 54
Figura 3-3-7: Impacto potencial de los cambios en los elementos de elección en la distribución de la demanda de TP 56
Figura 3-3-9: Enfoque centralizado de gestión de la evacuación en caso de terremoto. 58
Figura 3-3-10: Enfoque descentralizado de gestión de la evacuación en caso de terremoto. 58
Figura 3-3-11: Enfoque de UbiEvac para la gestión de la evacuación en caso de terremoto. 59
Figura 3-3-13: Esquema metodológico 63
Figura 3-3-14: TDSG en la red blockchain ethereum. 65
Figura 3-3-16: Esquema general de la recomendación de las atracciones o lugares de interés. 69
Figura 3-3-21: Extracción e incorporación de características contextuales multifactoriales. 71
Figura 3-3-22: Procedimiento del estudio abordado. 73
Figura 4-1: Modelo del sistema de recomendación para visitas guiadas. 81
Figura 4-2: Modelo del perfil del usuario. 84
Figura 4-3: Modelo del lugar y su adaptación. 87
Figura 4-4: Modelo del contexto aplicado a las visitas guiadas contextualizadas. 88
Figura 4-5: Taxonomía de los conceptos que integran la ontología 90
Figura 4-6: Taxonomía de los conceptos que integran el modelo del lugar turístico 91
Figura 4-7: Taxonomía de los conceptos del modelo del usuario 91
Figura 4-8: Diagrama de relaciones binarias del modelo del sistema de recomendación 92
Figura 4-9: Diagrama de relaciones binarias del modelo de las preferencias del usuario 93
Figura 4-10: Diagrama de las relaciones binarias del modelo de las características del lugar 94
Figura 4-11: Descripción de los atributos de las entidades del modelo 97

Figura 4-12: Descripción de los atributos del modelo del lugar 99
Figura 4-13: Descripción de los atributos del modelo de las preferencias del usuario...101
Figura 4-14: Descripción de los atributos del modelo de las características del lugar ...103
Figura 4-15: Ejemplo de los 5 primeros registros de los usuarios. 109
Figura 4-16: Ejemplo de los 2 primeros registros de los lugares. 110
Figura 4-17: Ejemplo de los 3 primeros registros de los ratings de los usuarios. 111
Figura 4-18: Matriz de puntajes 112
Figura 4-19: Matriz de similitud del coseno para los usuarios. 113
Figura 4-20: Ejemplo de las 5 primeras reseñas a lugares visitados.....118
Figura 4-21: Ejemplo de las frecuencias de las palabras en un texto de ejemplo 119
Figura 5-1: Evaluación del filtrado colaborativo modelo de referencia 124
Figura 5-2: Evaluación del filtrado colaborativo basado en el usuario usando las medias 125
Figura 5-3: Resultados del F1-score y RMSE usando datos demográficos del usuario 126
Figura 5-4: Resultado de la métrica RMSE usando clúster KNN para 5 pliegues 127
Figura 5-5: Métricas de desempeño usando clúster KNN para 5 pliegues.....127
Figura 5-6: Resultado de implementar el modelo en descomposición en valores singulares.....128
Figura 5-7: Resultados al aplicar el modelo híbrido usando la técnica de validación cruzada con 5 pliegues.....129
Figura 5-8: Ajuste del Accuracy del modelo.....129
Figura 5-9: Ajuste del accuracy para el parámetro C. 130
Figura 5-10: Resultados matriz de confusión.....131
Figura 5-11: Resultados de las métricas después del ajuste. 131
Figura 5-12: Resultados de la predicción de análisis de sentimientos para 4 reseñas. .132

Lista de tablas

Pág.

Tabla 1-1: Etapas de la metodología empleada....27
Tabla 2-1: Taxonomía de las aplicaciones basadas en la computación ubicua....36
Tabla 3-3-1: Tabla comparativa de los trabajos relacionados....75
Tabla 4-1: Glosario de términos de la ontología....89
Tabla 4-2: Descripción de las relaciones binarias del modelo del sistema de recomendación. 94
Tabla 4-3: Descripción de las relaciones binarias del modelo de las preferencias del usuario....95
Tabla 4-4: Descripción de las relaciones binarias del modelo de las características del lugar....95
Tabla 4-5: Diccionario de los conceptos que constituyen la ontología. 95
Tabla 4-6: Descripción de los atributos en detalle de las instancias del modelo 97
Tabla 4-7: Descripción de los atributos en detalle del modelo del lugar....99
Tabla 4-8: Descripción de los atributos en detalle del modelo de las preferencias del usuario....102
Tabla 4-9: Descripción de los atributos en detalle del modelo de las características del lugar....104
Tabla 4-10: Regla para buscar los lugares que tengan eventos en la fecha y hora del contexto temporal 104
Tabla 4-11: Regla para buscar los lugares que cumplen con el contexto temporal y luego buscar que tengan el tópico del usuario con id: id....105
Tabla 4-12: Regla para buscar los lugares que correspondan con las preferencias genéricas del usuario: la hora de salida, de llegada, su presupuesto y su fecha propuesta para salir....105
Tabla 4-13: Regla para buscar los lugares que no tengan restricciones para menores de edad....106

Lista de abreviaturas

Abreviatura	Término
Aml	Inteligencia ambiental
RS	Sistemas de recomendación
CARS	Sistemas de recomendación consientes del contexto
POI	Punto de interés
LBSN	Redes sociales basadas en la ubicación
MCS	Sistemas de detección de multitudes

1. Introducción

El propósito de esta investigación consiste en buscar nuevas formas de usar la información que proporcionan las tecnologías móviles en los sistemas de información, para buscar expandir el rango de posibilidades y acciones al usar la información del contexto en que se encuentra el usuario y sobre el que opera el dispositivo donde funciona el sistema. Las personas pueden hacer uso de los entornos que estén acondicionados para soportar tecnologías de redes inalámbricas (Restrepo Medina, 2012), no solo para conectarse con dispositivos móviles sino para aprovechar las ventajas y aspectos novedosos de los enfoques de computación ubicua e inteligencia artificial que se planean integrar en el modelo de esta de investigación.

1.1 Contextualización

En los últimos años han surgido nuevos enfoques que proponen el uso de un mayor número de características y propiedades que consideran la información del contexto, como es el caso de la inteligencia ambiental que tiene como objetivo el crear espacios inteligentes donde el entorno se adapte a las exigencias y necesidades de las personas (Restrepo Medina, 2012). La información que proviene de fuentes de datos como el contexto tiene una relación intrincada que puede ser desconcertante para las personas, los sistemas de recomendación son herramientas de filtrado de información muy útiles para ayudar de manera personalizada a descubrir la información que puede ser de interés dentro de un espacio de posibles opciones, lo cual facilita el proceso de toma de decisiones (Zhang et al., 2016).

Las tecnologías móviles permiten a las personas estar libremente en muchas partes conectadas accediendo a varios servicios, esto se ha vuelto parte de los espacios que

normalmente concurrimos y usamos diariamente. Existe un paradigma llamado computación sensible al contexto que aprovecha esta característica de los dispositivos tomando la información del ambiente que rodea dicho dispositivo, la cual está cambiando constantemente, con el fin de adaptarla según la ubicación de uso, según la información proveniente de los objetos y personas cercanas, y considerando los cambios de dichos objetos en el tiempo (Restrepo Medina, 2012). La información capturada se conoce como información de contexto y es de importancia porque permite el descubrimiento de nuevas oportunidades de uso de las tecnologías móviles como elementos que se pueden integrar en diversos contextos de forma colaborativa e inteligente.

Estas nuevas características que ofrecen las tecnologías móviles son parte de un comportamiento muy interesante que tiene por objetivo lograr que la interacción de la persona con el dispositivo y el sistema sea invisible, lo cual cambia la forma de ver y usar dichos dispositivos. Este comportamiento se conoce como computación ubicua (Santos, 2013). La computación sensible al contexto aplicado a los sistemas de información es un objeto de estudio el cual cubre un amplio espacio multidisciplinario de investigación que lo ha convertido en una herramienta para comprender y crear modelos de sistemas que se adaptan a las necesidades y perfiles de los usuarios con el fin de proporcionar servicios adecuados al usuario dentro de ambiente dinámico. Adicionalmente, estos modelos se pueden aplicar de forma transversal a varios dominios de interés en sistemas que hagan uso de las ventajas y características que ofrecen las tecnologías móviles en forma personalizada.

1.2 Justificación

Con los enfoques y paradigmas mencionados anteriormente, pertenecientes a la computación móvil y a la inteligencia artificial, se busca integrarlos con el fin de proponer un modelo de recomendación para visitas guiadas que mejoren la interacción de las personas con el contexto circundante buscando que el sistema se adapte al entorno y al usuario en tiempo real y responda de forma inteligente y adaptativa. Con el modelo de esta propuesta se planea desarrollar prototipos funcionales aplicados en la búsqueda de soluciones a problemas de diferentes categorías de interés, como las siguientes:

- Visitas escolares guiadas a museos orientadas al aprendizaje.
- Visitas guiadas que necesitan orientar la motivación de los visitantes mostrando sitios e información diferente al agotado producto turístico tradicional como son la riqueza patrimonial, arquitectónica y paisajística del lugar y que lo diferencia de otros sitios.
- Se propone abordar el tema de la gestión de los trámites en una institución prestadora de servicios ya que los gerentes aseguran que la gestión se puede facilitar y mejorar, mediante el uso de celulares y correos electrónicos, la asignación de citas y recursos a los usuarios en forma oportuna.

1.3 Hipótesis

Las posibilidades de interacción que tienen las personas con los sistemas de información pueden ser ampliadas y mejoradas a través de un modelo de sistema de recomendación usando las tecnologías móviles y técnicas de inteligencia artificial que faciliten el proceso de toma de decisiones al usar información del contexto basada en la computación ubicua y sensible al contexto para capturar datos, crear modelos e inferir conocimiento con el fin de razonar y actuar según los objetivos y necesidades de las personas.

1.4 Problema de Investigación

Una de las muchas posibilidades de interacción que tienen las personas con los sistemas de información es en el área de educación. Cuando las tecnologías móviles están disponibles y son ubicuas, los estudiantes cuentan con posibilidades de acceder, compartir y construir conocimiento fácilmente en varios lugares con adaptaciones diferentes. Se hace necesario el diseñar y crear nuevos modelos que faciliten esta nueva forma de aprendizaje y que sean diferentes al aprendizaje adquirido con los sistemas tradicionales utilizados en computadores de escritorio. La computación ubicua es una alternativa para proporcionar una noción de aprendizaje motivada por la movilidad, ubicuidad y la sensibilidad al contexto apoyada en el uso de las tecnologías móviles. Esta forma de

aprendizaje sucede en un contexto específico y fomenta un aprendizaje más profundo y significativo. Hay muchas promesas y potencial de la computación móvil y ubicua aplicada al área del aprendizaje, este campo todavía es poco estudiado. En términos de implementación hay todavía mucho terreno por recorrer, también en términos de reducir la barrera de adopción y el uso sostenido en prácticas de aprendizaje, al igual que su posible (Kano & Nakajima, 2018).

El uso de los dispositivos móviles es común en nuestra vida diaria, pero el usarlos para la educación y el aprendizaje todavía puede tomar un largo tiempo en nuestro medio.
La tarea de un sistema de recomendación es ofrecerle al usuario solo lo que es relevante para él y el concepto de información relevante guarda una relación entre el perfil del usuario y el contenido del objeto de interés(Ansari, s/f). Es difícil determinar qué es relevante para el usuario y para ello se deben definir algunas medidas o factores que ayuden a determinar esa relevancia, por ejemplo:

- La información más buscada y actualizada del objeto de interés.
- La información específica del objeto de interés, como el de más bajo costo y mejor calidad.
- La información de un objeto que pertenezca a otra persona con características similares a las de la persona que está buscando el objeto.
- La información de un objeto similar que el usuario ya haya intentado buscar en el pasado.

El problema aquí radica en la necesidad de tomar muchas decisiones para llegar a una buena recomendación. Y es en este punto donde la integración de computación ubicua y sensible al contexto con las técnicas de Inteligencia Artificial son de utilidad para obtener información de la persona y del contexto, y no únicamente de un conjunto de datos basados solo en las preferencias y el perfil de la persona.

1.5 Objetivo general

Desarrollar un modelo de sistema de recomendación personalizado ubicuo y sensible al contexto para la toma de decisiones en visitas guiadas.

1.6 Objetivos específicos

- Caracterizar los elementos de la computación ubicua y sensible al contexto, y de los sistemas de recomendación que se requieren con el fin de integrarlos en un modelo de recomendación para visitas guiadas, basado en computación ubicua y sensible al contexto.
- Identificar las metodologías empleadas para integrar las preferencias del contexto con las preferencias del perfil del usuario que permitan diseñar un sistema de recomendación sensible al contexto.
- Diseñar el modelo del sistema de recomendación ubicuo y sensible al contexto basado en los elementos analizados anteriormente.
- Desarrollar e implementar un prototipo funcional basado en el modelo propuesto aplicado a varios casos de estudio.
- Evaluar a través de métricas el desempeño del prototipo implementado aplicado a los casos de estudio.

1.7 Alcance

Elementos y características a considerar en el modelo:

- En el componente sensible al contexto del modelo propuesto, para el contexto del usuario se considerará el contexto relacionado con las preferencias y el estado emocional del usuario.
- En el componente sensible al contexto del modelo propuesto, para el contexto del ambiente se considerarán las funcionalidades y prestaciones que se encuentren en los dispositivos móviles y puedan incorporarse para la adquisición de la información del contexto.
- Se creará un modelo de sistema de recomendación para visitas guiadas que aplique computación sensible al contexto en dispositivos móviles.

- El modelo contará con un componente de personalización basado en el modelo del usuario y el modelo de los sitios de interés.
- En las pruebas del prototipo se considerarán dos o tres casos de prueba aplicados a visitas guiadas en el ámbito educativo, de turismo o de servicios.
- Se implementará el prototipo funcional del sistema de recomendación basado en el modelo sensible al contexto personalizado al usuario y a los puntos de interés.
- Se realizará una evaluación del desempeño del prototipo basado en métricas aplicadas a los casos de estudio.

Elementos y características que no se considerarán en el modelo:

- El modelo propuesto no considera los aspectos relacionados con la planificación de rutas para recomendarlas en los recorridos de una visita guiada, solo recomendará puntos de interés que concuerden con la información del contexto y del usuario en forma personalizada.
- En el componente sensible al contexto del modelo propuesto, para el contexto del usuario no se considerará el contexto relacionado con el comportamiento dinámico y el estado fisiológico del usuario.
- En el componente sensible al contexto del modelo propuesto, para el contexto del ambiente no se considerará aplicar técnicas de visión artificial y tampoco internet de las cosas.
- En el sistema de recomendación no se van a considerar los aspectos relacionados con la gestión de la privacidad del usuario.

1.8 Metodología

Para el desarrollo del proyecto se consideran seis etapas orientadas a la consecución de los objetivos a través de la ejecución de varias actividades establecidas previamente. En las diferentes etapas se utilizarán los métodos y las técnicas pertenecientes a las áreas de la computación sensible al contexto, los sistemas de recomendación, la computación ubicua, el modelo del usuario y su personalización, y las visitas guiadas conceptualizadas. Todo lo anterior con el fin de proponer un modelo que sirva de base para construir un

prototipo funcional aplicado a varios casos de estudio y sobre los cuales se puedan realizar pruebas de validación.

A continuación, se relacionan las etapas que conforman la metodología de trabajo, etapas que corresponden a los objetivos específicos propuestos y son las actividades que conforman el ciclo de vida de desarrollo del software (**¡Error! No se encuentra el origen e la referencia.**).

Tabla 1-1: Etapas de la metodología empleada

Etapa	Descripción	Objetivo	Actividades
1	Elaboración del marco teórico, y se realizó la revisión del estado del arte de cada una de las tecnologías y áreas requeridas para la elaboración del modelo de recomendación para visitas guiadas.	Caracterizar los elementos, las formas en que estos se relacionan, las metodologías y las arquitecturas usadas en las siguientes áreas y tecnologías, y se integrarán en un modelo de recomendación para visitas guiadas: - La computación sensible al contexto. - La computación ubicua. - Los sistemas de recomendación. - El modelado, perfilado y personalización del usuario. - Las visitas guiadas contextualizadas.	Se realizó la revisión bibliográfica de cada una de las tecnologías y áreas requeridas en la elaboración del modelo de recomendación para visitas guiadas. Se realizó la revisión de las metodologías empleadas para evaluar los modelos de los prototipos en cada una de las tecnologías y áreas requeridas en la elaboración del modelo de recomendación para visitas guiadas. Se realizó la revisión bibliográfica sobre metodologías de la computación sensible al contexto, sobre los modelos y aplicaciones que apliquen la computación sensible al contexto en visitas guiadas. Se realizó la revisión bibliográfica sobre las metodologías empleadas para integrar el contexto con los sistemas de recomendación.
2	Análisis y diseño del componente	Identificar las metodologías empleadas para integrar las preferencias del contexto con	Se determinaron y diseñaron los modelos del usuario y el modelo de los puntos de

Etapa	Descripción	Objetivo	Actividades
	sensible al contexto del modelo con la información de los sitios de interés y del usuario para el componente de visitas guiadas que se integrará con el sistema de recomendación.	las preferencias del perfil del usuario y aplicarlas en el diseño de un sistema de recomendación sensible al contexto.	interés que se requieren para desarrollar el componente sensible al contexto. Se determinó la técnica a emplear para desarrollar la personalización de las recomendaciones según la información y el modelo del usuario y de los puntos de interés de la visita guiada.
3	Análisis y diseño del componente de recomendación del modelo.	Diseñar el modelo conceptual del sistema de recomendación ubicuo y sensible al contexto con la ayuda de los elementos analizados en las etapas anteriores. Usar metodologías para integrar el contexto con el sistema de recomendación y obtener un modelo de sistema de recomendación sensible al contexto.	Se caracterizaron los elementos que componen un sistema de recomendación. Se determinaron las técnicas que se emplearon en el componente de recomendación: - Técnicas de Inteligencia Artificial - Computación Ubicua. - Sensibilidad al contexto. - Análisis de sentimientos
4	Integrar los componentes diseñados en las etapas anteriores para completar el diseño del modelo del sistema de recomendación. Se integró el componente sensible al contexto con el componente de recomendación y se obtuvo el diseño del modelo implementado	Implementar el diseño del modelo de recomendación sensible al contexto, considerando los componentes diseñados en las etapas 2 y 3.	Se aplicó la metodología seleccionada en la integración del contexto con el modelo de recomendación sensible al contexto. Se aplicó la metodología seleccionada para el diseño del modelo de recomendación híbrido y sensible al contexto. Se aplicó la metodología seleccionada para la personalización en el diseños del modelo de recomendación sensible al contexto.

Etapa	Descripción	Objetivo	Actividades
	del sistema de recomendación sensible al contexto.		
5	Desarrollo e implementación del modelo propuesto.	Desarrollar un prototipo de sistema de recomendación híbrido que se puede aplicar a dos o tres casos de estudio haciendo uso del modelo propuesto.	Se caracterizó las tecnologías más apropiadas para la implementación del modelo de sistema de recomendación híbrido. Se realizó la implementación de los modelos de usuario y de los puntos de interés, utilizando tecnologías para el prototipo de sistema de recomendación híbrido. Se realizó la implementación del componente de recomendación del modelo usando las tecnologías que permitieron aplicar las técnicas de inteligencias artificial seleccionadas. Se mplementó la integración de los componentes del modelo usando las tecnologías que permitan aplicar las técnicas de inteligencia artificial seleccionadas. Se desplegó el prototipo desarrollado y se realizaron unas pruebas iniciales con los casos de estudio.
6	Evaluación del prototipo funcional	Evaluar a través de métricas el desempeño del prototipo implementado para los casos de estudio considerados.	Se caracterizaron las métricas apropiadas para la evaluación del prototipo. Se establecieron los casos de estudio necesarios para la evaluación. Se recopilo la información necesaria para la evaluación.

Etapa	Descripción	Objetivo	Actividades
			Se realizó la evaluación del prototipo funcional. Se realizó un análisis de los resultados obtenidos de las pruebas para el desempeño del prototipo en los casos de estudio considerados.

1.9 Contribuciones

El trabajo inicia con el desarrollo de una ontología elaborada en Python, la cual incluye los elementos del usuario, el sitio de interés, las preferencias del usuario, las características del sitio, la información del contexto temporal, ambiental y geográfico del usuario y del sitio. La ontología se llama tourist y permite hacer inferencias sobre reglas que validan los escenarios propuestos en el modelo. La ontología sirve de línea base para construir un modelo en cualquier lenguaje de programación y adaptarlo para hacer trabajos de investigación en computación sensible al contexto y a técnicas de aprendizaje automático como análisis de sentimientos.

El modelo desarrollado a partir de la ontología fue implementado en varias propuestas de modelos que aplican la teoría de los sistemas de recomendación, y se implementaron modelos de sistemas de recomendación basados en conocimiento, basados en el usuario, basados en los lugares, en sistemas de recomendación que usan técnicas de clustering y de descomposición en valores singulares. Finalmente, el trabajo culmina con un sistema de recomendación híbrido que arrojó los mejores resultados en la evaluación de las métricas de interés para hacer predicciones de los sitios sugeridos y medir la calidad de dichas predicciones. Un modelo de técnica de machine learning para analizar los sentimientos de los comentarios de los usuarios sobre los lugares también fue implementado y evaluado.

1.10 Organización del documento

El documento está compuesto de los siguientes capítulos; el capítulo 2 con el marco teórico que presenta los conceptos generales de los paradigmas de la computación y que fundamentan el desarrollo de este trabajo. El capítulo 3, presenta la revisión del estado del arte, donde se describen los trabajos relacionados con los paradigmas de la computación abordados. El modelo propuesto basado en la ontología tourist, se presenta en el capítulo 4, en este capítulo se muestran los componentes de los siguientes modelos: visitas guiadas, del usuario y su personalización, de los sitios de interés y sus características, y de la información del contexto. En este mismo capítulo se muestra la implementación de los modelos de sistemas de recomendación basados en filtrado colaborativo y una aplicación del modelo a la técnica de aprendizaje de máquina de análisis de sentimientos para determinar el nivel de percepción del sitio de interés visitado por el usuario. En el capítulo 5 se presenta la evaluación de los modelos presentados en el capítulo anterior. Y finalmente, en el capítulo 6 se presentan las conclusiones y trabajo futuro. Al final se incluye una sección con los anexos a los materiales que respaldan los paradigmas y conceptos que sustentan esta tesis de maestría.

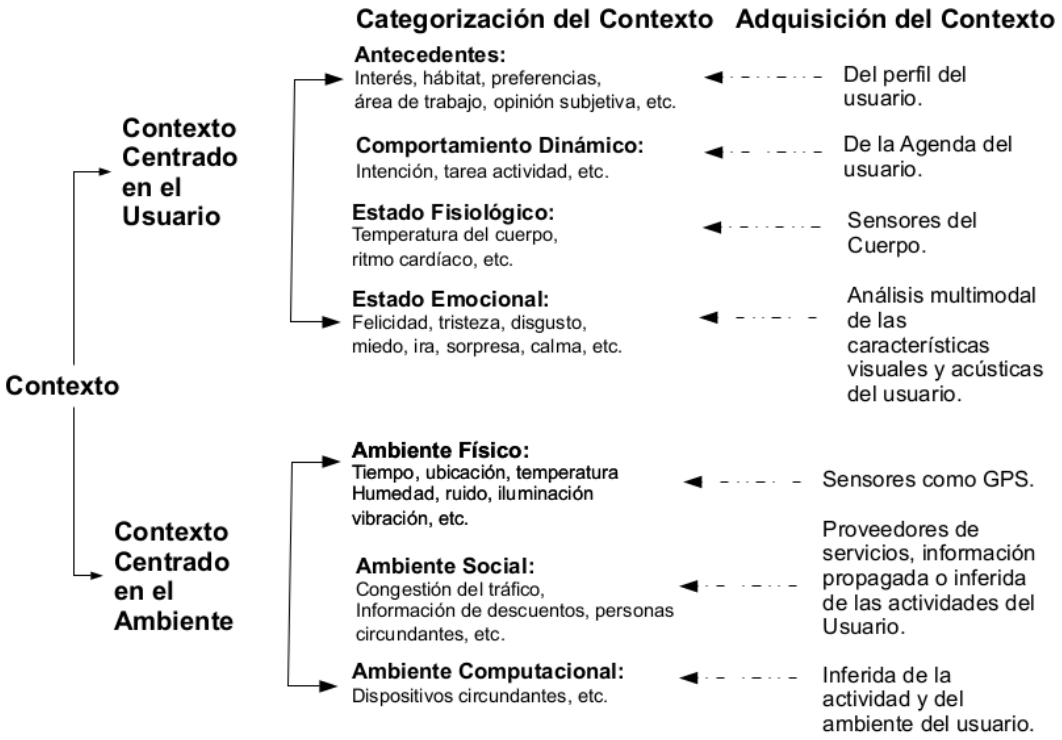
2. Marco teórico

Este capítulo tiene como objetivo presentar los conceptos más relevantes relacionados con la investigación que soportan el modelo de recomendación, a saber: Computación Sensible al Contexto, Sistemas de recomendación, Computación Ubicua, Modelos del Usuario y su personalización, Visitas Guiadas Contextualizadas, Ontologías y Análisis de sentimientos.

2.1 Computación sensible al contexto

El desarrollo en paralelo de la computación ubicua y de las interfaces inteligentes amigables al usuario, han conducido en el mundo, a que la funcionalidad y la capacidad de los computadores haya sido embebida en el interior de muchas clases de objetos, dotándolos con la capacidad de reconocer y responder a las necesidades individuales de los usuarios en forma adecuada y transparente. Este comportamiento se conoce como inteligencia ambiental o AmI (por sus siglas en inglés) según Abowd et al, (2002) que tiene como objetivo hacer que lleguen las tecnologías de la información a todos, a cada hogar, a cada escuela, y a cada negocio.

Figura 2-2-1: Categorización contextual de la información del contexto.



Fuente: (Feng et al., 2004).

El contexto es un componente de esencial importancia en la AmI. Bickerton (2017) veía el contexto desde la perspectiva tradicional de un programa en el campo de las ciencias de la computación, que trata de ser independiente del contexto: para la misma entrada se proporciona la misma salida, independiente del contexto de la entrada. Así, ellos proponen una definición concreta de contexto. “Contexto puede ser considerado todo lo que afecta el proceso de cálculo y ejecución de un programa a excepción de sus entradas y salidas explicitas”. Según Dey (2001) y Chalmers (2004), un sistema es sensible al contexto si usa el contexto para proporcionar información y servicios relevantes al usuario, donde la relevancia depende de la tarea del usuario. Desde el punto de vista de la aplicación, la información sirve para caracterizar la situación de una entidad, esta entidad puede ser considerada relevante en la interacción entre el usuario y la aplicación. La entidad puede ser una persona, lugar, objeto, el mismo usuario o la misma aplicación.

En el trabajo de Abowd et al. (2002), se define el contexto desde la perspectiva de la gestión de los datos, contexto es el objetivo del ambiente externo del usuario y su estado subjetivo interno. Esto requiere conocer la información que necesita el usuario durante su

interacción con el sistema de gestión de los datos. De esta forma la sensibilidad al contexto puede ser definida como el uso del contexto en servicio de las necesidades de información del usuario. Una distinción del contexto de manera conceptual según Feng et al, (2004) es el contexto centrado en el usuario y el contexto centrado en el ambiente (Figura 2-2-2).

2.2 Sistemas de recomendación

Los sistemas de recomendación RS (por sus siglas en inglés), surgieron como una solución al problema de la sobrecarga que se presenta cuando se filtra la información según las necesidades específicas del usuario. Los RS pueden ser vistos como mediadores de la experiencia del usuario en la búsqueda de información en un contexto digital y son de gran ayuda para hacer la misma función en el mundo real al momento de la toma de decisiones (Mcdonald, s/f)

Los primeros sistemas de recomendación que surgieron fueron los sistemas de recomendación basados en contenido, con los que se buscaba definir un espacio común para representar ciertas características pertenecientes a los elementos considerados y a los usuarios que desean recomendación sobre estos elementos. Para un usuario en particular, los elementos similares a los de su perfil son recomendados sin considerar la información de otros usuarios. Por otra parte, los sistemas de recomendación basados en filtrado colaborativo, buscan usar las preferencias y elecciones de usuarios similares a un usuario en particular como base para generar la recomendación (Villegas et al., 2018).

Los sistemas de recomendación basados en contenido son fáciles de entender y demuestran ser un buen punto de partida para moverse entre los elementos que componen el sistema y permiten obtener recomendaciones para nuevos usuarios o nuevos elementos añadidos, esto se conoce como el problema de arranque en frio. Sin embargo, requieren de la tarea incomoda de realizar una descripción detallada de todos los elementos con solo un conjunto común de características, no funciona con información o contenido dado a entender, solo puede manejar recomendaciones de elementos complementarios y, como

se centra en un solo usuario, no permite hacer recomendaciones de elementos inesperados.

Los sistemas de recomendación de filtrado colaborativo se basan en las preferencias comunes de muchos usuarios para obtener recomendaciones complementarias y de elementos substitutos, pero también pueden sorprender a los usuarios con recomendaciones de elementos inusuales. Sin embargo, necesitan más datos de los usuarios para funcionar bien, y no brindan una alternativa al problema del arranque en frio.

Un tercer tipo de sistema de recomendación, son los sistemas de recomendación híbridos, son un punto medio entre los dos sistemas de recomendación mencionados anteriormente. Los sistemas de recomendación hechos a medida para influenciar la información del contexto (como la ubicación, el tiempo, o los acompañantes) durante el proceso de recomendación son conocidos como sistemas de recomendación sensibles al contexto o CARS (por sus siglas en inglés) para context-aware recommended systems (Raza & Ding, 2019).

Los sistemas de recomendación usan la información de elementos, usuarios, y preferencias, siendo la principal fuente de información la matriz de elementos por la matriz de usuarios que contiene las informaciones de las preferencias del usuario por cada elemento individual.

2.3 Computación ubicua

En este particular momento de innovación, en lo que concierne a las tecnologías de información, la computación ubicua, constituye una innovación basada en una red que conecta a todos los dispositivos, y se considera como la fusión de tecnologías en el espacio físico y electrónico (Jeon et al., 2007).

Tabla 2-1: Taxonomía de las aplicaciones basadas en la computación ubicua

Criterio	Característica	Descripción
Sujeto	Generalidad	Característica en la cual la mayoría de la sociedad usa la computación ubicua a través de ambientes dotados con capacidad de acceder a la información y realizar y operaciones de computo de forma autónoma por medio de interfaces de acceso.
	Especialidad	Característica en la cual individuos o grupos específicos usan la computación ubicua para la consecución de un propósito específico.
Tiempo	Siempre	Característica en la cual en cualquier momento la conectividad a la red está garantizada.
	Temporalmente	Característica que es usada en un momento específico por una solicitud temporal del usuario de acuerdo a la construcción definida por la computación ubicua.
Lugar	Movilidad	Característica en la cual la conectividad a la red está garantizada en cualquier lugar para que el usuario haga uso de la computación ubicua.
	Fijación	Característica en la cual un ambiente con computación ubicua puede usarse en un lugar específico.

Fuente: (Jeon et al., 2007).

La computación ubicua busca que se pueda acceder a la información desde cualquier lugar: en las paredes, en su muñeca y en elementos desechables dispuestos a ser usados cuando se los necesite. Su objetivo es la disponibilidad no intrusiva de los computadores en todo el ambiente de forma invisible para el usuario (Dourish & Damasceno, 2016).

En el trabajo realizado por Jeon et al, (2007) se menciona que Mark Weiser define las principales características de la computación ubicua de la siguiente forma. Primero, cada dispositivo se puede conectar a una red. Segundo, los usuarios no tienen que estar enterados de este proceso de conexión. Tercero, esto sucede por medio de una interfaz amigable a los humanos, y servicios apropiados a la situación son proporcionados en la red en el momento adecuado.

Cabe señalar que la computación ubicua considera que los recursos de computo están disponibles de forma omnipresente, en la vida diaria y asume el propósito de ofrecer a las personas y aplicaciones la información necesaria para las más diversas necesidades

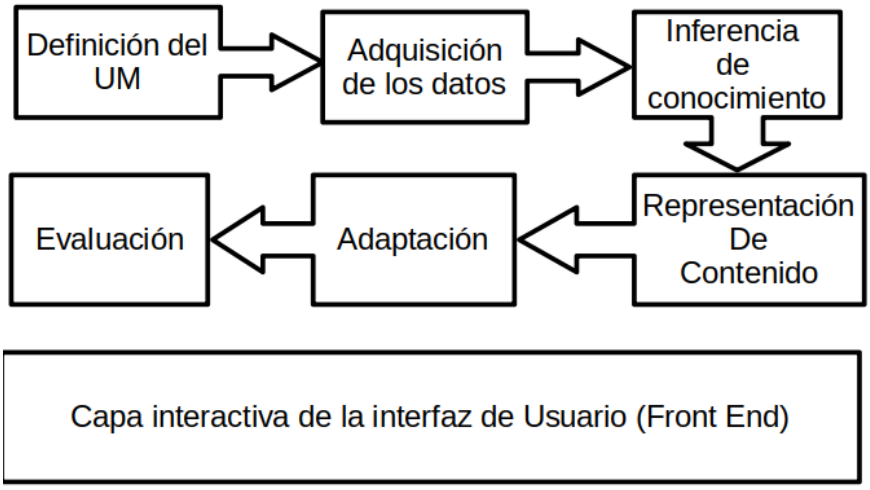
(Chalmers, 2004). La computación ubicua está enmarcada en un ambiente con las características de cualquier lugar, momento y persona, por lo que se define la siguiente taxonomía para una aplicación o servicio que funcione con la computación ubicua (Tabla 2-1):

- Sujeto: Debería ser fácil de usar por cualquier persona.
- Tiempo: Debería ser usada en cualquier momento.
- Lugar: Debería ser usada en cualquier lugar.

2.4 Modelo del usuario y su personalización

Un Modelo del Usuario o UM (por sus siglas en Inglés) en Inteligencia Artificial es una estructura de datos con las características particulares de un usuario en un momento determinado del tiempo(Cena et al., 2019). El modelado del usuario es el proceso de creación, actualización y mantenimiento de un UM (Brusilovsky, 1996).

Figura 2-2-2: Proceso del modelado del Usuario.



Fuente: (Cena et al., 2019).

En la identificación elaborada por (Cena et al., 2019) se hace una identificación de las fases que componen el proceso del modelado del usuario (Ver figura 2-2-4):

- Definición del modelo del Usuario.
- Adquisición de los datos.
- Inferencia de conocimiento de los datos.
- Representación del contenido del modelo del Usuario.
- Adaptación basada en los datos del Modelo.
- Evaluación del Modelo del Usuario.

La computación ubicua busca desarrollar infraestructuras ricas que permitan el acceso a las tecnologías de la información con una adecuada capacidad de computo, que de manera discreta y automática apoye y promueva la vida diaria de los seres humanos(Madeira, 2012). Un ambiente dominado por este tipo de computación debe ser centrado en los usuarios que lo habitan, lo cual demanda la necesidad de interfaces personalizadas y adaptativas que sean acorde al contexto. Por lo tanto, la personalización es un elemento importante en los objetivos de la computación ubicua con el fin de ofrecer las interfaces inteligentes para la interacción con sus usuarios.

La personalización del usuario puede ser definida como un método “para proveer a los usuarios con lo que quieren o necesitan sin tener que llamarlos para preguntarles sus requerimientos de manera explícita” (Chang et al., 2001). La forma de interacción de las personas con la información ha cambiado debido al hecho de que las tecnologías de la computación ubicua y móvil se han integrado en la vida diaria. La rapidez con que la información y los servicios han penetrado e inundado la vida de las personas hace que surja la necesidad de la personalización y la creación de modelos de usuario que apliquen las ideas de la computación ubicua y móvil.

Un servicio personalizado tiene la capacidad de especializarse para que las interacciones con cada uno de sus usuarios sucedan de acuerdo a sus preferencias, gustos, hábitos u otras características. Esto resulta en una relación de tipo uno-a-uno entre el servicio y el usuario, que es muy diferente al modelo tradicional uno-a-muchos. Los beneficios pueden ser para el usuario, la mejora de la experiencia de uso del servicio y para los que proporcionan el servicio porque contarían con la capacidad de ajustarse de manera precisa a las necesidades de los usuarios (Potonniée, 2002). Este autor propone como solución

considerar dos características para la personalización: la administración eficiente de la privacidad y la presencia ubicua, y no solo depender de los perfiles del usuario compuestos con información nominativa, social o de comportamiento.

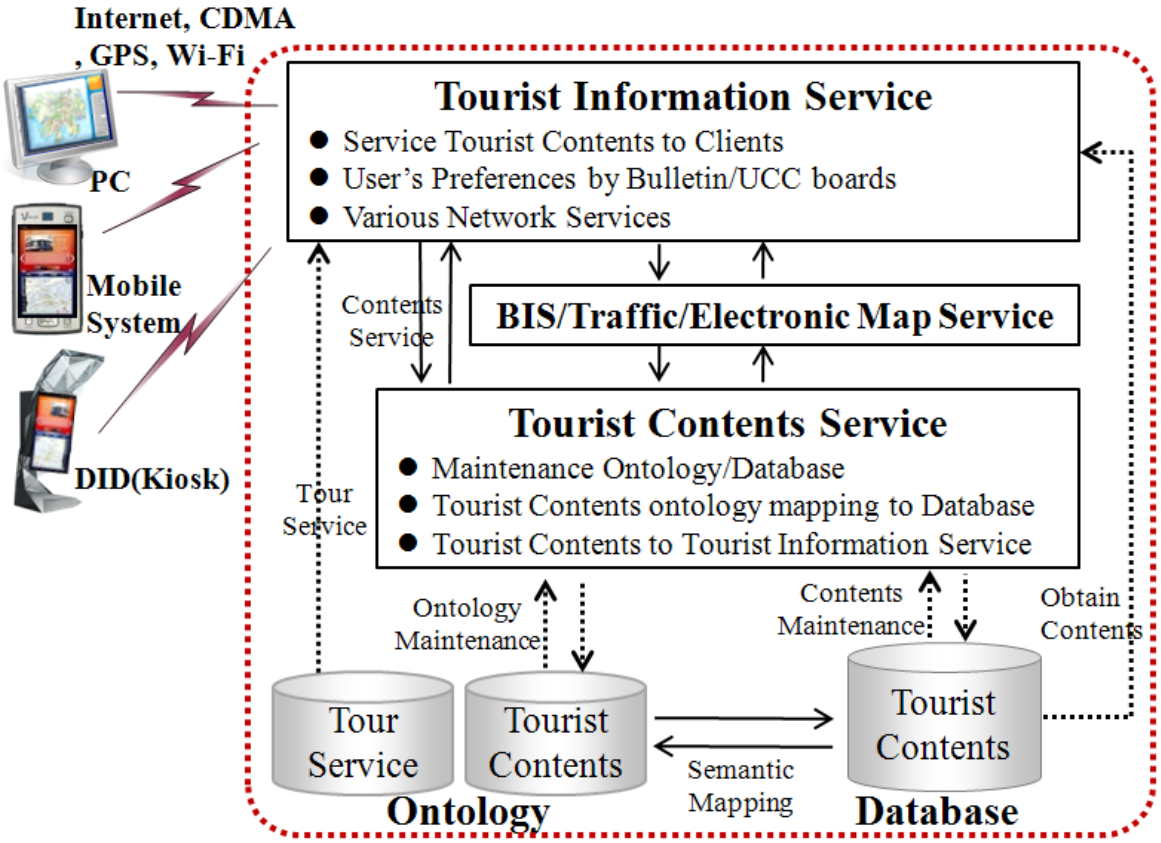
2.5 Visitas guiadas contextualizadas

Las redes sociales basadas en la ubicación o LBSNs (pos sus siglas en inglés) para locations-based social networks según el trabajo de Jiao et al , (2019) con sus diferentes aplicaciones de servicios han sido objeto de un rápido desarrollo impulsado por las tecnologías del Internet móvil, de posicionamiento, de los sensores inalámbricos y la popularidad de los teléfonos inteligentes. Según Hu et al. (2015), los puntos de interés o POIs (por sus siglas en Inglés) para Point of interests son la representación de una ubicación individual (ej., un restaurante o punto de referencia) que son de interés para las personas (McKenzie et al., 2015). Las recomendaciones basadas en POIs son una de las tareas más importantes en los LBSNs, la cual consiste en recomendar lugares a los usuarios. Sin embargo, los sistemas generales basados en la recomendación de POIs, solo pueden hacer recomendaciones globales e ignorar el tiempo del contexto que tendría el resultado de la recomendación (Jiao et al., 2019).

2.6 Ontologías

Muchos estudios acerca de enfoques ontológicos para servicios de información de turismo inteligente han sido realizados. Pero son pocos los estudios que consideran los servicios del recorrido y las preferencias del usuario usando una ontología. En el trabajo de Park et al. (2009), se presenta una ontología aplicada al área de información turística, para una plataforma con varios servicios turísticos inteligentes, en un área local usando las siguientes ontologías: ontología de contenidos turísticos y ontología de servicios turísticos, además de aplicar tecnología ubicua y un enfoque de entornos web.

Figura 2-3: Arquitectura del sistema OiUTIS



Fuente: (Park et al., 2009)

La palabra ontología, fue sugerida en 1992 por Tom Gruber, quien la definió como "una especificación de una conceptualización". En filosofía se refiere al sujeto de la existencia en el sentido de que describe los conceptos y relaciones que pueden existir para un agente o una comunidad de agentes. Esta definición es consistente con el uso de la ontología como un conjunto de definiciones de conceptos, pero es más general que él. Las primitivas representaciones suelen ser clases, atributos y relaciones. Dichas definiciones pueden incluir información sobre significado y restricciones sobre su aplicación lógicamente consistente (Gruber, 1993).

Usando las ontologías anteriormente mencionadas los autores diseñaron un sistema de información turística inteligente y ubicua (OiUTIS) basado en ontologías y en el sistema del Servicio de Información turística de la ciudad metropolitana de Busan (TISB). La arquitectura del sistema OiUTIS consta de cuatro partes: 1) un sistema de Servicio de Contenido Turístico, 2) un sistema de Servicio BIS/Tráfico/Mapas Electrónicos, 3) un

sistema de Servicio de Información Turística y 4) un sistema de Cliente. La Figura 2-4 muestra la arquitectura del sistema OiUTIS y presenta las funciones de los subsistemas.

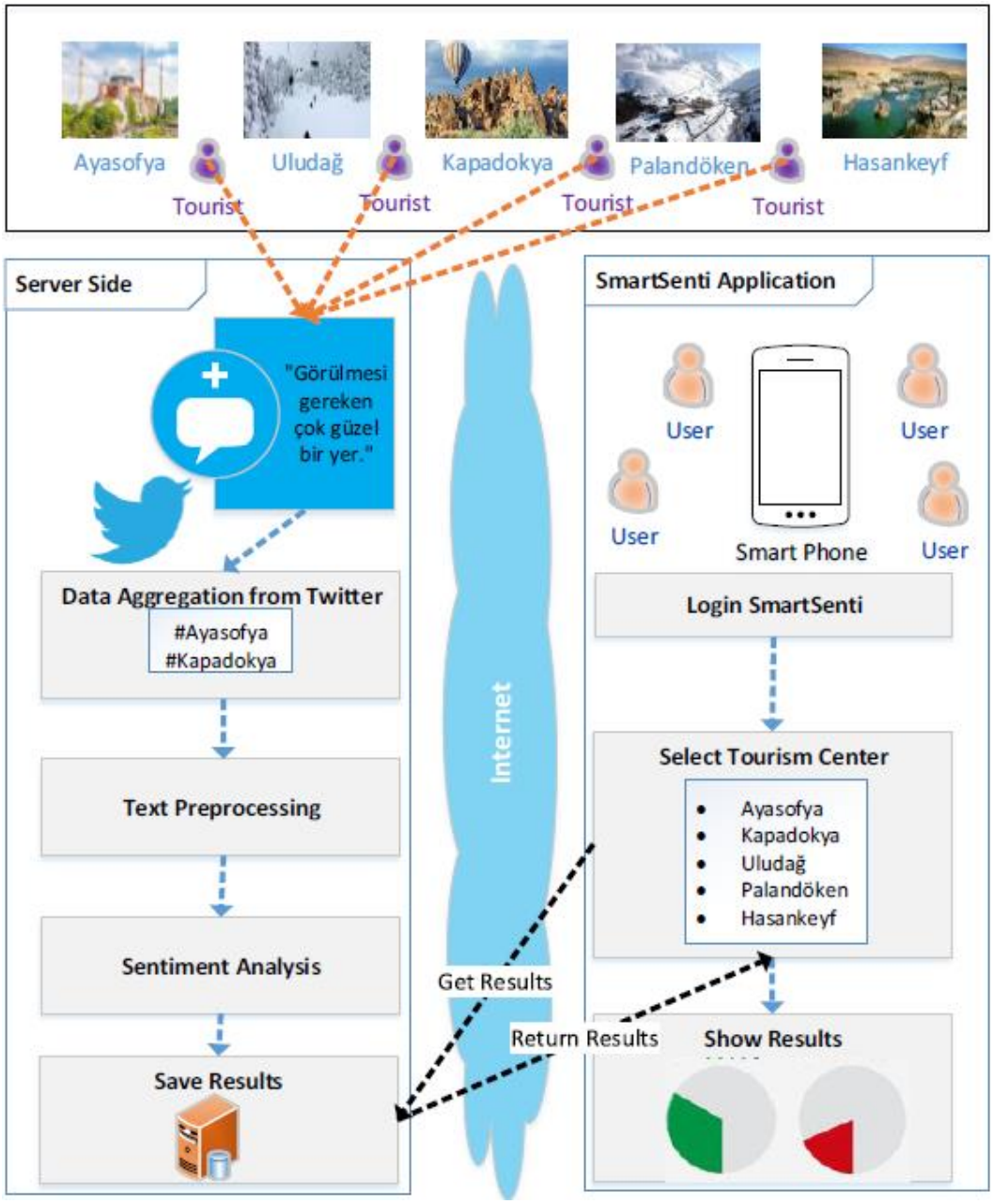
2.7 Análisis de sentimientos

Los turistas antes de hacer un viaje a un lugar específico, buscan en las redes sociales los comentarios que han sido realizados por visitantes que ya han visitado dicho lugar, esta actividad requiere de un gran esfuerzo y tiempo. Estos comentarios constituyen un mecanismo de interacción para transferir sus opiniones entre los diferentes usuarios. El análisis de sentimientos se trata de todo este proceso de revelar emociones en estos comentarios (Alaei et al., 2019). En este proceso es necesario subprocesar el contenido del texto de los comentarios además de extraer ciertas características, este subproceso de extracción busca eliminar el texto innecesario y dejar solo el contenido en forma normal por medio de tareas que se aplican usando algoritmos de Inteligencia Artificial (Etaiwi & Naymat, 2017).

Pensando en buscar una aplicación de análisis de sentimientos y resolver el problema de la búsqueda exhaustiva de comentarios, los autores Salur et al. (2019), han desarrollado una aplicación llamada SmartSenti para analizar los sentimientos de los comentarios que se realizaron en publicaciones de Twitter, sobre centros turísticos, utilizando métodos de aprendizaje automático y aprendizaje por transferencia. El trabajo consta de la recopilación de tweets sobre cinco lugares turísticos de Turquía. Estos tweets se clasificaron como positivos o negativos después de procesarlos y extraer sus características. Los resultados de esta clasificación se muestran en una aplicación móvil desarrollada por los autores. Los autores proponen una solución de software para hacer turismo de manera inteligente. La figura 2-5 muestra su estructura básica. El sistema se compone de dos partes básicas, las cuales son las operaciones de cómputo del lado del servidor, y las operaciones de cómputo del lado del dispositivo móvil. En el lado del servidor se desarrolla una aplicación en java que recopila publicaciones de Twitter, y realiza los cálculos y operaciones sobre los datos de texto para el análisis de sentimientos. Por otra parte, los autores desarrollaron una aplicación Android que publica los resultados de los centros turísticos basados en la evaluación de los sentimientos de las publicaciones de Twitter.

Para el análisis de sentimientos de los centros turísticos, los autores seleccionaron cinco centros turísticos de diferentes ubicaciones de Turquía, a saber, "Ayasofya", "Uludağ", "Kapadokya", "Palandöken" y "Hasankeyf". Los hashtags de Twitter están determinados para cada uno de estos centros. Además, estos hashtags se guardaron en una base de datos relacional relacionada con cada uno de los centros turísticos.

Figura 2-4: Arquitectura del sistema SmartSenti.



Fuente: (Salur et al. 2019)

2.8 Conclusiones del capítulo

En este capítulo, se han presentado en forma breve las ideas básicas de los diferentes enfoques y paradigmas que conforman la base del sistema propuesto en este trabajo a saber; computación sensible al contexto, sistemas de recomendación, computación ubicua, modelo del usuario y su personalización, visitas guiadas personalizadas, ontologías y análisis de sentimientos. En el siguiente capítulo se indaga sobre el estado del arte de dichos enfoques para conocer las tendencias de la computación ubicua y sensible al contexto, y determinar el punto de partida sobre el cual se basan el resto de capítulos, especialmente en el modelado y personalización del usuario, los lugares y la información del contexto.

3. Estado del arte

En esta tesis de maestría, se propone un modelo de sistema de recomendación para visitas guiadas fundamentado en la investigación de algunas de las características más importantes de los sistemas informáticos ubicuos que incluyen, la computación sensible al contexto, la computación ubicua, el modelado del usuario y su personalización. Para tal fin, se presenta la revisión de algunos trabajos de investigación relacionados con los temas que conforman la definición del modelo propuesto, en dicha revisión se identifican las fortalezas y limitaciones de cada uno de ellos desde una perspectiva conceptual.

3.1 Trabajos relacionados con la computación sensible al contexto

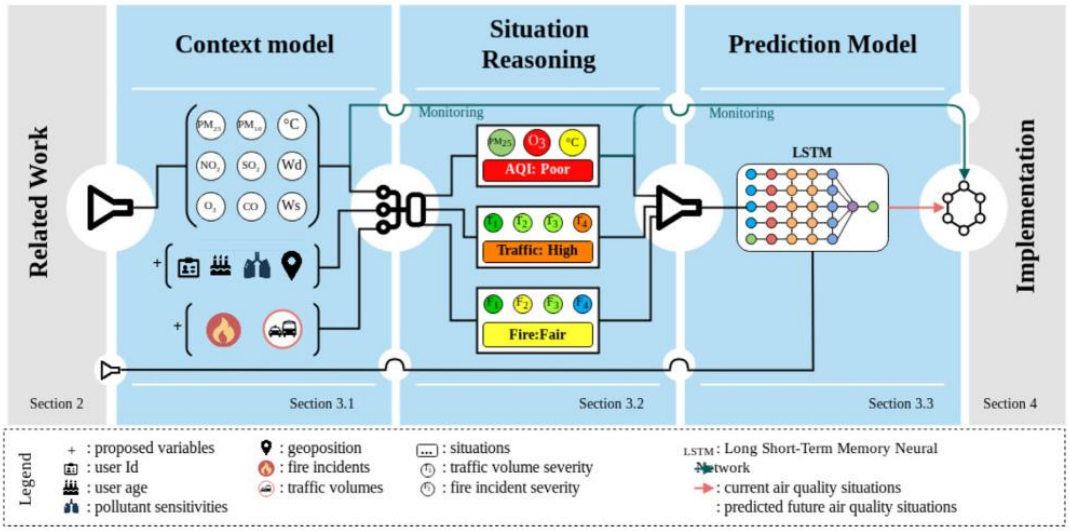
Los sistemas sensibles al contexto hacen uso de información contextual que es recopilada para brindar a los usuarios servicios de alto nivel en aplicaciones de diferentes campos. A continuación, veremos algunos trabajos propuestos sobre dicho tema.

3.1.1 Predicción de la calidad del aire sensible al contexto habilitada por inteligencia artificial en ciudades inteligentes

Un sistema novedoso de calidad del aire AQ (por sus siglas en Inglés) sensible al contexto, llamado mi índice de calidad del aire MyAQI (por sus siglas en Inglés) presentado en por Schürholz et al, (2020) busca superar las limitaciones encontradas por los autores en otros trabajos de AQ basados en el contexto, en los cuales el enfoque principal se orienta en

mejorar los algoritmos de aprendizaje de máquinas, y en afinar los parámetros de la calidad del aire que se usan en los modelos de predicción y aplicarlos en nuevas técnicas.

Figura 3-3-1: Bloques de construcción del sistema de predicción QA sensible al contexto (MyAQI)



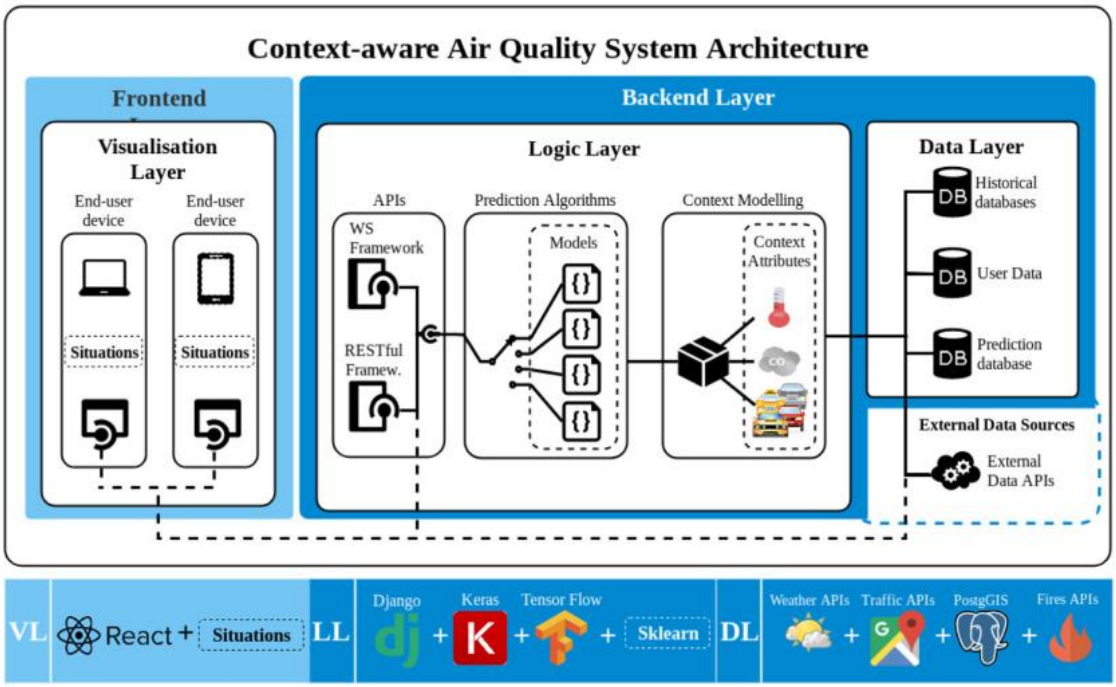
Fuente: (Schürholz et al., 2020).

El sistema propuesto es un sistema de predicción sensible al contexto que consiste de tres bloques de construcción, como se muestra en la figura 3-3-3. A saber: (i) Modelado del contexto, (ii) Razonamiento de la situación; y (iii) Modelo de predicción.

En el modelado del contexto se usa la teoría del espacio contextual CST (por sus siglas en inglés), que ha demostrado ser una herramienta robusta pero simple para formalizar los fenómenos de la vida real en un modelo computacional. Según esta teoría, se debe definir primero el espacio de aplicación, el cual comprende todas las variables y valores que integran el modelo. Los espacios del contexto son subconjuntos de este espacio, formalizados como espacios euclidianos N-dimensionales definidos sobre una colección de atributos de contexto, donde N representa el número de atributos seleccionados. Un atributo de contexto representa una característica específica, y es crítica para satisfacer las funciones del sistema. Cada atributo consiste de un Id, un tipo de valor y un rango de valores que se puede asignar dentro del espacio del contexto. Los estados del contexto son las colecciones de valores asignados a cada atributo de contexto durante el funcionamiento del sistema.

Finalmente, una situación representa un evento en el mundo real y pertenece a uno o más espacios de situación, que son un subconjunto de los estados del contexto. En CST, las situaciones pueden ser derivadas como el resultado de interceptar los estados del contexto con los espacios de situación. En el razonamiento de la situación se sigue el enfoque CST, se deben especificar los espacios de situación. Se especifican tres situaciones in el sistema MyAQI, y se describen de la siguiente forma.

Figura 3-3-2: Arquitectura del modelo de predicción del sistema MyAQI y el flujo del funcionamiento general de la predicción



Fuente: (Schürholz et al., 2020).

El resultado de la fase de razonamiento y las situaciones que son deducidas se utilizan como datos de entrada en el modelo de predicción. Un modelo de predicción que consume los datos del medio ambiente de AQ y los atributos de contexto extendidos para predecir situaciones requiere adoptar una técnica de predicción confiable. Por atributos de contexto extendidos se entienden los atributos que han sido transformados a través del modelo de razonamiento de la situación para los valores relevantes de cada situación. Los bloques de construcción que conforman el sistema de predicción sensible al contexto (el modelo

del contexto, el modelo de razonamiento de la situación y el modelo de predicción) fueron implementados en un programa de software, cuya arquitectura consta de tres capas, representadas en la figura 3-3-4.

Las capas de datos, lógica, visual, backend y frontend, solo se usan como capas de meta datos para el sistema MyAQI. La capa de datos trae la información necesaria para alimentar los atributos del modelo del contexto del api externo del sistema AQ, también los datos de los incidentes, los datos de la cantidad del tráfico, y los datos almacenados en las bases de datos relacionales, junto con la información ingresada por el usuario. La capa lógica está integrada de tres módulos: (i) el modulo para el modelado del contexto, que mapea los datos crudos en atributos de contexto que se puedan usar, (ii) el módulo de predicción, que usa un algoritmo que realiza el análisis de los datos, esta predicción realizada con los atributos del contexto, para aumentar la información conocida, y (iii) el sistema MyAQI.

Fortalezas:

- El enfoque propuesto también contribuye a aumentar la conciencia del personal profesional y público acerca de cómo los problemas de AQ afectan a las ciudades y a su población, permitiendo así que se tomen mejores decisiones para los sectores de la salud pública y la planificación urbana.
- Los resultados personalizados que produce el sistema MyAQI aumentan la conciencia del ciudadano individual sobre como la contaminación del aire podría afectarle, lo cual genera un efecto en cadena.

Limitaciones:

- Un aspecto importante a abordar, es la incertidumbre introducida por los datos reales debido a posibles perturbaciones o por el mal funcionamiento de las estaciones de medición, para esto se pueden explorar algoritmos que trabajen con datos producidos solo por dispositivos individuales que detecten la contaminación de aire y que sean distribuidos a cada usuario.
- Para todas las estaciones de medición, excepto una, la mejora en la predicción es muy clara cuando se usa la información del contexto ambiental. La falla en esta estación puede representar una falla en la correlación entre las variables del contexto extendido

y el AQ en el área, probablemente proveniente de otra fuente de contaminación. Las otras estaciones se vieron más influenciadas por los incendios forestales.

3.1.2 Reconocimiento de actividad sensible al contexto y adaptable dinámicamente con relojes inteligentes: un estudio de caso sobre el tabaquismo

En la investigación realizada por Agac et al. (2021), se aborda la siguiente pregunta de investigación "¿Cuál es el impacto de cambiar de manera dinámica los parámetros para el reconocimiento de la actividad en el desempeño tanto en términos de la tasa de reconocimiento como en el consumo de recursos?". Para este propósito, los autores proponen un algoritmo sensible al contexto, Conawact, el cual activa diferentes sensores, la tasa de censado de datos, y características que se piden en el momento de la demanda de acuerdo al contexto del usuario. El algoritmo es antecedido por otro algoritmo que se encarga de detectar el estado que busca si el usuario está realizando una actividad simple o más compleja. Si la actividad es simple, entonces solo el acelerómetro es solicitado en el momento del censado y la actividad es clasificada usando un número pequeño de características. Por otro lado, si la actividad es compleja, también se activa el giroscopio y luego estos dos sensores son solicitados a una tasa de transferencia de datos más alta y más características son extraídas para para clasificar la actividad.

El algoritmo para seleccionar los parámetros sigue un enfoque que considera un conjunto de parámetros más grande que en otros estudios: sensores, características, tasa de captura de datos y el tamaño de la ventana. Para entrenar el clasificador, los autores usan un conjunto de datos, previamente recogido por ellos mismos, que incluye once participantes que realizan 10 actividades diferentes. Los autores evalúan el desempeño del algoritmo en un escenario donde los usuarios realizan las actividades de manera secuencial. Se realiza una comparación del desempeño con dos casos, en el primer caso todos los parámetros son fijados previamente y en el segundo caso algunos de los parámetros se adaptan en forma dinámica.

El conjunto de datos se obtuvo usando una aplicación de un reloj inteligente utilizado por 11 participantes en un periodo de tres meses. La aplicación registra las lecturas de la marca del tiempo, el acelerómetro trío-axial y el giroscopio de un reloj inteligente y un teléfono inteligente. Todos los datos se muestrean a 50 Hz. Las actividades realizadas son fumar de pie (smokeSD), fumar sentado (smokeST), fumar mientras camina (smokeWalk), fumar en una conversación grupal (smokeGroup), beber mientras está de pie (drinkSD), beber mientras está sentado (dinkST), comer (comer), pararse (pararse), sentarse (sentarse), caminar (caminar).

En este conjunto de datos, se hace énfasis en el reconocimiento del tabaquismo en diferentes formas, así como en actividades similares que pueden confundirse con fumar. Todas las actividades son realizadas por todos los participantes, excepto smokeGroup y smokeWalk que son realizadas por 8 y 3 participantes en total, respectivamente. Durante la recopilación de los datos, cada participante utilizó un teléfono inteligente y un reloj inteligente. Los teléfonos inteligentes se colocaron en el bolsillo del pantalón derecho y los relojes inteligentes en la posición de la muñeca derecha. Se pidió a los participantes que realizaran cada una de las diez actividades al menos cinco veces, en el lugar que quisieran y cuando quisieran. Por lo tanto, la repetición de las actividades no fue continua. El único requisito era mover los brazos hacia arriba y hacia abajo antes de comenzar y después de completar la actividad para marcar el inicio y el final de la actividad.

Para detectar el estado se definieron dos tipos de actividades, que se describen como simples y complejas. Las actividades simples son en su mayoría periódicas, mientras que las complejas no lo son, lo que dificulta su reconocimiento solo con un acelerómetro. El estado de la actividad (simple o complejo) se puede utilizar como disparador para activar o desactivar los parámetros relacionados, como el sensor. Los autores, identificaron sentarse, pararse, y caminar como actividades simples, mientras que las demás actividades se clasificaron como complejas.

Debido a la simplicidad de funcionamiento en un reloj inteligente, utilizamos el método basado en reglas para la detección de los estados. Se usa el algoritmo de selección de parámetros dinámicamente adaptable de Conawact para el reconocimiento de cada actividad, este algoritmo contiene tres módulos; que son el módulo de detección de estado, el módulo de sesión y el módulo de clasificación.

Fortalezas:

- Se propone el algoritmo de Conawact, que selecciona los parámetros para el reconocimiento de la actividad de manera dinámica y según el tipo de actividad que se esté realizando. Este algoritmo supera el enfoque donde se usan parámetros estáticos, la tasa de reconocimiento es 6% más que la tasa del otro enfoque y un 22% menos de consumo de energía.
- El algoritmo Conawact analiza el consumo de recursos en términos de energía, memoria y CPU, lo que no se analiza al mismo tiempo en otros estudios.

Limitaciones:

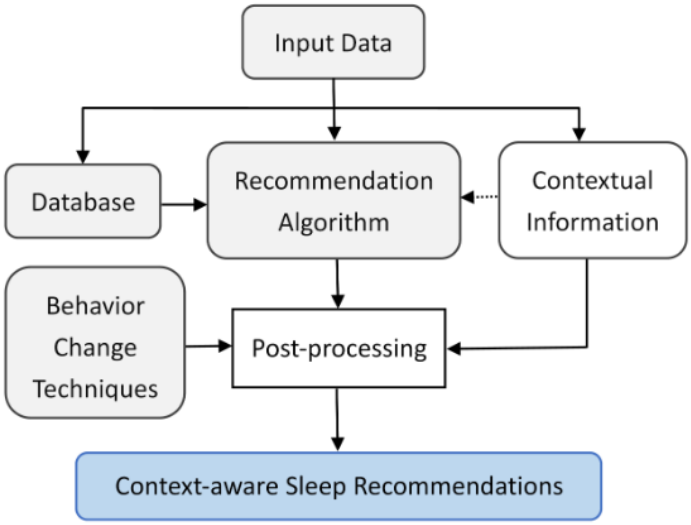
- Se cuentan con diez tipos de actividades para trabajar con el algoritmo sensible al contexto, en la vida diaria realizamos muchas más actividades que tienen una configuración y variación más genérica, si estos nuevos datos se agregaran al conjunto de actividades inicial ya definido se podría mejorar el rendimiento del reconocimiento del algoritmo. En otro estudio los autores utilizaron una recolección de datos que incluían actividades como lavar platos, trabajar en la computadora, lo cual tuvo un efecto positivo en el aumento de las predicciones de las actividades de fumar y no fumar.
- Aunque en la literatura se ha propuesto la idea de utilizar un método de clasificación jerárquico, en este trabajo los autores siguieron un enfoque holístico considerando un conjunto más amplio de parámetros. Los autores argumentan que probaron el rendimiento con un conjunto específico de actividades utilizando un conjunto de datos específico. Sin embargo, el algoritmo no está diseñado solo para estas actividades y se pueden agregar nuevos parámetros.

3.1.3 Sistemas de recomendación consientes del contexto para la salud del sueño: en forma de revisión narrativa

En el trabajo de Liang et al, (2022) se define el término CASHRS (por sus siglas en inglés) que significa, sistema de recomendación consiente del contexto para la salud del sueño, y que se ve como un campo disciplinario de investigación en el que confluyen varios

dominios de investigación, que incluyen los sistemas de recomendación para la salud, la computación ubicua y móvil, la computación sensible al contexto, tecnologías de persuasión, interacción humano computador, consumidores electrónicos, e informática para la salud.

Figura 3-3-3: Arquitectura de los sistemas CASHRS.



Fuente: (Liang et al., 2022)

Los autores definen una arquitectura que se ilustra en la figura 3-3-5, que típicamente consiste de 4 componentes principales: el ingreso de datos, una base de datos, el algoritmo de recomendación, y las técnicas aplicadas al cambio de comportamiento.

El ingreso de datos se puede obtener de manera explícita o implícita para inicializar el proceso de recomendación. La base de datos guarda la información de los perfiles del usuario y de los diferentes objetos (ej., consejos de higiene para el sueño). El algoritmo de programación utiliza los datos ingresados y la base de datos para sugerir diferentes formas de intervención en forma de comportamientos que apuntan al usuario. Los sistemas CASHRS se adaptan al contexto del usuario, ya que consideran su contexto como cualquier factor que puede influir en la cualidad y continuidad del sueño durante la noche.

Fortalezas:

- La literatura sobre los sistemas CASHRS sigue siendo muy poca y carece de un marco sistemático de estudio. El sueño no se ha estudiado en la misma medida que otros

temas relacionados con la salud, como el ejercicio y la dieta, en la investigación CARS, probablemente debido a la preocupación de que las tecnologías digitales puedan no ser adecuadas para el sueño porque pueden interferir con el sueño en sí. Sin embargo, la mayoría de los estudios revisados en este trabajo establecieron evidencia de la eficacia de CASHRS para mejorar el sueño.

- Los enfoques de los sistemas CASHRS también comparten características únicas que no se observan en otros CARS de salud porque el sueño comprende construcciones multidimensionales que a menudo no son directamente controlables.

Limitaciones:

- Los sistemas CASHRS necesitan cerrar la brecha entre la recomendación y la acción para facilitar el inicio de un cambio de comportamiento y fomentar el cumplimiento sostenido de las recomendaciones.
- Los sistemas CASHRS investigados, solo consideran la información del contexto que realmente se utilizó en el algoritmo de recomendación, pero la información que fue capturada por el sistema y que no se usó en el algoritmo de recomendación fue descartada. Esta información constituye factores del contexto que no se pueden ignorar ya que pueden influir en el resultado al variar en diferentes frecuencias.

3.2 Trabajos relacionados con la computación ubicua

La computación ubicua brinda tecnología que se entrelaza discretamente en la vida cotidiana y el ambiente circundante (Alkhafaji et al., 2020). A continuación, veremos algunos trabajos propuestos sobre dicho tema.

3.2.1 Un esquema teórico para diseñar entornos de aprendizaje inteligente y ubicuo para el patrimonio cultural al aire libre

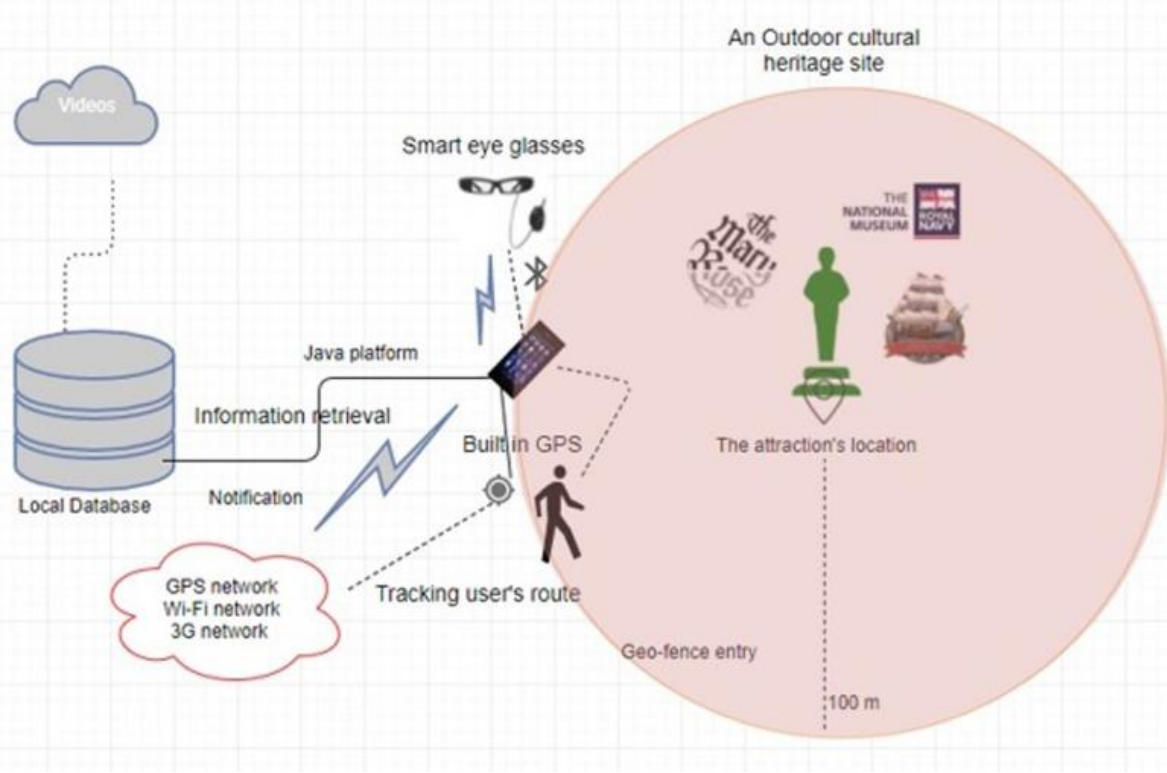
Un esquema teórico para diseñar entornos de aprendizaje inteligentes y ubicuos o FoSLE (por sus siglas en inglés) para el patrimonio cultural al aire libre, que considera los

requisitos de los usuarios (Alkhafaji et al., 2020). El esquema aprovecha las tecnologías móviles y ubicuas (es decir, gafas inteligentes y realidad aumentada) con el objetivo de mejorar la participación de los visitantes. La metodología utiliza un enfoque de diseño centrado en el usuario, y una versión adaptada de la metodología de ingeniería socio cognitiva o SCE (por sus siglas en Inglés) para ayudar a responder las preguntas de investigación, ¿Cómo utilizan las personas las tecnologías ubicuas con fines de aprendizaje en los sitios del patrimonio cultural?, ¿Cuáles son los elementos esenciales para desarrollar un entorno de aprendizaje ubicuo que utilice tecnologías móviles y portátiles para sitios del patrimonio cultural y que satisfaga las necesidades del usuario?. SCE consta de las etapas de análisis y diseño, que están conectadas en una etapa intermedia, el modelo de tareas. El enfoque de este trabajo está en la etapa de análisis, que condujo al desarrollo del esquema FoSLE.

Se realizaron de manera secuencial tres estudios empíricos de campo utilizando métodos mixtos con datos recopilados mediante grupos focales, cuestionarios, y técnicas de entrevista. Este enfoque involucra a los usuarios finales en el proceso de diseño, lo que tiene el beneficio de diseñar una herramienta en función de sus perspectivas para beneficiar un aprendizaje adaptativo. Basándose en los resultados de los estudios, se formuló el esquema teórico, FoSLE, para diseñar entornos de aprendizaje inteligente y ubicuo o S-ULE (por sus siglas en inglés) y para aprender de manera informal en contextos de patrimonio cultural.

El esquema consta de seis categorías, que son: usuario, contenido, diseño de aprendizaje, diseño de interacción, contexto, desafíos y obstáculos. Se realizó un análisis para extraer un conjunto de requisitos generales que permitieran diseñar los servicios. Sobre la base de un subconjunto de estos requisitos, se diseñó un prototipo de aplicación móvil de un servicio sensible al contexto para entornos de patrimonio cultural al aire libre, llamado SmartC. Esta aplicación nativa de Android utiliza servicios basados en la ubicación (LBS) para identificar la ubicación de los usuarios y permitir que el dispositivo proporcione información instantánea en movimiento sobre los sitios cercanos del patrimonio cultural.

Figura 3-3-4: Arquitectura de SmartC.



Fuente: (Alkhafaji et al., 2020).

Se utilizó tecnología de Geo-fence, la cual permite colocar un límite alrededor de un área geográfica, funcionando cuando un usuario entra o sale del área, que se identifica por la latitud y longitud del área. El dispositivo móvil se activa cuando un usuario ingresa a dicha zona virtual, que se rastrea mediante el sistema de posición global o GPS (por sus siglas en inglés) del dispositivo. El dispositivo envía una notificación para alertar al usuario cuando se acerca a una atracción. Las notificaciones se envían simultáneamente por medio de la aplicación ya sea desde la interfaz basada en dispositivos móviles o la interfaz basada en gafas cuando se activa el dispositivo móvil, ver la figura 3-3-6.

Fortalezas:

- Apoyar el aprendizaje en movimiento se considera un tipo de aprendizaje importante en el momento actual, lo que ayuda a los alumnos a mantenerse al día con el rápido ritmo de la vida.

- Los anteojos inteligentes mejorarían la interacción en términos de liberar las manos de los visitantes y les darían la oportunidad de recibir información sobre las atracciones y, al mismo tiempo, asociarían su visión con los artefactos físicos.

Limitaciones:

- Aunque los participantes eran de diferentes nacionalidades y antecedentes, la mayoría de ellos eran residentes en el Reino Unido, lo que puede haber introducido un sesgo cultural.
- Es posible que las categorías no cubran absolutamente todos los aspectos del diseño, ya que algunos no se incluyeron ampliamente en la investigación, como los aspectos técnicos del diseño, que solo se mencionaron como desafíos.

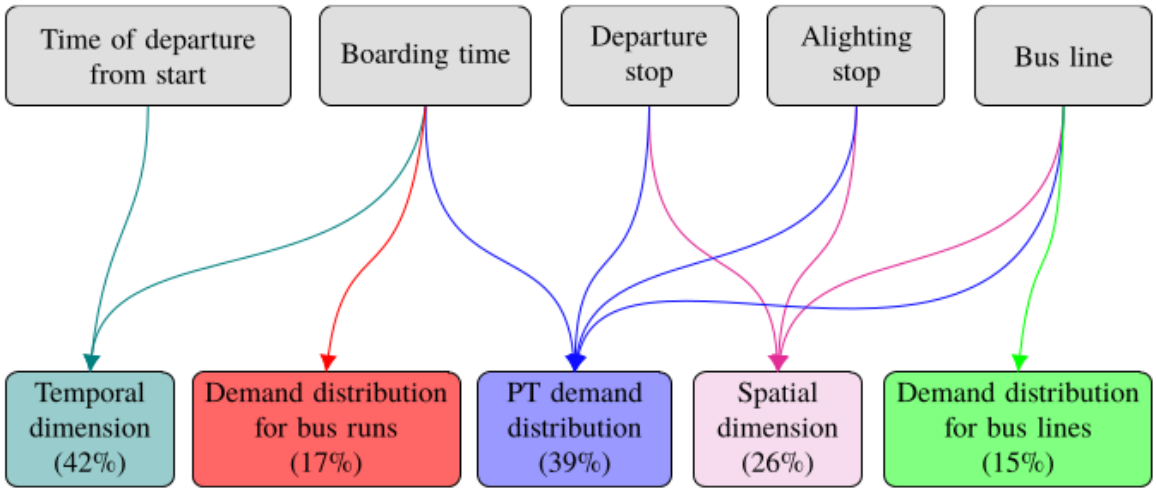
3.2.2 Opciones de ruta de los pasajeros de autobús después de consultar información ubicua en tiempo real

La información ubicua de pasajeros en tiempo real URTPI (por sus siglas en inglés) permite a los usuarios del transporte público PT (por sus siglas en inglés) tomar mejores decisiones de viaje tanto en las etapas previas al viaje como en las etapas de ruta (Islam & Fonzone, 2021). Este estudio investiga el impacto de URTPI en la elección de la ruta de los pasajeros en autobús. Con este fin, se realizó una encuesta a los pasajeros de autobuses en la ciudad de Edimburgo, Reino Unido, y se recopiló un total de 1645 respuestas completadas. Más de la mitad de los participantes de la encuesta utilizaron al menos una fuente de información ubicua. Los resultados de la encuesta revelan que alrededor del 55% de los usuarios de URTPI cambiaron al menos un aspecto de su viaje. Cambiar la hora de salida con respecto a la hora de inicio y la hora de embarque son las dos acciones más populares tomadas por los pasajeros de autobuses después de consultar a URTPI. Las decisiones de los pasajeros están influenciadas por la información sobre la hora de llegada del autobús, la ruta del autobús y la distancia a pie. El estudio demuestra el impacto potencial del cambio en las opciones de los pasajeros en la distribución de la demanda de TP. Los autores descubrieron que la distribución de la demanda de recorridos de autobuses podría modificarse potencialmente en un 17% y para las líneas de autobuses en un 15%. La distribución general de la demanda de la red podría verse afectada en el 42% de los casos

como resultado de la consulta a URTPI. Este estudio aboga por que los planificadores y operadores de transporte deben tener en cuenta el impacto potencial de URTPI para hacer mejores predicciones de la distribución de la demanda de TP.

Se investigaron los siguientes elementos de elección de ruta para comprender cómo se influye en la elección de ruta de los pasajeros después de consultar a URTPI. Hora de salida desde el inicio, hora de embarque, parada de autobús de salida, parada de autobús de bajada, ruta/línea de autobús.

Figura 3-3-5: Impacto potencial de los cambios en los elementos de elección en la distribución de la demanda de TP



Fuente: (Islam & Fonzone, 2021).

La figura 3-3-7 presenta cómo los cambios en los diferentes elementos de elección afectan la distribución de la demanda de TP. Aproximadamente el 42% de los usuarios de URTPI realizan cambios en su dimensión de elección temporal, mientras que los cambios en la dimensión de elección espacial se observan en el 26% de los casos. El impacto de los cambios en la distribución de la demanda de TP se estima combinando los cambios que se producen en todos los elementos de elección, excepto en el momento de la salida del inicio. En consecuencia, para el 39% de los viajes que realizan los usuarios de URTPI, la distribución de la demanda de PT se ve afectada por la consulta de información. El cambio de hora de embarque se observa en el 21% de los casos; sin embargo, los cambios en la distribución de pasajeros para los recorridos de autobús ocurren solo si los pasajeros

cambian su hora de abordaje sin cambiar la línea de autobús. Se observa que la distribución de la demanda de recorridos de bus se ve afectada por el 17% de los viajes realizados por los usuarios de URTPI.

Fortalezas:

- El manejo de la información para el transporte público se ha vuelto ubicua.
- La información ubicua de pasajeros en tiempo real (URTPI) permite el acceso a la información por parte de los pasajeros de transporte público desde cualquier lugar y en cualquier momento.

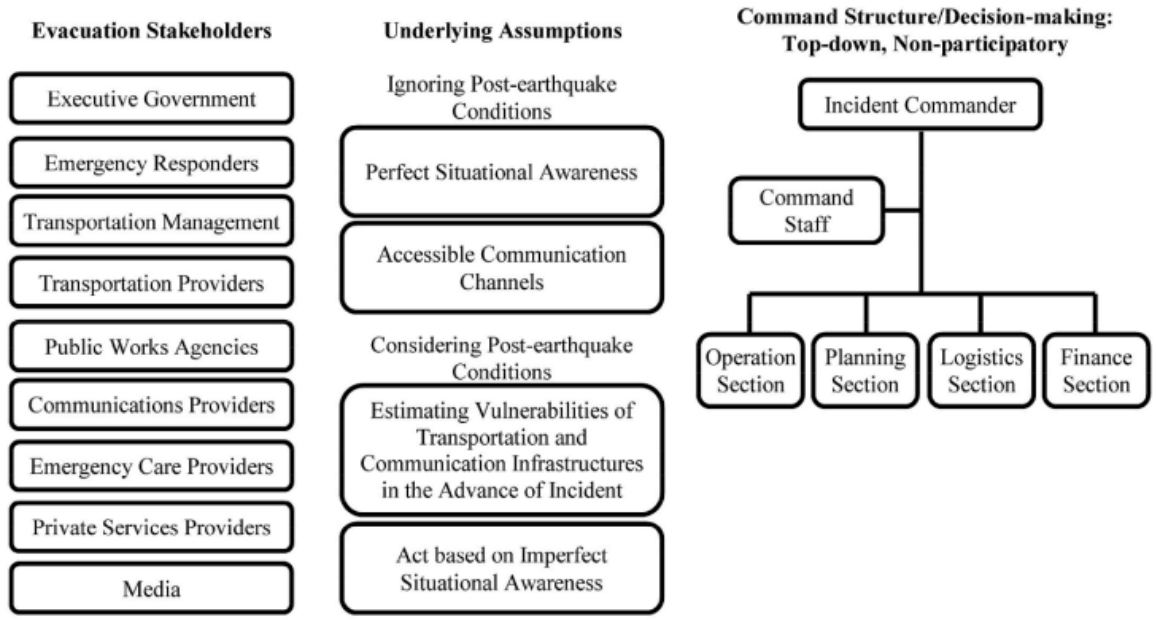
Limitaciones:

- El impacto de URTPI se estudió considerando la disponibilidad de alternativas, como autobuses frecuentes y líneas alternativas. Aunque URTPI puede ser más importante para los pasajeros cuando no hay alternativas disponibles, esto no se capturó en la encuesta de paradas de autobús.
- En este trabajo no se considera el usuario desde un enfoque de estudio como elemento que se puede modelar y tomar decisiones que permitan realizar acciones personalizadas.

3.2.3 Asistencia ubicua para la evacuación al aire libre basada en GIS: una solución eficaz respuesta a desastres sísmicos

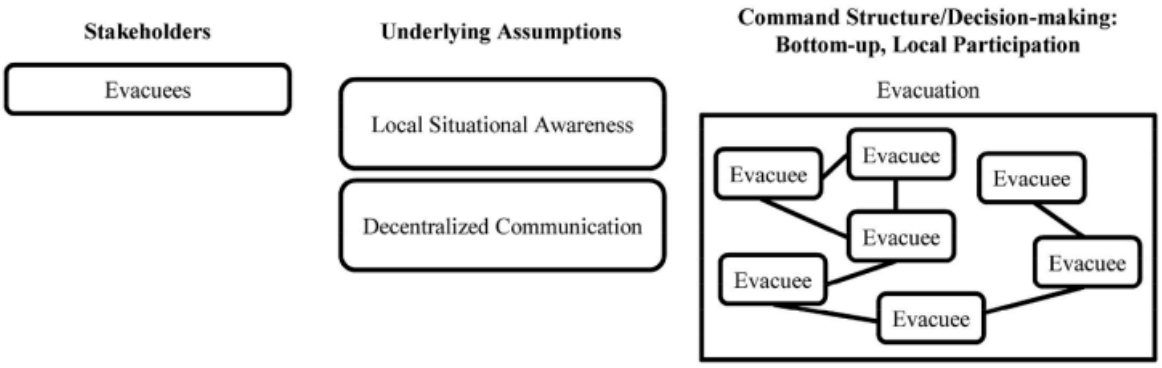
En Ghafoori et al. (2022), presentan el sistema UbiEvac (por sus siglas en inglés), llamado Ubiquitous-GIS-based outdoor evacuation assistance. Este sistema busca acompañar a los evacuados de terremotos identificando la ruta más corta y rápida entre los puntos de origen y destino. Para evaluar el desempeño de UbiEvac se compara con otros sistemas centralizados en las autoridades y descentralizados e independientes de evacuación. La solución propuesta ofrece ubicuidad, participación pública y la toma de decisiones que integra a las autoridades y a los ciudadanos para generar rutas de evacuación.

Figura 3-3-6: Enfoque centralizado de gestión de la evacuación en caso de terremoto.



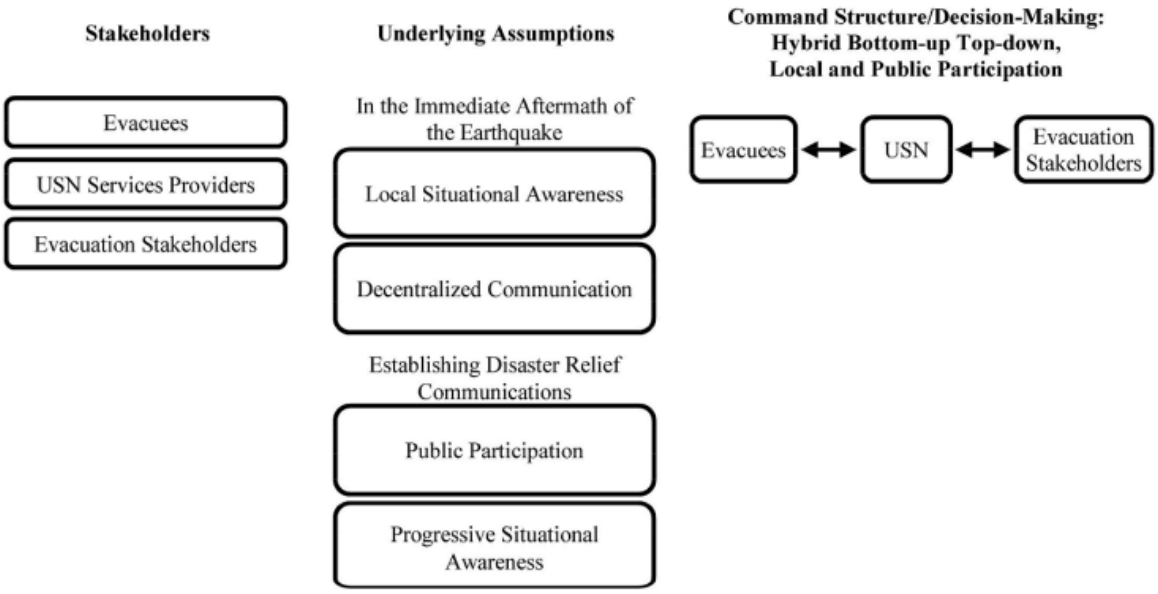
Fuente: (Ghafoori et al., 2022).

Figura 3-3-7: Enfoque descentralizado de gestión de la evacuación en caso de terremoto.



Fuente: (Ghafoori et al., 2022).

Figura 3-3-8: Enfoque de UbiEvac para la gestión de la evacuación en caso de terremoto.



Fuente: (Ghafoori et al., 2022).

La metodología para implementar UbiEvac considera los enfoques centralizados y descentralizados para la gestión de terremotos como se muestran en las siguientes (figuras: 3-3-10, 3-3-11, y 3-3-12). En términos de los interesados, suposiciones, estructuras de mando, y toma de decisiones, además de retratar la idea general de como UbiEvac se relaciona con estos enfoques.

Para entender UbiEvac, considere el escenario después de un terremoto con un servicio de asistencia a la evacuación llamado (Servicio UbiEvac) que ayuda a los evacuados que están caminando o en vehículo, proporcionándoles las rutas de evacuación más cortas y rápidas entre los sitos de origen y destino. UbiEvac debe basarse en un sistema Ubicuo-GIS, lo que implica que:

Es accesible en todas partes, es decir, se ejecuta en cualquier momento y en cualquier lugar donde GSN y/o teléfonos inteligentes estén presentes.

Es compatible con cualquier usuario: proporciona rutas de evacuación para evacuados equipados y no equipados a través de smartphones y actuadores de GSN (VMS5 respectivamente).

Es robusto a las consecuencias de los terremotos y funciona bajo cualquier condición: se adapta principalmente a fuentes locales descentralizadas (GSN) disponibles y, en segundo lugar, a fuentes global-centralizadas (VGI) a través de comunicaciones de socorro en casos de desastre basadas en LTE.

Fortalezas:

- El sistema UbiEvac involucra la participación de las autoridades y el público para la toma de decisiones de evacuación. Estas decisiones están integradas con el uso de una red de sensores ubicua y un sistema de información geográfica ubicuo para mejorar las deficiencias de sensibilidad situacional de manera más rápida.
- En este trabajo se investiga el impacto de la integración entre la estandarización y fusión de datos necesarios para la integración de múltiples fuentes de información.

Limitaciones:

- La asistencia a las evacuaciones al interior de edificaciones y otras infraestructuras, constituyen escenarios totalmente diferentes a las evacuaciones al exterior. Esta condición constituye un tema de investigación diferente, por lo que las evacuaciones al interior están fuera del alcance de este trabajo y por lo tanto no son consideradas.
- Dado que el enfoque propuesto de UbiEvac es simulado, esta investigación no pretendía una implementación con todos los detalles de estandarización, control de calidad y fusión de datos y, por lo tanto, estos detalles se simplificaron.

3.3 Trabajos relacionados con los sistemas de recomendación

Los sistemas de recomendación aplicados al turismo intentan ofrecer sugerencias personalizadas a los usuarios. Su objetivo es extraer las preferencias de los usuarios y presentar recomendaciones según sus preferencias. A continuación, veremos algunos trabajos propuestos sobre dicho tema.

3.3.1 Un novedoso sistema de recomendación turística en el contexto del comercio social

En el trabajo realizado por Esmaeili et al. (2020), se propone un modelo de recomendación social-híbrido para recomendar atracciones turísticas que utiliza la información sobre las comunicaciones individuales y las opiniones de otros viajeros para proporcionar una lista de recomendaciones personalizada para cada usuario activo basada en la confianza, la reputación y la similitud. Un usuario activo es un usuario al que el sistema de recomendación sugiere una lista de destinos.

En la arquitectura del sistema de recomendación se desarrollan cinco componentes para analizar los datos en la plataforma de comercio social turística. Cada componente funciona de la siguiente manera. El componente para determinar del tipo de usuario activo asigna el usuario a un grupo en función de la información implícita y explícita sobre dicho usuario y las atracciones turísticas. El componente para identificar usuarios similares extrae un subconjunto de usuarios en función de la similitud de su comunidad o características demográficas con el usuario activo. El componente de detección de la calificación de valores atípicos detecta las calificaciones que presentan los usuarios con poco conocimiento o a los usuarios falsos para asignarlos a una atracción. El componente de extracción de la lista de elementos calcula una tasa de cada atracción para el usuario activo en función de su grupo. El componente de provisión de la lista de recomendaciones crea una lista de destinos adaptada al usuario activo en función de sus calificaciones.

Fortalezas:

- En esta investigación se ven capacidades de modelado del usuario y su personalización. Al tratarse de un sistema de comercio social, que no solo incluye la información sobre los artículos, los usuarios y sus transacciones, sino que también contienen las relaciones sociales de los usuarios y sus opiniones.
- En esta investigación, asumiendo el impacto de la relación social en la toma de decisiones al viajar, los autores proponen un sistema de recomendación híbrido social basado en la confianza, reputación, similitud y comunidades sociales.

Limitaciones:

- Si las empresas pueden alentar a los usuarios a contribuir a la presentación de sus puntos de vista, la restricción de compartir su información por motivos de privacidad se eliminará y se podrán utilizar datos más completos para el sistema de recomendación.
- En esta investigación no se fue posible obtener retroalimentación de los usuarios. Y al considerar las características del modelo, bajo esta situación, los resultados de la evaluación serían diferentes comparados con otros enfoques.

3.3.2 Navegando a través del complejo sistema de transporte: un enfoque heurístico para la recomendación de turismo urbano

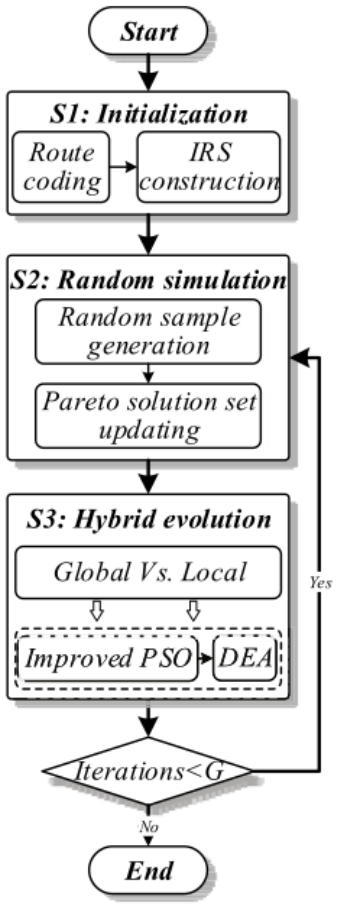
En este trabajo los autores intentan abordar el problema asociado con la elección de los modos de transporte en el turismo urbano. Los autores Zheng et al , (2020) proponen un modelo de itinerarios personalizado que tiene en cuenta la elección del modo de transporte. Se realizó un estudio de casos en Chengdu, China, para examinar el desempeño del modelo. Los resultados mostraron que, en comparación con los métodos existentes, este enfoque obtuvo un mejor rendimiento. Además, de crear itinerarios más sensibles, variados y personalizados.

Uno de los elementos críticos en un sistema de recomendación turística es el problema de diseño de viajes turísticos TTDP (por sus siglas en inglés) que planifica rutas para maximizar la participación turística considerando varias limitaciones. A diferencia de los estudios generales sobre TTDP, el diseño de rutas turísticas para turistas urbanos requiere la integración de varios modos de transporte, a nivel de funcionalidad. Los distintos modos de transporte implican diferentes tiempos de viaje que a su vez involucran un nivel de incertidumbre o riesgo y los turistas tienen diferentes actitudes hacia el riesgo. Sin olvidar que la congestión del tráfico tiene diferentes impactos en el tiempo de viaje según el modo de transporte que se elija.

Por lo tanto, en este estudio se abordaron estos desafíos claves proponiendo un modelo de itinerarios diarios personalizados para los turistas de la ciudad, considerando la elección del modo de transporte TTDP-TMC (por sus siglas en inglés). En particular, se

consideraron un par de objetivos en conflicto (utilidad y riesgo) y muchas limitaciones asociadas con atracciones, turistas y redes de transporte urbano. Los autores optaron por la adopción de la definición de optimización de Pareto y se diseñar un enfoque heurístico de clasificación no dominado NSHA (por sus siglas en inglés) con optimización de enjambre de partículas mejorada PSO (por sus siglas en inglés) y un algoritmo de evolución diferencial DEA (por sus siglas en inglés).

Figura 3-3-9: Esquema metodológico



Fuente: (Zheng et al., 2020)

El esquema general del NSHA se ilustra en la Figura 3-3-13, y se compone de la inicialización, la simulación aleatoria y la evolución híbrida. La inicialización implica la codificación de las rutas con un cromosoma asimétrico de longitud variable y triple capa, y se construye un conjunto de rutas inicial IRS (por sus siglas en inglés) utilizando un algoritmo codicioso mejorado. En una simulación aleatoria, se generan muestras aleatorias para determinar las soluciones factibles, a partir de las cuales se produce un conjunto de

soluciones de Pareto. Finalmente, las soluciones factibles siguen una estrategia de evolución híbrida que fusiona la PSO mejorada y la DEA.

Fortalezas:

- Se evidencia el uso de la personalización ya que un viaje completo se divide en etapas y estas a su vez se dividen en un número de vértices por destino (paradas). En cada etapa, la utilidad ganada depende del vértice visitado. Específicamente, la utilidad depende de la preferencia del usuario, del vértice visitado y del tiempo real pasado en dicho vértice.
- Se evidencia el uso del modelado del usuario ya que en el modelo se puede considerar efectivamente el perfil de riesgo del usuario y, por lo tanto, se emplea para minimizar la probabilidad de que no se complete el recorrido dentro del tiempo asignado.

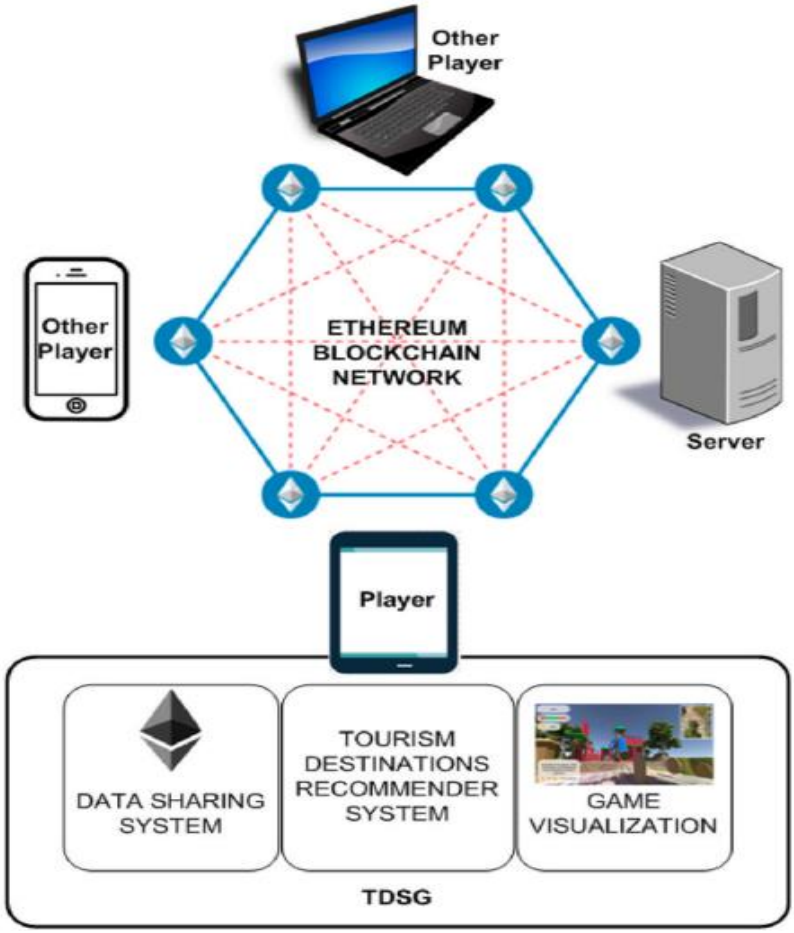
Limitaciones:

- En este estudio no se evidencia el uso de la computación sensible al contexto.
- En este estudio no se evidencia el uso de la computación ubicua ya que los datos turísticos se recopilaron por medio de encuestas a turistas realizadas cerca de las salidas de las estaciones de tren o del aeropuerto.

3.3.3 Sistema de recomendación descentralizado para inteligencia ambiental de juegos serios de destinos turísticos utilizando un enfoque de calificación conocido y desconocido

En el trabajo de Arif et al. (2023), se utiliza un sistema de recomendación descentralizado en un juego serio para transferir conocimientos sobre las opciones de destinos turísticos recomendados a los jugadores, llamado Juego Serio de Destinos Turísticos, TDSG (por sus siglas en inglés). También ofrece inteligencia ambiental integrada para aumentar la comodidad, y la conectividad de los jugadores. El sistema utiliza el enfoque de calificación conocida y desconocida para mejorar la precisión de las recomendaciones. Además, el estudio propone el uso de la tecnología blockchain para el intercambio de datos descentralizado para respaldar el sistema de recomendación.

Figura 3-3-10: TDSG en la red blockchain ethereum.



Fuente: (Arif et al., 2023)

En un sistema TDSG, cada jugador puede usar el juego y obtener recomendaciones basadas en preferencias y datos de calificación de otros jugadores. Por lo tanto, cada jugador que ejecute un dispositivo con TDSG debe conectarse con otros dispositivos a través de la red blockchain de Ethereum. La figura 3-3-16 muestra la relación entre los dispositivos en el sistema TDSG con sus tres partes principales; el sistema TDRS, el sistema de transferencia de datos y el componente de visualización del juego. Todo jugador conectado a la red debe tener instalado un dispositivo con el sistema TDSG. Al mismo tiempo, el servidor guarda una copia de seguridad de las preferencias de los datos y valoraciones de los jugadores.

Este estudio ofrece un sistema TDRS para respaldar el TDSG de inteligencia ambiental, que puede generar recomendaciones basadas en datos de criterios de calificación para

destinos turísticos y datos de preferencias de los jugadores. En otras palabras, cada TDRS puede generar recomendaciones de dos maneras. Las recomendaciones se generan en función de la similitud de las calificaciones de los jugadores con otros conjuntos de datos de clasificación de jugadores, y la forma se basa en la idoneidad de la preferencia de un jugador frente a los resultados de clasificación del conjunto de datos de preferencia de otro jugador.

Fortalezas:

- El uso de la tecnología blockchain para el intercambio de datos descentralizado garantiza una mejor seguridad de los datos y comodidad del usuario.
- El estudio presenta un sistema de recomendación de criterios múltiples que puede generar recomendaciones basadas en calificaciones de destinos turísticos o preferencias de los jugadores, garantizando la disponibilidad tanto para los turistas nuevos como para los que regresan.

Limitaciones:

- La disponibilidad de datos garantiza que los datos estarán disponibles previa solicitud, lo que puede limitar la accesibilidad de los conjuntos de datos de la investigación.
- La implementación de la tecnología blockchain y un sistema de recomendación de criterios múltiples puede requerir un alto nivel de experiencia técnica y recursos, lo que podría limitar la adopción del sistema propuesto en aplicaciones prácticas.

3.4 Trabajos relacionados con el modelado de visitas guiadas contextualizadas

Los sistemas orientados a turismo tienen el objetivo de mejorar la experiencia del usuario y la calidad del servicio mientras se ayuda a los usuarios a encontrar sitios y atracciones relevantes. A continuación, veremos algunos trabajos propuestos sobre dicho tema.

3.4.1 Sistemas de recomendación basados en filtrado colaborativo para Negocios, Atracciones y Destinos Turísticos

Esta investigación presenta diferentes técnicas en el campo del turismo, que incluyen sensibilidad a los sentimientos, modelado múltiple de tópicos, sistemas de recomendación con mejora de características visuales (Aldayel et al., 2023). En este estudio los autores evaluaron el desempeño de varios algoritmos de filtrado colaborativo para predecir la calificación de hoteles y la recomendación de atracciones turísticas. Entre los algoritmos evaluados se consideraron BiVAE por sus siglas en inglés (bilateral variational autoencoder), LightGCN por sus siglas en inglés (light graph convolutional neural network), SAR por sus siglas en inglés (simple algorithm for recommendation), SVD por sus siglas en inglés (singular value decomposition) y NMF por sus siglas en inglés (Non-negative matrix factorization). El estudio también consideró la configuración de los varios hiperparámetros para cada algoritmo y las métricas de evaluación utilizadas, como la precisión, el recall, y la precisión promedio media (MAP). Los resultados mostraron la eficacia de las diferentes técnicas de CF a la hora de predecir valoraciones y recomendar atracciones turísticas.

La arquitectura del sistema de recomendación propuesto por los autores implica la clasificación de los RS en diferentes categorías, como filtrado colaborativo (CF), filtrado basado en contenido (CB), filtrado híbrido, RS demográfico, RS basado en conocimiento, recomendador consciente del riesgo. RS, RS de red social y RS sensible al contexto. El sistema emplea CF y CB como enfoques comunes para recomendar elementos a los usuarios en función de su comportamiento de usuario anterior y el comportamiento de usuarios similares, además de analizar las características de diferentes actividades para hacer recomendaciones.

Fortalezas:

- Los algoritmos, como BiVAE, LightGCN, SAR, SVD y NMF, son capaces de proporcionar recomendaciones personalizadas a los turistas en función de sus preferencias y comportamiento.
- El algoritmo BiVAE, en particular, superó a otros modelos en términos de clasificación y métricas de predicción, lo que indica su eficacia a la hora de

proporcionar recomendaciones precisas.

Limitaciones:

- La investigación no menciona explícitamente ninguna desventaja o limitación de los algoritmos y sistemas de recomendación basados en filtrado colaborativo discutidos. El objetivo del artículo era evaluar el rendimiento de varios algoritmos de recomendación y proporcionar información sobre el proceso de recomendación empleado por los agentes de viajes, así como implicaciones prácticas para los especialistas en marketing turístico.

3.4.2 Sistema de recomendación para turismo basado en agrupamiento semántico y análisis de sentimientos

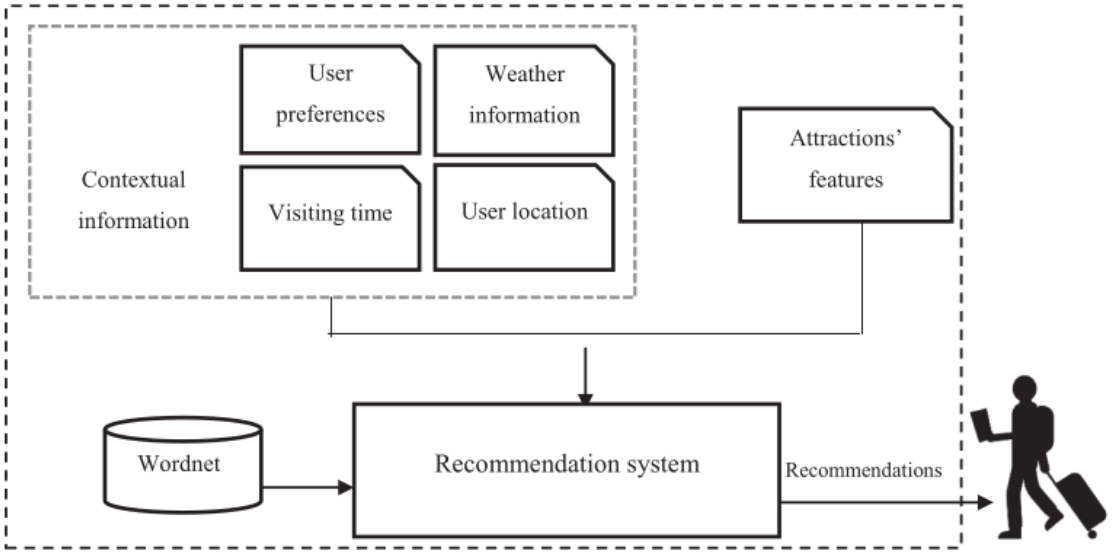
En este trabajo los autores presentan un sistema de recomendación para turismo que extrae las preferencias de los usuarios para brindar recomendaciones personalizadas (Abbasi-Moud et al., 2021). Para ello, se utilizan las opiniones de los usuarios en las redes sociales de turismo para extraer sus preferencias. Luego, se procesan los comentarios y se agrupan semánticamente, para ser analizados sentimentalmente con el fin de detectar las preferencias de un usuario turista. De manera similar, todas las reseñas de los usuarios que son agregadas sobre una atracción se utilizan para extraer las características de estos puntos de interés.

Finalmente, se propone un sistema de recomendación que compara semánticamente las preferencias de un usuario con las características de las atracciones para sugerir los puntos de interés más coincidentes para el usuario. Los resultados de la evaluación revelan una alta eficiencia del sistema de recomendación propuesto en términos de precisión, recordatorio y medida f.

El sistema de recomendación propuesto consta de tres etapas. En la primera etapa, se extraen las preferencias del usuario de sus comentarios y reseñas. De igual forma, en la segunda etapa se extraen las características de los atractivos turísticos de las reseñas de los turistas. Finalmente, en la tercera etapa, se presentan las recomendaciones adecuadas

en base a la información contextual, así como la similitud entre las preferencias de los usuarios con las características de los atractivos turísticos. La información contextual utilizada en este método incluye el clima, la hora, la ubicación y las preferencias del usuario.

Figura 3-3-11: Esquema general de la recomendación de las atracciones o lugares de interés.



Fuente: (Abbasi-Moud et al., 2021)

La idea principal del sistema de recomendación propuesto es comparar las preferencias de un turista con las características de las atracciones cercanas y devolver las atracciones más similares. Para calcular la similitud de las preferencias de un usuario con las características de una atracción, se calcula la similitud máxima de cualquiera de los elementos de preferencia del usuario con respecto a todas las características de la atracción. Por lo tanto, estas similitudes máximas se promedian para todos los elementos de preferencia del usuario. Finalmente, si las condiciones climáticas son nevadas, lluviosas o tormentosas, se dará prioridad a las ubicaciones interiores sobre las exteriores para reducir los problemas de visitas. Por otro lado, si se espera que haga buen tiempo, se mostrarán los resultados originales. El esquema general de la recomendación se muestra en la Figura 3-3-18.

Fortalezas:

- El sistema utiliza la información contextual de la hora, la ubicación y el clima para filtrar elementos inadecuados y aumentar la calidad de las sugerencias con respecto a la situación actual.
- Este trabajo propone un sistema de recomendación para turismo consciente del contexto mediante el análisis de sentimientos. En este sistema, se aprovechan el procesamiento de texto y el análisis de opiniones para extraer las preferencias de los usuarios.

Limitaciones:

- Aunque en esta investigación se amplía la idea inicial al extraer características agregadas de las atracciones de las reseñas de usuarios y proponer un sistema de recomendación personalizado y utilizar el contexto del usuario en consideración, no se evidencia alguna característica que se aplique a un modelo de visita guiada.
- Si bien el tiempo, la ubicación, el clima, las preferencias de los usuarios y las características de las atracciones se han aprovechado como información contextual, se ha descuidado otra información contextual, como el tráfico de las rutas, las diversas condiciones de los individuos, los usuarios grupales y su entorno.

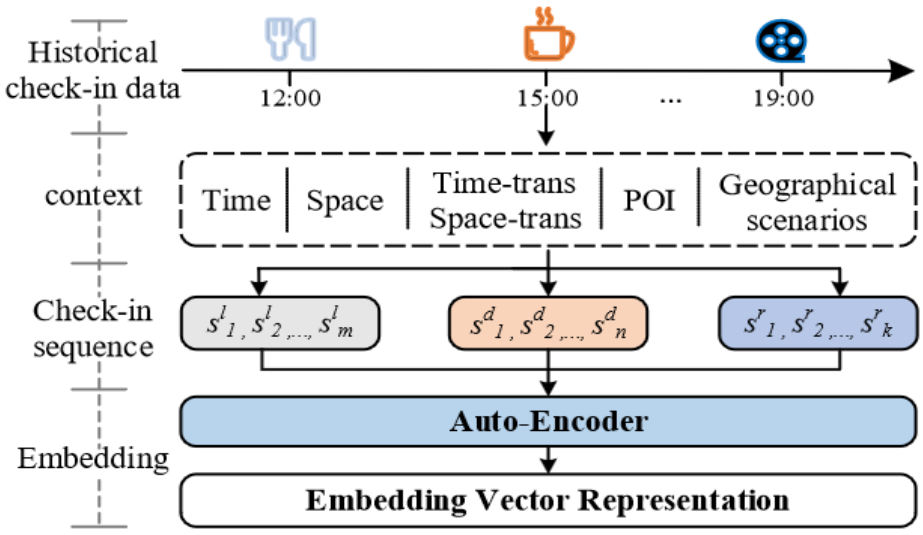
3.5 Trabajos relacionados con el modelado del usuario y su personalización

Teniendo en cuenta que planificar los itinerarios de visitas al extranjero lleva mucho tiempo, es laborioso y conlleva a desafíos relacionados con los Idiomas hablados en entornos desconocidos en otros países, hace que los riesgos de realizar estas actividades aumenten, para ello se han creado sistemas que consideran las preferencias del usuario y su personalización. A continuación, veremos algunos trabajos propuestos sobre dicho tema.

3.5.1 Recomendaciones de puntos de interés basadas en preferencias inmediatas del usuario e influencias contextuales

El modelo propuesto por Li et al, (2023) se incorpora la extracción de características multifactor y un módulo de incorporación para extraer información de características contextuales de la secuencia de registro histórico de un usuario. También incluye un módulo de modelado de preferencias compuestas del usuario que agrega las preferencias de intereses del usuario a través de una red de atención de dos capas y captura las preferencias de intereses de los puntos de interés objetivo para obtener representaciones de preferencias a largo y corto plazo. Además, el modelo presenta un módulo de predicción recomendado que utiliza una función de predicción para generar sugerencias de puntos de interés luego de la fusión ponderada de preferencias inmediatas, a corto y largo plazo.

Figura 3-3-12: Extracción e incorporación de características contextuales multifactoriales.



Fuente: (Li et al., 2023).

Antes de extraer las preferencias del usuario, la información contextual de las características sobre intervalos temporales, espaciales y espaciotemporales, escenas geográficas e interacciones de puntos de registro se extraen de los registros el registro de los históricos del usuario, formando secuencias instantáneas, de largo y de corto plazo. Y luego se transforman en representaciones vectoriales integradas mediante técnicas de autocodificación para ayudar al modelo a capturar mejor la relación entre el usuario y el PDI, ver figura 3-3-23.

Fortalezas:

- Modelado integral de preferencias: el modelo incorpora preferencias inmediatas, a largo plazo y a corto plazo, proporcionando una representación integral de las preferencias del usuario.
- Mecanismo de atención: el uso de una red de atención de doble capa permite que el modelo ajuste dinámicamente los pesos según los intereses del usuario, mejorando la capacidad de generalización y la precisión de la predicción.

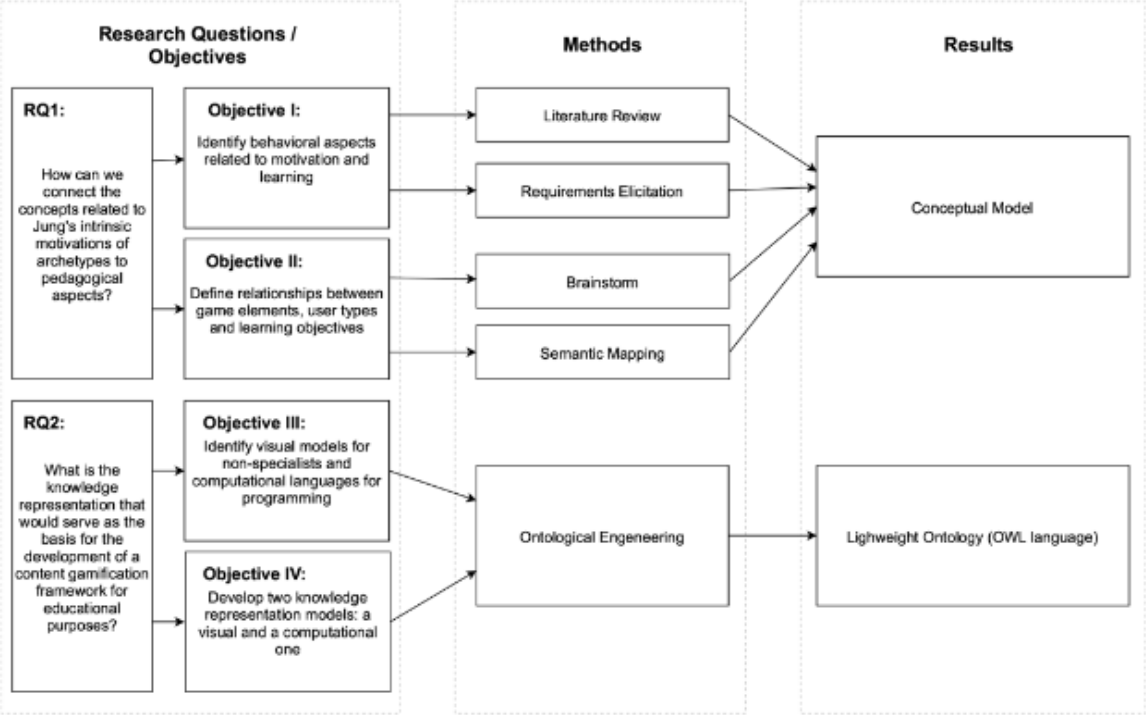
Limitaciones:

- Dependencia de datos históricos: el método de extracción de preferencias inmediatas depende únicamente de la secuencia de registro histórico de un usuario, lo que potencialmente resulta en preferencias de usuario menos precisas cuando los datos históricos son escasos.
- Problemas de escasez de datos y arranque en frío: como muchos sistemas de recomendación, el modelo puede enfrentar problemas de escasez de datos y arranque en frío, lo que puede afectar la precisión de las recomendaciones.

3.5.2 Una ontología para modelar perfiles y actividades de usuarios en la educación gamificada

La ontología ligera analizada en el trabajo de Palomino et al, (2022) se puede utilizar para desarrollar un marco de gamificación de contenido con fines educativos al proporcionar una representación modelo de conceptos y sus relaciones, específicamente adaptada al ámbito educativo. Esta ontología, desarrollada en OWL, está diseñada para representar modelos de conocimiento ricos y complejos, lo que la hace adecuada para su uso en el desarrollo de sistemas semánticos inteligentes. Además, la ontología proporciona una representación visual del conocimiento que puede ser entendida y utilizada por especialistas no informáticos, como los profesores, y puede servir como base para el desarrollo de un marco de gamificación de contenidos con fines educativos. Además, la ontología conecta conceptos relacionados con las motivaciones intrínsecas de los arquetipos de Jung con aspectos pedagógicos, proporcionando una base para integrar elementos motivacionales y pedagógicos en el marco de gamificación. Ver figura 3-3-24.

Figura 3-3-13: Procedimiento del estudio abordado.



Fuente: (Palomino et al., 2022)

Fortalezas:

- El desarrollo de una ontología ligera que proporciona un modelo estructurado para representar conceptos de gamificación, aspectos pedagógicos y elementos motivacionales en el ámbito educativo.
- Esta ontología puede respaldar el desarrollo de un marco de gamificación de contenido con fines educativos al integrar elementos motivacionales y pedagógicos en el diseño, mejorando en última instancia la experiencia del usuario y los objetivos de aprendizaje.

Limitaciones:

- La eficacia de la gamificación en la educación sigue siendo un tema de debate, con conclusiones contradictorias sobre su impacto en los resultados del aprendizaje.
- Puede haber desafíos en la implementación de la ontología ligera y el marco de gamificación en diversos entornos educativos con diferentes antecedentes estudiantiles y necesidades psicológicas.

3.6 Tabla comparativa de los trabajos relacionados

En esta sección se muestra la síntesis de los trabajos relacionados con las temáticas de la computación sensible al contexto, la computación ubicua, los sistemas de recomendación, el modelado de visitas guiadas contextualizadas, el modelado del usuario y su personalización, y técnicas de machine learning empleadas. La tabla 3-1 ilustra si los trabajos relacionados consideran cada una de las características más importantes de los sistemas informáticos ubicuos que integran la definición del modelo propuesto. El ícono '√' indica que los autores incluyen la característica mencionada, el ícono 'X' que no incluyen la característica. Y el ícono de interrogación '?', indica que no es claro si los autores consideran el criterio.

El modelo de recomendación para visitas guiadas es un trabajo que relaciona los conceptos de computación ubicua, sensible al contexto, y los sistemas de recomendación con el fin de proporcionar a los usuarios sugerencias y recomendaciones de lugares y servicios que se adapten a su perfil y contexto, en la tabla 3-3-1, se puede observar una comparación entre algunos trabajos que aplican los conceptos de sensibilidad al contexto, modelado del usuario y su personalización, y el modelado de visitas guiadas contextualizadas, todos en combinación con las ideas básicas de la computación ubicua, los sistemas de recomendación y técnicas de machine learning. En algunos trabajos se puede observar la adaptación de algunos de estos aspectos considerados, pudiendo concluirse que algunos de estos trabajos consideran la sensibilidad al contexto, otros la computación ubicua, otros consideran los sistemas de recomendación, en otros el modelado del usuario y su personalización, así como el modelado de visitas guiadas según el contexto, y algunos otros consideran algunas técnicas de machine learning.

El modelo propuesto en esta tesis considera las fortalezas y las técnicas que utilizan los trabajos revisados anteriormente, adicionalmente, considera otras técnicas que pueden ser de utilidad para mejorar la efectividad de este tipo de modelos en entornos de visitas guiadas sensible al contexto.

Tabla 3-3-1: Tabla comparativa de los trabajos relacionados

Trabajos Relacionados	Computación Sensible al Contexto	Computación Ubicua	Sistema de Recomendación	Modelado del Usuario y su Personalización	Técnicas de Machine Learning
(Alkhafaji et al., 2020)	√ Visitas en grupo o por si solo, tiempo de la visita, moviéndose, sentado	√ Tecnologías móviles y wearables, servicios móviles basados en la ubicación	X	√ Tópico herencia cultural, preferencias del usuario	X
(Esmaeili et al., 2020)	X	X	√ Híbrido basado en redes sociales , recomienda atracciones turísticas	√ Busca similitudes en sus intereses confianza, reputación relaciones sociales	X
(Schürholz et al., 2020)	√ Incidentes de incendios forestales, volúmenes de tráfico	√ Estaciones de monitoreo de calidad del aire	X	√ Perfil de salud del usuario	√ Red neuronal, Deep learning
(Zheng et al., 2020)	X	X	√ Orientado al punto de interés	√ Historial de check-in, preferencias	X
(Abbasi-Moud et al., 2021)	√ Tiempo, ubicación y clima	?	√ Compara las preferencias del usuario con las características del lugar	√ Preferencias, reviews	√ Clustering semántico, análisis de sentimientos
(Agac et al., 2021)	√ Actividades del usuario, fumando en grupo, fumando solo	√ Sensores de movimiento en tecnologías móviles y wearables	X	X	√ Support vector machine, random forest, multilayer perceptron
(Islam & Fonzone, 2021)	√ Transporte público hora de salida, hora de llegada, interrupción nivel de ocupación	√ Información basada en teléfonos inteligentes	X	X	X
(Ghafoori et al., 2022)	√ Sensiblidad situacional, Red conceptual de información física de datos detectados por sensores	√ Red de sensores ubicua, sistema de información geográfico ubicuo	X	X	X
(Liang, 2022)	√ Entrevistas, bandas, relojes, cámaras, sensores de temperatura, ambiente del cuarto, hora de acostarse, hora de levantarse	√ Computación del sueño ubicua	√ Recomendaciones para mejorar el sueño, basados en conocimiento, híbridos, colaborativos	√ Género, edad, datos históricos de seguimiento automático, modelado de usuarios personalizado	X
(Palomino et al., 2022)	X	X	X	√ Personalización individual, características, perfil, modelo de personalización	X

Aldayel et al., 2023	X	X	√ Filtrado colaborativo, recomienda atracciones	X	√ Deep learning
(Arif et al., 2023)	X	? Plataforma blockchain de ethereum, Decentralized technology	√ Sistema de recomendación de criterios múltiples	√ Características personales, calificación de destinos	√ Red neuronal
(Li et al., 2023)	√ Intervalos temporales, geográficos y espacio-temporales	√ El recorrido del viaje móvil del usuario	√ Modelo de recomendación basado en preferencias, recomendación de POI personalizada	√ Check-in histórico de los usuarios	X

Fuente: El autor.

Las técnicas identificadas en estos trabajos se consideran importantes ya que son de gran relevancia para que un usuario pueda tener una buena experiencia en los recorridos turísticos. Las recomendaciones de información deben adaptarse según las preferencias, y el contexto no lo solo del usuario sino del lugar que se esté visitando. Los aspectos de la adaptación deben aprovechar las ventajas que ofrece la información del contexto en cuanto a su ubicación. Las técnicas de machine learning permiten mejorar las recomendaciones al considerar el análisis de sentimientos aplicado a los textos de las opiniones registradas por los usuarios acerca de los lugares previamente visitados. Los trabajos revisados no cubren o cubren parcialmente los criterios indicados, por lo cual se propone realizar el modelo que se presenta en este trabajo y cubrir todos los criterios evaluados.

3.7 Conclusiones del capitulo

La revisión del estado del arte permite ver que los conceptos considerados en los trabajos relacionados son los más usados para modelar y personalizar los usuarios en un intento de adaptarlos a un entorno determinado. Los trabajos relacionados con la computación sensible al contexto han demostrado ser un desafío a la hora de ser desarrollados e implementados, además es necesario conocer la ubicación de los usuarios y es difícil el lograr su participación ya que no quieren perder su privacidad. Sin embargo, la sensibilidad al contexto es una característica importante porque permite que los sistemas tengan una mejor comprensión sobre el estado de los usuarios que integran el sistema, sin olvidar que la información del contexto puede ser muy variada y muchas veces puede quedar por fuera del alcance de la aplicación. Los trabajos relacionados con la computación ubicua

mencionan que los usuarios pueden recibir servicios adaptados a su perfil, contexto y ubicación, desde cualquier lugar y momento. Pero la computación ubicua se concentra en aspectos muy técnicos que descuidan la aplicación del modelo del usuario y su personalización al centrarse en aspectos de requisitos de diseño.

Los trabajos relacionados con los sistemas de recomendación consideran el modelado del usuario y su personalización, además de la definición de un perfil para hacer comparaciones, pero descuidan el contexto con información que podría mejorar las recomendaciones de estos sistemas. Estos sistemas no solo incluyen información sobre artículos, los usuarios y sus transacciones, sino que también contienen información sobre las relaciones sociales de los usuarios y sus opiniones. Estos trabajos no muestran uso de la computación ubicua ya que los datos son recopilados por medio de encuestas a los usuarios que se encuentran cerca de los lugares visitados. Los trabajos relacionados con el modelo de visitas guiadas contextualizadas muestran estudios donde realizan la ubicación del usuario de manera móvil, y destacan el transporte relevante a la ubicación, para facilitar la planificación del viaje, como información del contexto solo se considera la ubicación geográfica del usuario que es obtenida por medio de la función de posicionamiento del teléfono móvil. Si bien en estos trabajos se considera el tiempo, la ubicación, el clima, las preferencias de los usuarios y las características de los lugares como información contextual, se ha descuidado otra información contextual, como el tráfico de las rutas, las condiciones del usuario, los usuarios grupales y su entorno, lo que puede limitar la capacidad del sistema para atender todos los requisitos del usuario.

Los trabajos relacionados con el modelado del usuario y su personalización no consideran la adaptación del usuario a la sensibilidad del contexto, y algunos trabajos utilizan algoritmos de aprendizaje no supervisados que son débiles para medir el rendimiento del modelo, los trabajos presentados tienen limitaciones para capturar de manera efectiva el proceso de evolución de los intereses del usuario para manejar sus preferencias futuras.

En el capítulo siguiente se presentan los componentes que integran el modelo propuesto del sistema de recomendación para visitas guiadas, basado en computación ubicua y sensible al contexto. La caracterización de los elementos que integran el modelo del usuario, del lugar y del contexto. La ontología desarrollada para el representar el sistema de visitas. Los modelos de sistemas de recomendación implementados, así como un

modelo de análisis de sentimientos y para finalizar el capítulo de evaluación usando las métricas de desempeño de cada modelo.

4. Modelo Propuesto

A continuación, se presenta la caracterización de las entidades y conceptos que hacen parte del modelo del sistema de recomendación para visitas guiadas tales como el usuario, sus preferencias genéricas, las reseñas del usuario sobre un sitio de interés, el sitio de interés y el horario de las actividades que se hacen en el sitio de interés. En esta caracterización expondremos el modelo del usuario y su personalización, el modelo del lugar (punto de interés POI) (Hu et al., 2015) y el modelo del contexto aplicado al modelo de visitas guiadas contextualizadas. Después, se presenta la ontología *tourist*, que conforma el conjunto de conceptos y categorías en el modelo de visitas guiadas contextualizadas indicando sus propiedades y relaciones, además de la definición de reglas de inferencia que validan la ontología en los escenarios propuestos.

A partir de la ontología desarrollada se muestran varios modelos de sistemas de recomendación que se basan en datos simulados que partiendo de la ontología se adaptan a modelos de datos en pandas para ser procesados en python para ser aplicados en los diferentes enfoques de sistemas de recomendación abordados en la propuesta. En este trabajo consideraremos la aplicación del modelo en varios enfoques de sistemas de recomendación, como son; de filtrado colaborativo basado en el usuario, filtrado basado en conocimiento usando información del perfil del usuario, filtrado usando un algoritmo de K-means y usando un filtrado basado en descomposición en valores singulares y finalmente un filtrado en un sistema de recomendación híbrido.

El modelo nos ofrece la posibilidad de usar las reseñas del usuario en la técnica de aprendizaje de máquina de análisis de sentimientos para determinar el nivel de aceptación del sitio de interés por parte del usuario. Y finalmente expondremos las métricas usadas con los resultados obtenidos que nos permiten validar el modelo.

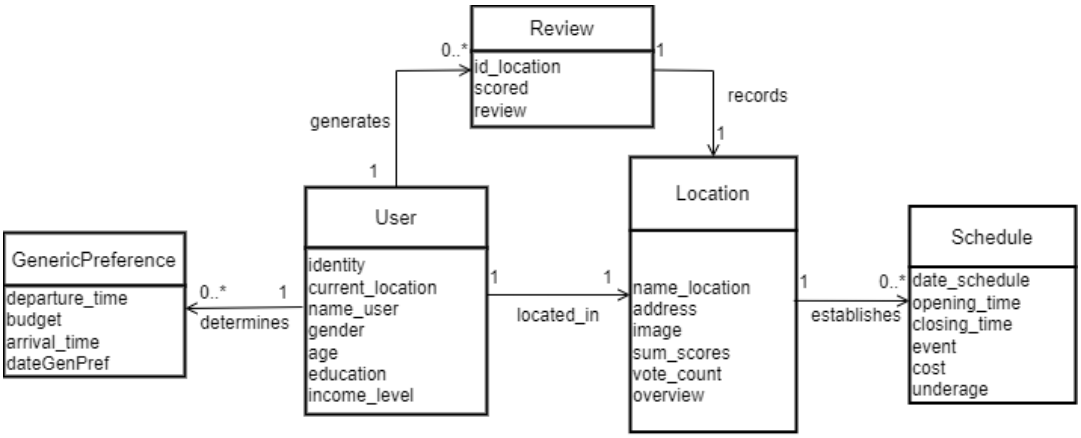
4.1 Caracterización de los conceptos asociados al sistema de recomendación para visitas guiadas

A continuación, tenemos los principales conceptos que componen el dominio del sistema de recomendación para visitas guiadas, se presentan en forma de diagrama de clases que describe la estructura del sistema, el diagrama ofrece una perspectiva conceptual para representar los conceptos del dominio.

4.1.1 Modelo del sistema de recomendación para visitas guiadas

Los usuarios que están realizando una actividad turística, están menos dispuestos a esperar para obtener información sobre un lugar, está situación muestra la necesidad que tienen los sistemas turísticos por conocer las necesidades de los usuarios y estar preparados para recibirlos con una variedad de servicios personalizados y actualizados según esta característica (Buhalis & Law, 2008). Respondiendo a esta necesidad se pretende hacer una representación de un modelo para visitas que además de considerar el lugar y al usuario considere las preferencias del usuario y las características del lugar como por ejemplo los horarios de sus eventos para que se consideren en una recomendación más personalizada según las necesidades del usuario. El modelo propuesto considera como punto de partida los siguientes elementos:

Figura 4-1: Modelo del sistema de recomendación para visitas guiadas.



Fuente: El autor.

- **GenericPreference:** Constituye las preferencias generales del usuario para representar su itinerario ya que constituye la hora que le gustaría salir, el dinero que está dispuesto a gastar, la hora que le gustaría regresar y la fecha de dicha preferencia. Para cada GenericPreference es necesario aclarar que esta es diferente para cada día, por lo cual se debe definir la fecha.
- **User:** Este concepto representa la información del usuario, además indica la ubicación actual del lugar en donde este se encuentra. El usuario pude hacer muchas reseñas y calificaciones de cada lugar visitado.
- **Location:** Representa el punto de interés, además de sus datos básicos considera la suma de todas las calificaciones que le han realizado todos los usuarios, así como el total de usuarios que han votado. Estos datos se usan para calcular las métricas de los lugares mejores calificados para mostrarlos en las recomendaciones. Igualmente cuenta con un overview textual para usarlo en filtrado colaborativo usando un algoritmo que calcula la distancia entre los lugares que tienen un overview más parecido para recomendarlo según dicha característica.
- **Schedule:** Este concepto contiene las propiedades para representar un evento o una actividad realizada en un lugar. Un lugar puede tener muchos schedules. Es importante aclarar que en este modelo consideramos si a un evento pueden asistir menores de edad con la propiedad underage.
- **Review:** Se usa para representar las reseñas realizadas por los usuarios a un lugar, también considera una calificación numérica de 0 a 5. Estos datos son usados para

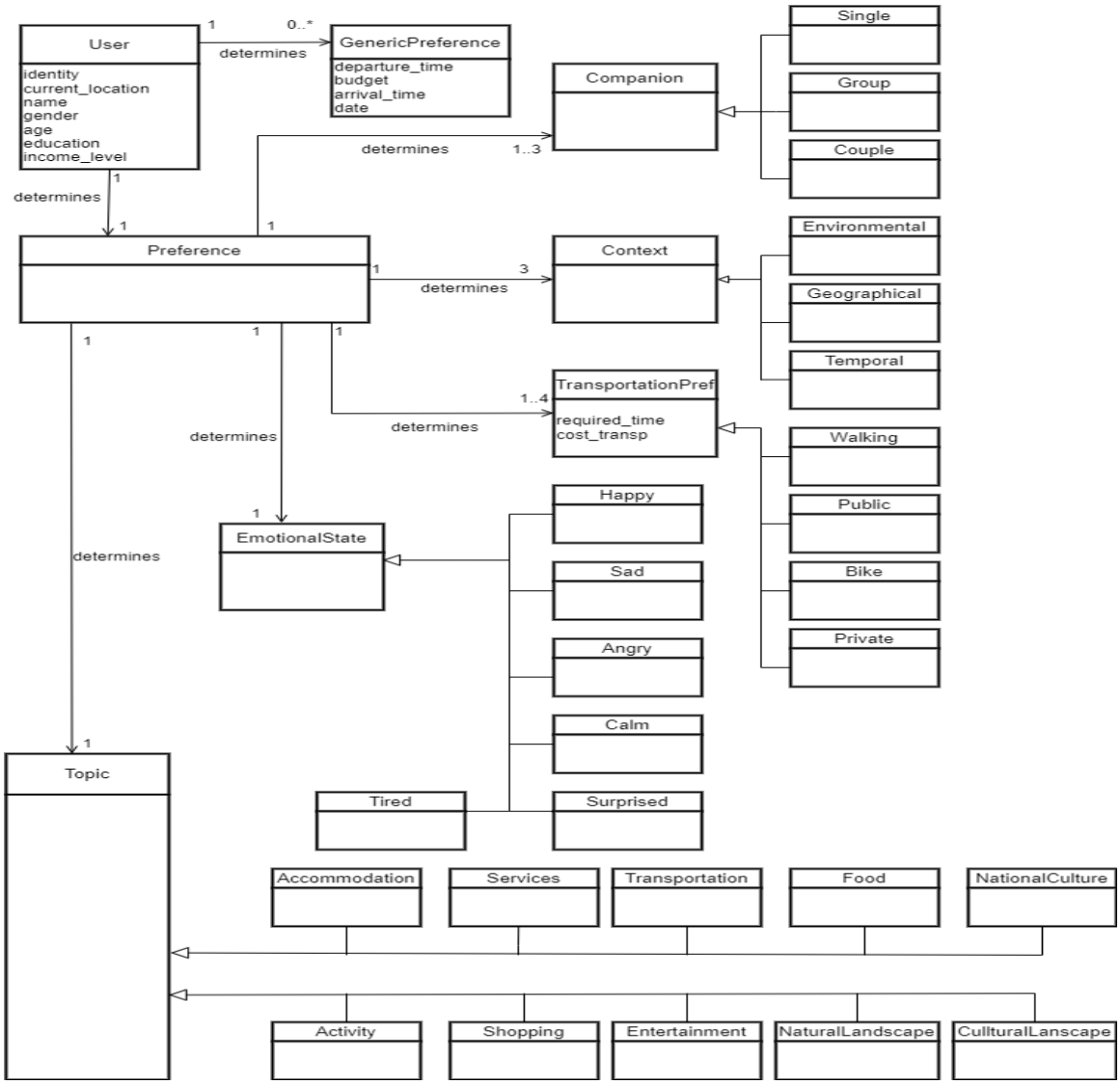
hacer las predicciones sobre un lugar no visitado por un usuario, en los distintos modelos de recomendación planteados en este trabajo.

La figura 4-1, muestra el modelo inicial para el modelo de sistema de recomendación para visitas guiadas. Este modelo permite representar los lugares que se encuentran al exterior en el espacio abierto, al interior de otro lugar de mayor tamaño, y los lugares que representan un servicio al interior de alguna organización. Es necesario aclarar que, para modelar el usuario y la forma de personalizarlo, en este trabajo se profundizará en la representación de sus preferencias que incluyen su estado emocional, su estado social con su compañía para viajar, sus preferencias para transportarse a un lugar, los tópicos de interés para visitar un lugar determinado y las preferencias del contexto. Igualmente, para modelar el lugar y la forma de adaptarlo al usuario, en este trabajo se incluirán las características del lugar que están conformadas por las preferencias de transporte para llegar a dicho lugar, el tópico de interés que etiqueta al lugar, y la información del contexto actual al momento de visitarlo.

4.1.2 Modelo del usuario y su personalización

Las aplicaciones móviles permiten acceder casi a cualquier hora y en cualquier lugar, a la información y a las preferencias del usuario, así como a su ubicación, esta información puede ser usada y filtrada evitando congestiones en el proceso de acompañamiento durante toda la actividad relacionada con una visita a un lugar determinado (Höpken et al., 2010).

Figura 4-2: Modelo del perfil del usuario.



Fuente: El autor.

En este modelo se busca representar las preferencias del usuario, considerando su estado emocional con el concepto de EmotionalState, se estado social con el concepto de Companion, representar las preferencias de transporte con el concepto de TransportationPref, conocer el tópico de interés del usuario para buscar un lugar que tenga dicho tópico, esto se representa con el concepto de Topic y finalmente representar el contexto el cual puede ser geográfico, temporal y ambiental, representado en los conceptos; Geographical, Temporal, y Environmental respectivamente como se muestra a continuación. Estos conceptos permitirán personalizar las recomendaciones del usuario,

contar con un perfil más completo, que considera tanto su disponibilidad como su estado emocional, además de sus intereses y preferencias para desplazarse y las condiciones ambientales de los lugares que desee visitar en un determinado momento. Y como se mencionó en el modelo anterior permite que el usuario pueda ir calificando y generando reseñas de los lugares que visitados. Todo esto enriquece el modelo propuesto y permite aplicar sistemas de recomendación y técnicas de aprendizaje de máquina.

El modelo del usuario y su personalización es representado en la figura 4-2. A continuación, se detallan cada uno de sus conceptos y sus atributos. Partiendo de los conceptos explicados previamente en el apartado anterior, a saber; User y GenericPreference, que ya han sido mencionados.

- **Preference:** Este concepto representa las preferencias del usuario las cuales pueden ser; Companion, Context, TransportationPref, EmotionalState y Topic.
- **Companion:** Este concepto se usa para indicar el estado social del usuario en su ambiente circundante, este puede se divide en tres tipos; Single, para indicar que viaja solo, Group para indicar que viaja en grupo, y Couple para indicar que está viajando en pareja.
- **Context:** Este concepto representa la información del contexto. Y se divide en; Environmental, Geographical y Temporal.
- **TransportationPref:** Este concepto se usa para indicar las preferencias del usuario para viajar a un lugar. Tiene como propiedades; required_time, para indicar el tiempo que está dispuesto a invertir para llegar a un lugar y cost_transp para indicar el monto de dinero que está dispuesto a gastar en transporte. Este concepto tiene los siguientes tipos de medios de transporte para llegar a un lugar; Walking, para caminando, Public para indicar que le gustaría viajar en un medio de transporte público, Bike para indicar que le gustaría llegar en moto y Private que indica que le gustaría viajar en un medio de transporte privado.
- **EmotionalState:** Indica el estado emocional del usuario, y se divide en los siguientes tipos; Happy, Sad, Angry, Calm, Surpriced y Tired. Este modelo permite agregar nuevos estados emocionales como Normal.
- **Topic:** Este concepto está pensado para manejarlo como una etiqueta para identificar los sitios de interés y está divido en los siguientes tipos: Accommodation, Services, Transportation, Food, NationalCulture, Activity, Shopping, Entertaintment, NaturalLandscape y CulturalLandscape. Es posible agregar nuevos tópicos.

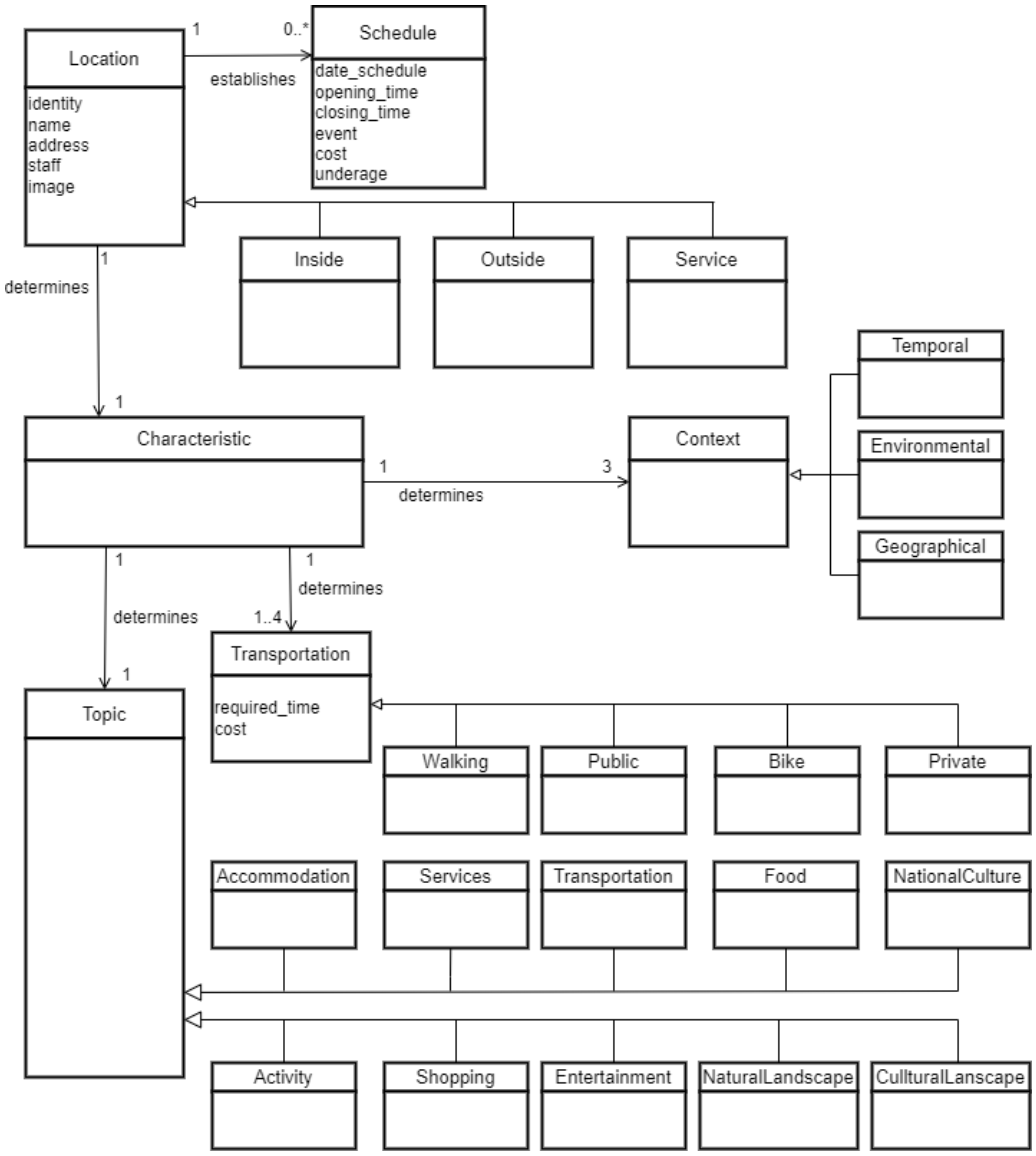
4.1.3 Modelo del lugar y su adaptación

Los teléfonos inteligentes modernos ofrecen funcionalidades muy interesantes que permiten aprovechar los servicios basados en la ubicación o LBS (por sus siglas en inglés), que determinan la ubicación del usuario para proveerle de servicios personalizados según dicha ubicación (Chen & Tsai, 2019). Esto indica que es muy importante contar con información del lugar no solo con su ubicación sino con información que enriquezca una mejor adaptación con el usuario. En el modelo propuesto se presenta una adaptación del lugar según un conjunto de características que se concentran en el concepto llamado Characteristic, y que se corresponden con las preferencias del usuario para permitir obtener una mejor recomendación, es decir que se consideran las preferencias del usuario en cuanto a las preferencias de transporte y tópico del lugar y la información del contexto que corresponde con la del lugar y la del usuario para adaptar mejor dicha personalización. Veamos los componentes que integran el modelo del lugar, ver figura 4-3. A continuación, se detallan cada uno de sus conceptos y sus atributos. Partiendo de los conceptos explicados previamente en el apartado anterior, a saber; Location y Schedule, que ya han sido mencionados.

- **Location:** Además de considerar lo que ya se ha explicado en el modelo inicial sobre este concepto. Se agrega que este concepto puede ser de tres tipos; Inside, Outside y Service.
- **Characteristic:** Este concepto representa las características del lugar las cuales pueden ser; Context, TransportationPref y Topic.
- **Context:** Este concepto representa la información del contexto del lugar. Y se divide en; Environmental, Geographical y Temporal.
- **TransportationPref:** Este concepto se usa para indicar las características para llegar a un lugar. Tiene como propiedades; required_time, para indicar el tiempo que se requiere para llegar a un lugar y cost_transp para indicar el para llegar a dicho lugar usando el medio de transporte seleccionado. Este concepto tiene los siguientes tipos de medios de transporte para llegar a un lugar; Walking, para caminando, Public para indicar que le gustaría viajar en un medio de transporte público, Bike para indicar que le gustaría llegar en moto y Private que indica que le gustaría viajar en un medio de transporte privado.

- **Topic:** Este concepto está pensado para manejarlo como una etiqueta para identificar los sitios de interés y está divido en los siguientes tipos: Accommodation, Services, Transportation, Food, NationalCulture, Activity, Shopping, Entertaintment, NaturalLandscape y CulturalLandscape. Es posible agregar nuevos tópicos.

Figura 4-3: Modelo del lugar y su adaptación.



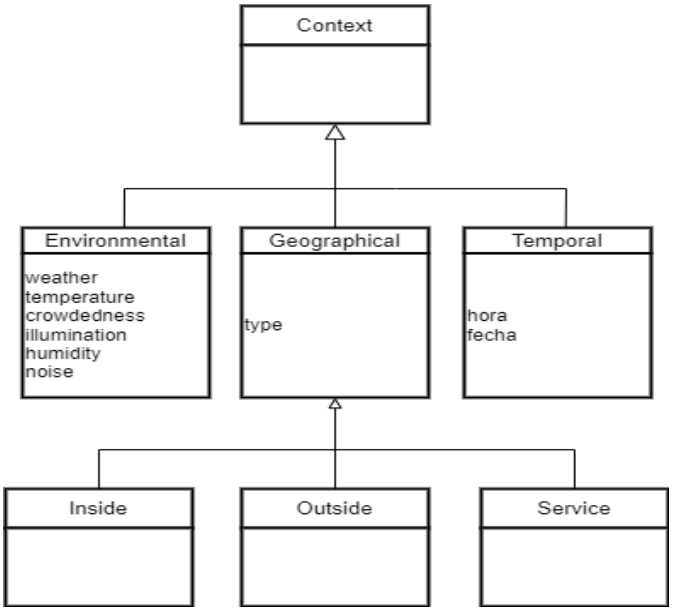
Fuente: El autor.

4.1.4 Modelo del contexto aplicado al modelo de visitas guiadas contextualizadas

El contexto se puede centrar en el ambiente físico (Abowd et al., 2002), el cual puede considerar el tiempo, la ubicación, la temperatura, la humedad, el ruido, la iluminación, etc. Toda esta información puede ser obtenida por medio de sensores, servicios web y dispositivos como los GPS para determinar su ubicación.

En este modelo vamos a considerar los siguientes conceptos para el modelo del contexto aplicado al modelo de visitas guiadas contextualizadas, tanto para las preferencias del usuario como para las características del lugar. Es decir que el usuario puede determinar sus preferencias por determinados tipos de contexto, las cuales pueden coincidir con el contexto del lugar en el que se encuentre.

Figura 4-4: Modelo del contexto aplicado a las visitas guiadas contextualizadas.



Fuente: El autor.

En la figura 4-4, podemos observar que el contexto puede ser de tres tipos: Environmental, Geographical y Temporal, donde cada uno de ellos considera las siguientes características:

- **Environmental:** Weather, Temperature, Crowdedness, Ilumination, humidity y noise.

- **Geographical:** Para determinar si un lugar se encuentra al interior, Inside, al exterior, Outside y de tipo Service para indicar si es un servicio que se ofrece en un lugar.
- **Temporal:** Para indicar el contexto del tiempo al momento actual de realizar una acción al estar ubicado en un lugar específico.

4.2 Ontología tourist del modelo de recomendación para visitas guiadas, basado en computación ubicua y sensible al contexto

4.2.1 Conceptualización de la ontología

La conceptualización de la ontología propuesta, consiste de las siguientes actividades; construcción del glosario de términos, construcción de la taxonomía de conceptos, construcción del diagrama de relaciones binarias, construcción del diccionario de conceptos, descripción de las relaciones binarias en detalle, descripción de los atributos de instancia en detalle, definición de reglas, y definición de instancias. Estas actividades se muestran a continuación.

- Construcción del glosario de términos

  En esta sección se definen los conceptos pertenecientes a dominio del modelo de recomendación para visitas guiadas y que componen la ontología.

Tabla 4-1: Glosario de términos de la ontología

Categorías	Conceptos
User	User, identity, current_location, name_user, gender, age, education, income_level, GenericPreference, departure_time, budget, arrival_time, date, Review, id_location, review
Location	Location, name_location, address, image, sum_scores, vote_count, overview, Schedule, date_schedule, opening_time, closing_time, event, cost, underage, InsideLocation, OutsideLocation, ServiceLocation
Preference	EmotionalState, Happy, Sad, Angry, Calm, Surpriced, Normal, Tired

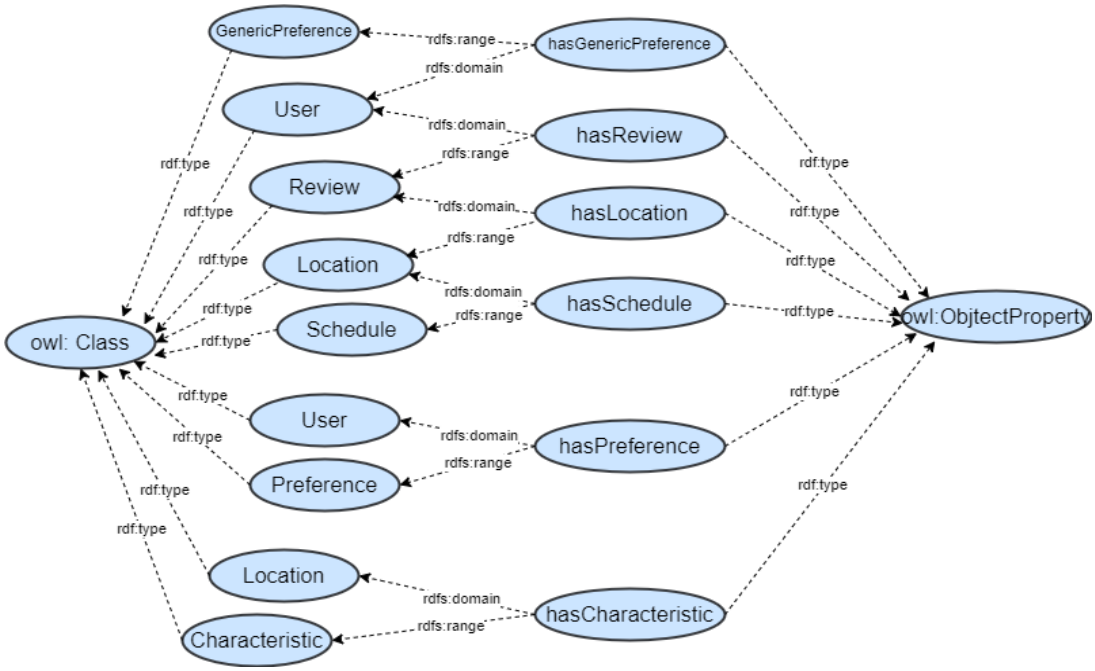
	Companion, Single, Group, Couple, Transportation, required_time, cost, Walking, Public, Bike, Private, Context, Topic, Accomodation, Services, Transportation, Food, NationalCulture, Activity, Shopping, Environment NaturalLandscape, CulturaLandscape
Context	Context, Environmental, weather, temperature, crowdedness, illumination, moisture, noise, Geographical, Temporal, hora, fecha

Fuente: El autor.

- Construcción de la taxonomía de conceptos

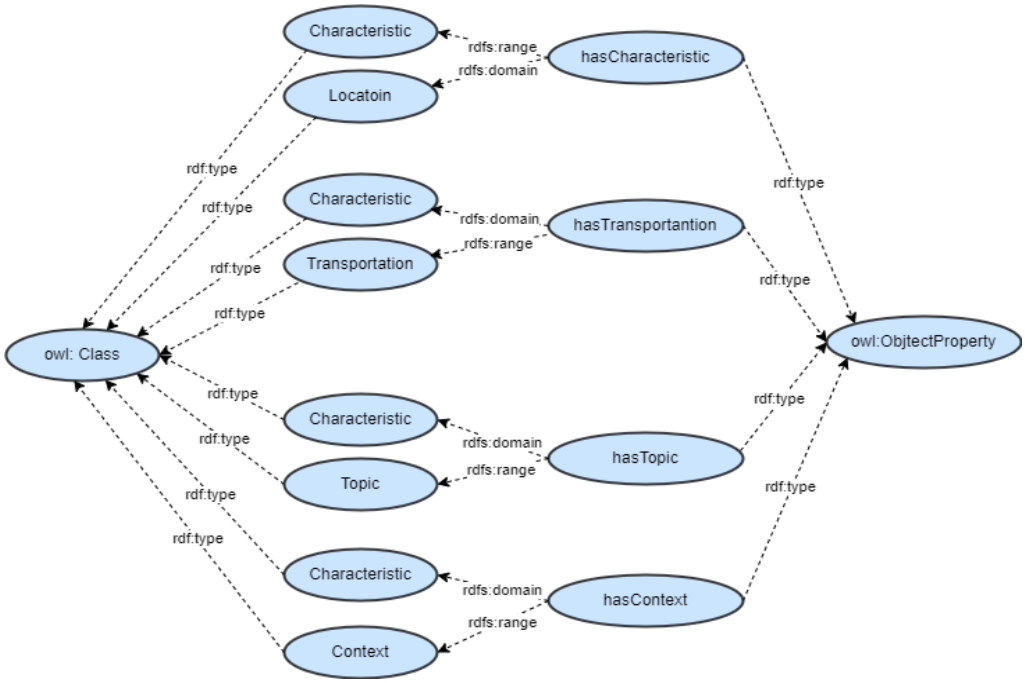
  En esta sección se puede ver la taxonomía del modelo propuesto, del modelo del lugar y la taxonomía del modelo del usuario.

Figura 4-5: Taxonomía de los conceptos que integran la ontología



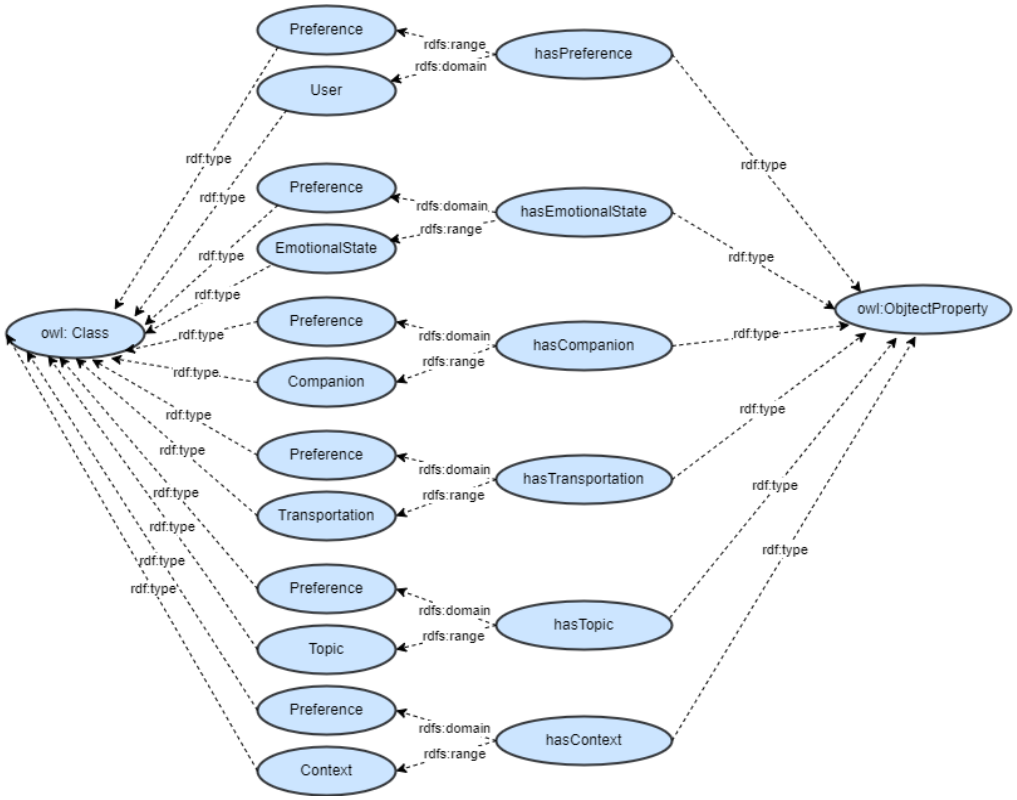
Fuente: El autor.

Figura 4-6: Taxonomía de los conceptos que integran el modelo del lugar turístico



Fuente: El autor.

Figura 4-7: Taxonomía de los conceptos del modelo del usuario

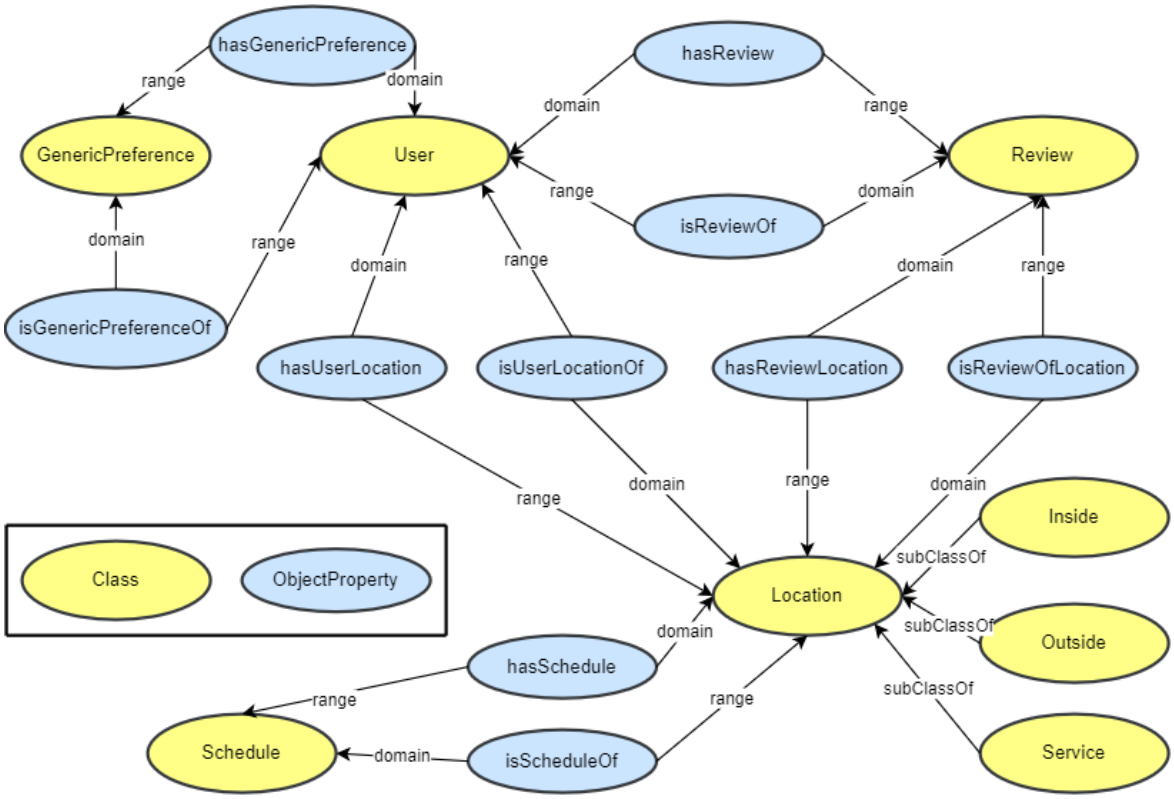


Fuente: El autor.

- Construcción del diagrama de relaciones binarias

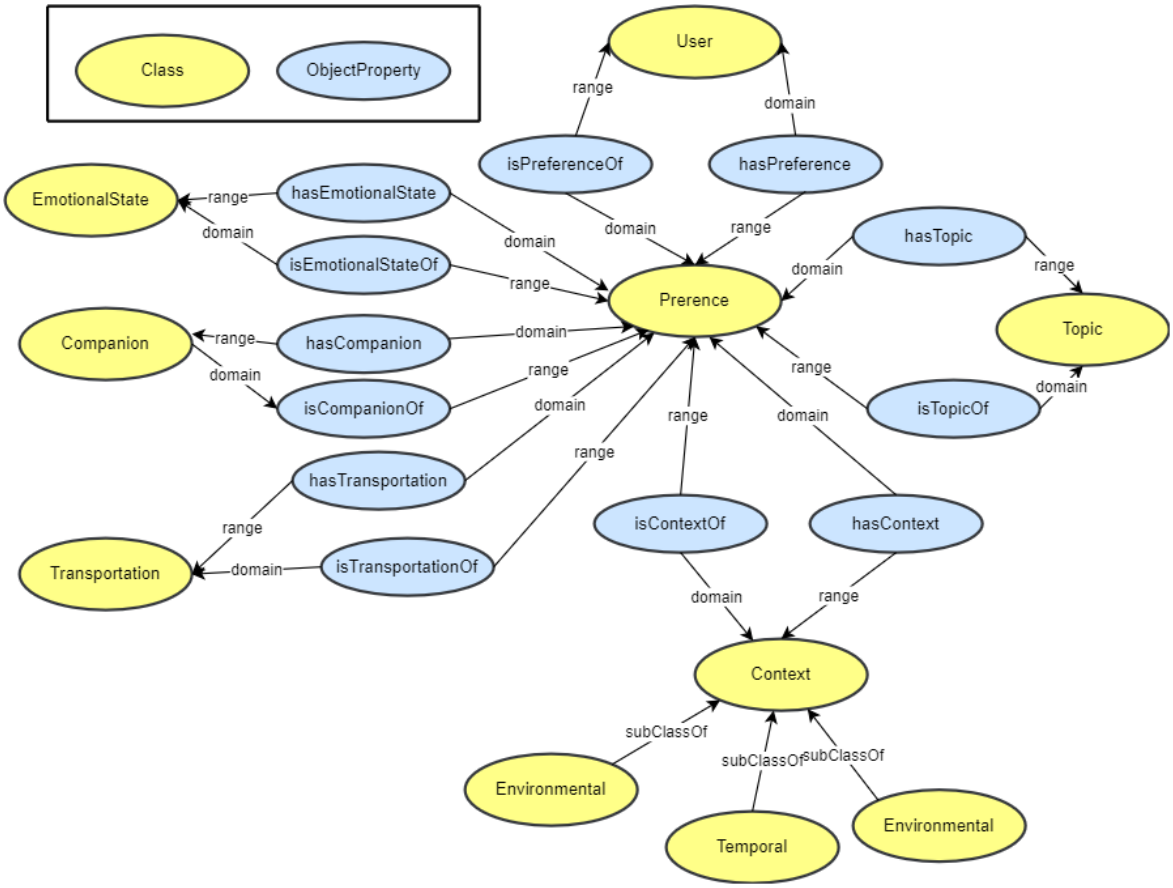
  En esta sección se pueden ver los diagramas de las relaciones binarias entre los conceptos mostrados en la taxonomía. Estas relaciones binarias son para el modelo del sistema de recomendación, para el modelo del usuario y para el modelo del lugar.

Figura 4-8: Diagrama de relaciones binarias del modelo del sistema de recomendación



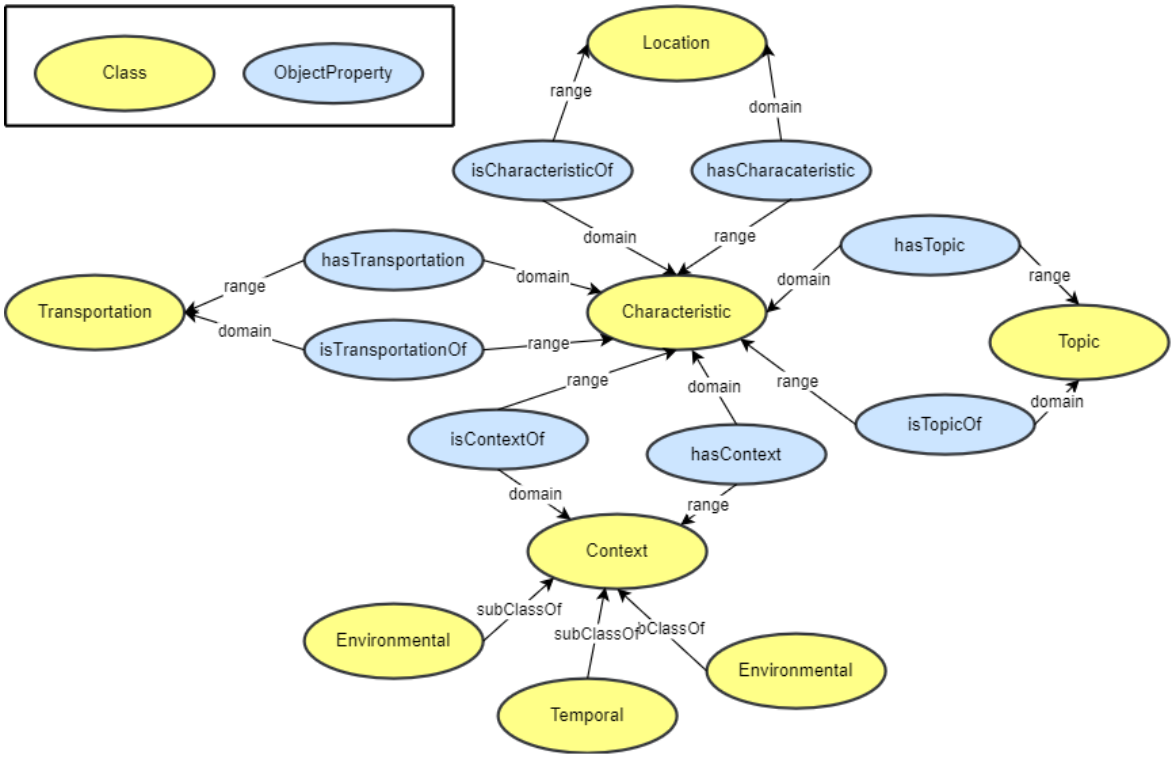
Fuente: El autor.

Figura 4-9: Diagrama de relaciones binarias del modelo de las preferencias del usuario



Fuente: El autor.

Figura 4-10: Diagrama de las relaciones binarias del modelo de las características del lugar



Fuente: El autor.

- Descripción de las relaciones binarias en detalle

Tabla 4-2: Descripción de las relaciones binarias del modelo del sistema de recomendación.

Relación	Dominio (Clase Origen)	Rango (Clase Destino)	Relación Inversa
hasGenericPreference	User	GenericPreference	isGenericPreferenceOf
hasReview	User	Review	isReviewOf
hasUserLocation	User	Location	isUserLocationOf
hasSchedule	Location	Schedule	isScheduleOf
hasReviewLocation	Location	Review	isReviewLocationOf

Fuente: El autor.

Tabla 4-3: Descripción de las relaciones binarias del modelo de las preferencias del usuario

Relación	Dominio (Clase Origen)	Rango (Clase Destino)	Relación Inversa
hasEmotionalState	Preference	EmotionalState	isEmotionalStateOf
isPreferenceOf	Preference	User	hasPreference
hasTopic	Preference	Topic	isTopicOf
isContextOf	Context	Preference	hasContext
hasCompanion	Preference	Companion	isCompanionOf
hasTransportation	Preference	Transportation	isTransportationOf

Fuente: El autor

Tabla 4-4: Descripción de las relaciones binarias del modelo de las características del lugar

Relación	Dominio (Clase Origen)	Rango (Clase Destino)	Relación Inversa
isCharacteristicOf	Characteristic	Location	hasCharacteristic
hasTopic	Characterisctic	Topic	isTopicOf
hasTransportation	Characteristic	Transportation	IsTransportationOf
isContextOf	Context	Characteristic	hasContext

Fuente: El autor.

- Construcción del diccionario de conceptos

Tabla 4-5: Diccionario de los conceptos que constituyen la ontología.

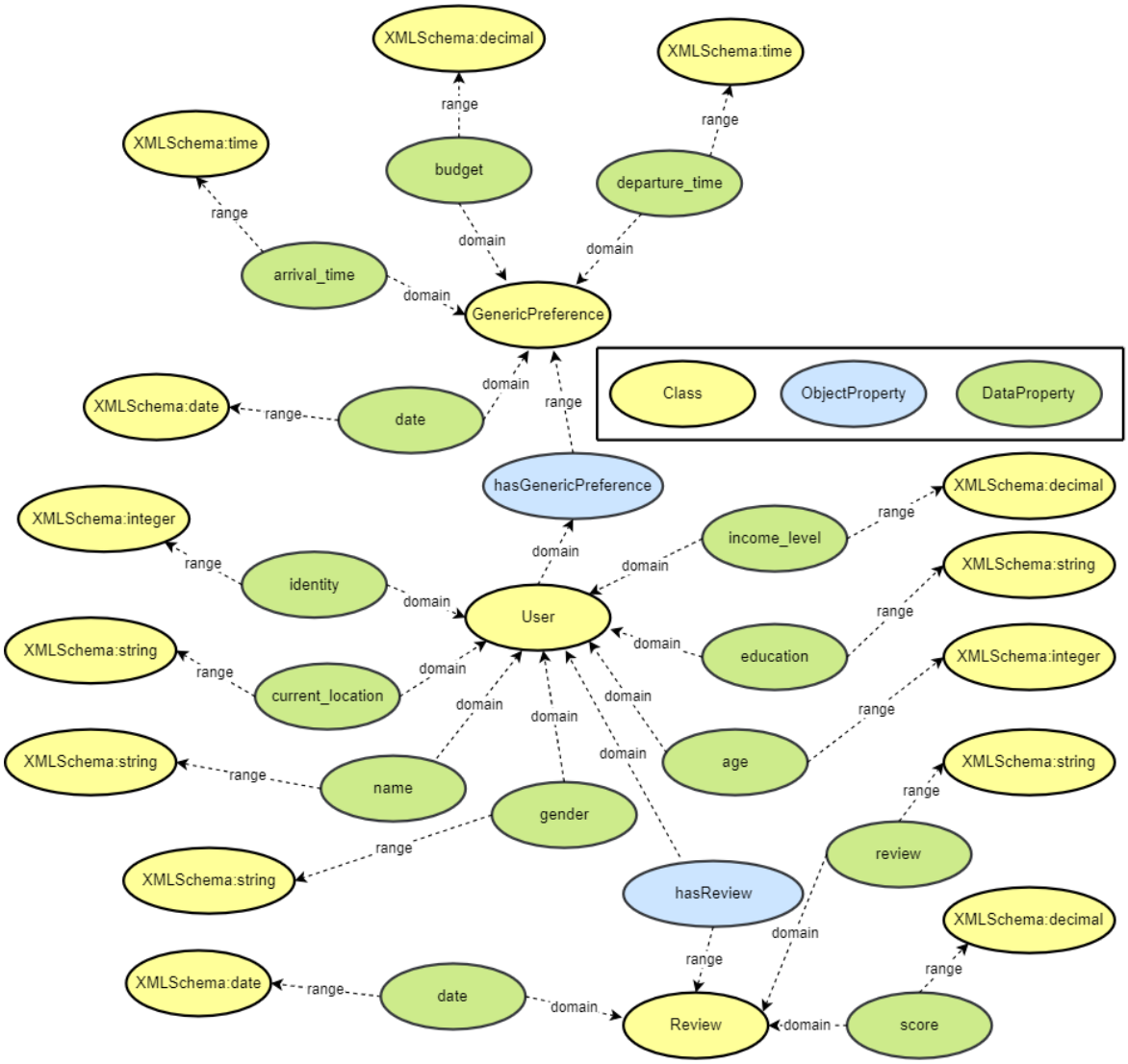
Clase (Class)	Atributos (DataProperty)	Relaciones (ObjectProperty)
GenericPreference	departure_time, budget, arrival_time, date	isGenericPreferenceOf
User	Identity, current_location, name, gender, age,	hasGenericPreference

	education, income_level	
Review	IdLocation, score , review	hasReviewLocation
Location	identity, name, address, staff, image, cost, feature	IsLocationOf, hasPreference, hasSchedule
Schedule	day, opening_time, closing_time	isScheduleOf
Preference, EmotionalSate, Companion, Context, Topic		isPreferenceOf
Transportation	required_time, cost	isPreferenceOf
Environmental	weather, temperature, crowdedness, illumination moisture, noise	isPreferenceOf

Fuente: El autor

- Descripción de los atributos de las instancias

Figura 4-11: Descripción de los atributos de las entidades del modelo



Fuente: El autor

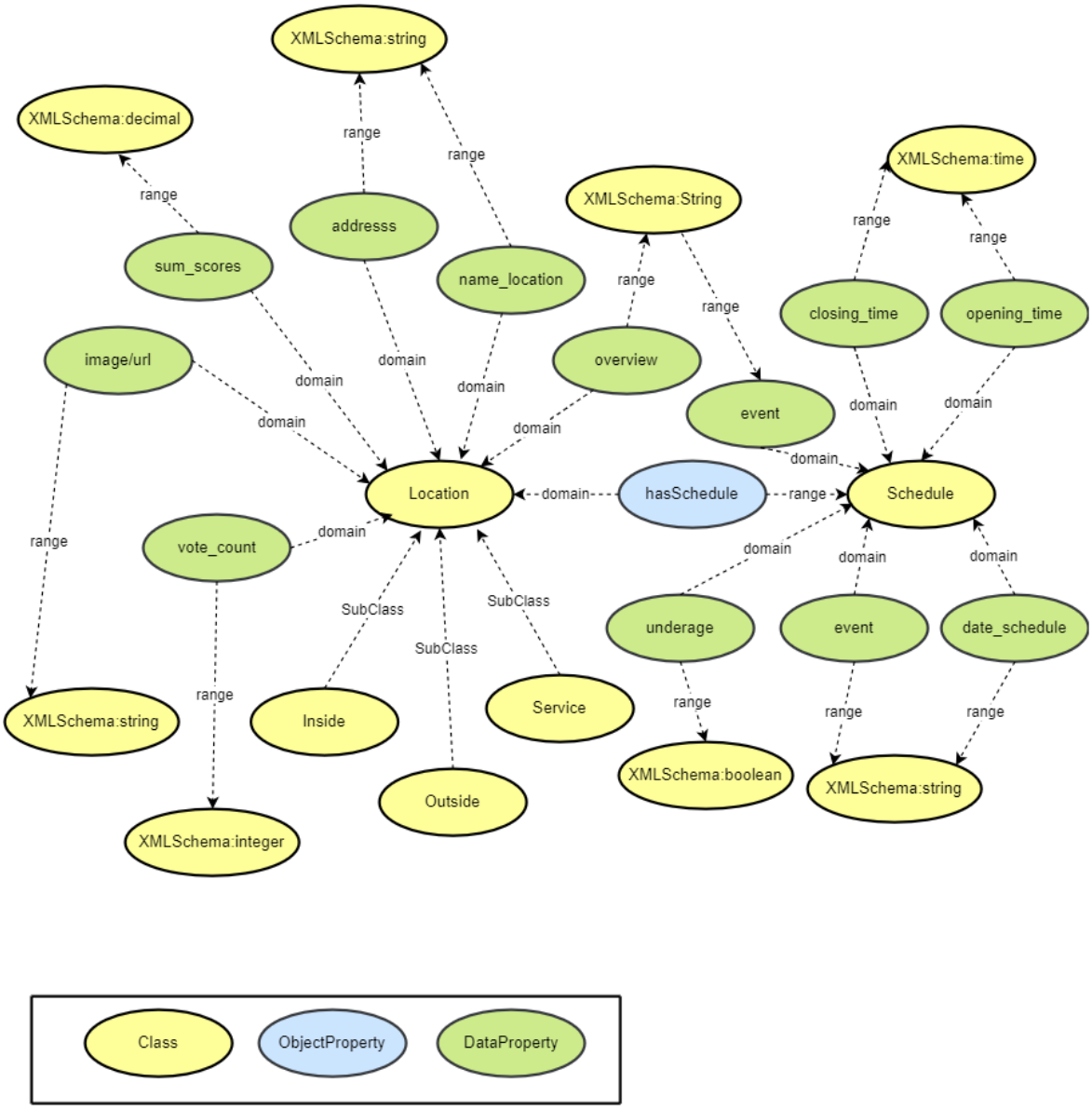
Tabla 4-6: Descripción de los atributos en detalle de las instancias del modelo

Clase	Atributo	Tipo	Cardinalidad
GenericPreference	departure_time	XMLSchema:time	1
	budget	XMLSchema:decimal	1
	arrival_time	XMLSchema:time	1
	date	XMLSchema:date	1
User	identity	XMLSchema:integer	1

	current_location	Location	1
	name	XMLSchema:string	1
	gender	XMLSchema:string	1
	age	XMLSchema:integer	1
	education	XMLSchema:string	1
	income_level	XMLSchema:decimal	1
Review	date	XMLSchema:date	1
	score	XMLSchema:decimal	1
	review	XMLSchema:String	1

Fuente: El autor

Figura 4-12: Descripción de los atributos del modelo del lugar



Fuente: El autor

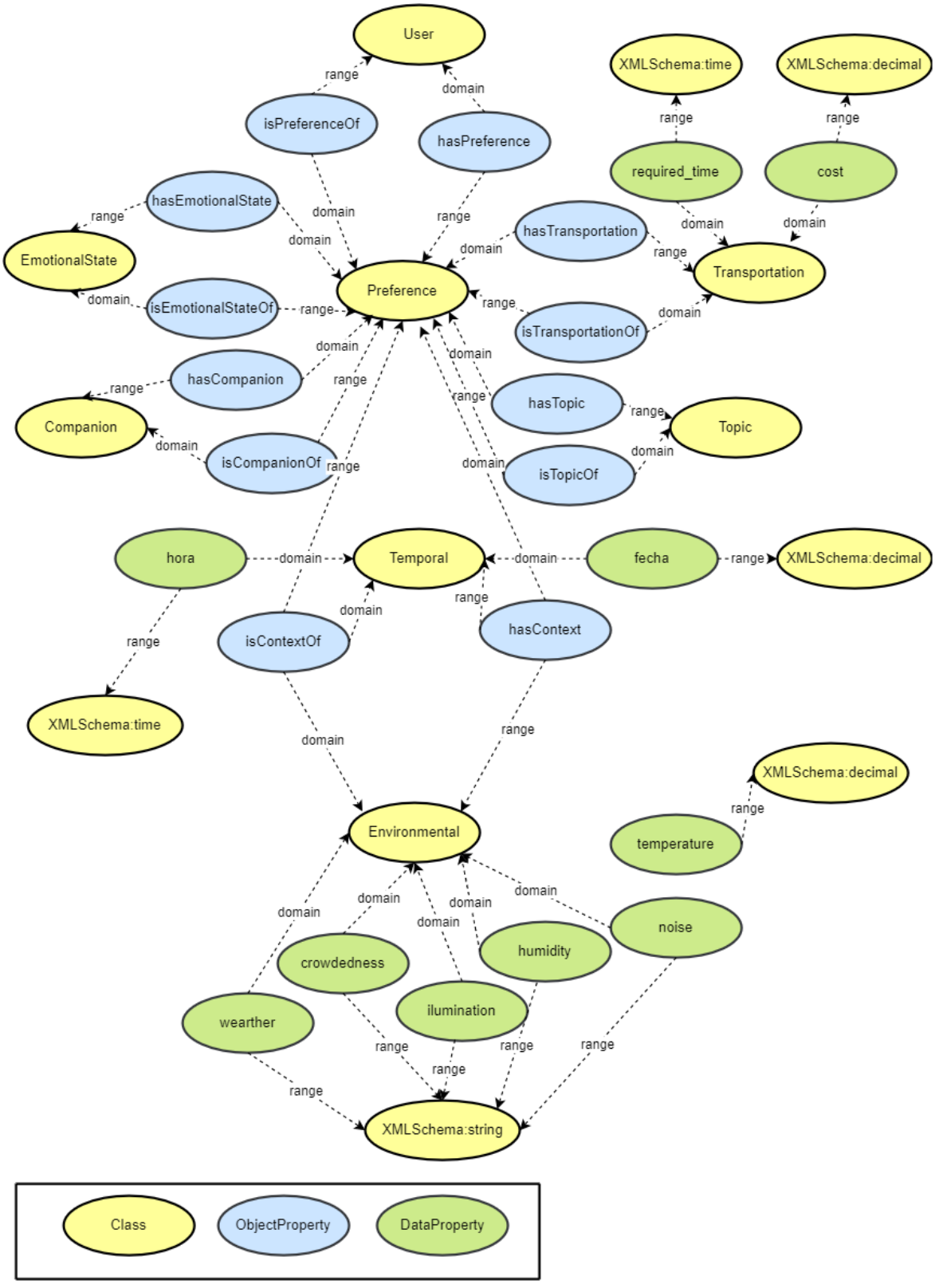
Tabla 4-7: Descripción de los atributos en detalle del modelo del lugar

Clase	Atributo	Tipo	Cardinalidad
Location	overview	XMLSchema:string	1
	name_location	XMLSchema:string	1
	address	XMLSchema:string	1

	sum_scores	XMLSchema:decimal	1
	image/url	XMLSchema:string	1
	count	XMLSchema:integer	1
Schedule	date_schedule	XMLSchema:date	1
	opening_time	XMLSchema:time	1
	closing_time	XMLSchema:time	1
	underage	XMLSchema:boolean	1
	event	XMLSchema:string	1

Fuente: El autor

Figura 4-13: Descripción de los atributos del modelo de las preferencias del usuario



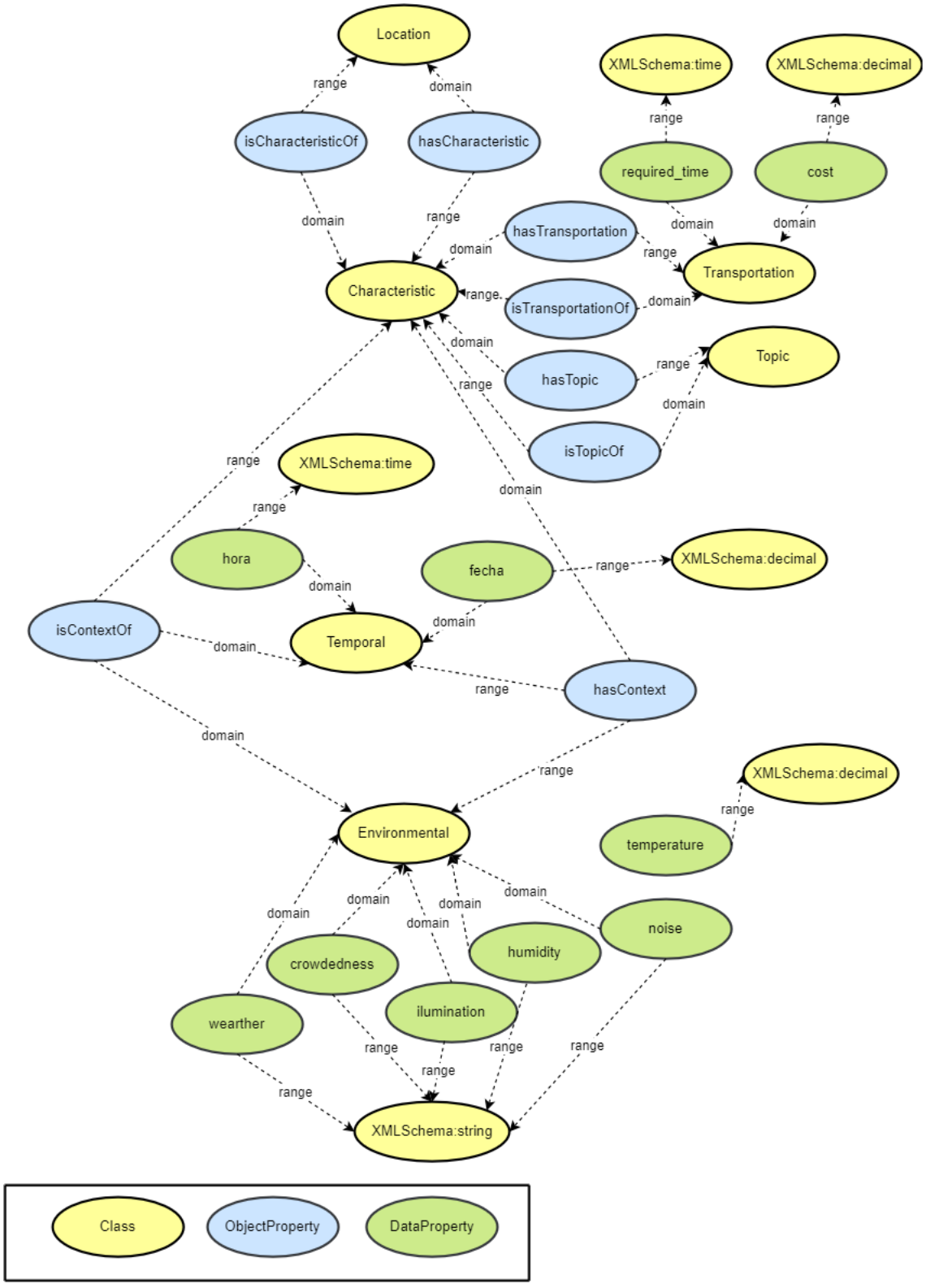
Fuente: El autor

Tabla 4-8: Descripción de los atributos en detalle del modelo de las preferencias del usuario

Clase	Atributo	Tipo	Cardinalidad
Transportation	Required_time	XMLSchema:time	1
	cost	XMLSchema:decimal	1
Temporal	Hora	XMLSchema:time	1
	fecha	XMLSchema:date	1
Environmental	weather	XMLSchema:string	1
	crowdedness	XMLSchema:string	1
	ilumination	XMLSchema:string	1
	humidity	XMLSchema:string	1
	noise	XMLSchema:string	1
	temperature	XMLSchema:string	1

Fuente: El autor

Figura 4-14: Descripción de los atributos del modelo de las características del lugar



Fuente: El autor

Tabla 4-9: Descripción de los atributos en detalle del modelo de las características del lugar

Clase	Atributo	Tipo	Cardinalidad
Transportation	Required_time	XMLSchema:time	1
	cost	XMLSchema:decimal	1
Temporal	Hora	XMLSchema:time	1
	fecha	XMLSchema:date	1
Environmental	weather	XMLSchema:string	1
	crowdedness	XMLSchema:string	1
	ilumination	XMLSchema:string	1
	humidity	XMLSchema:string	1
	noise	XMLSchema:string	1
	temperature	XMLSchema:string	1

Fuente: El autor

- Definición de reglas

  Para la definición de las reglas se utilizó el lenguaje SWRL por sus siglas en inglés (Semantic Web Rule Language) que es un lenguaje para definir reglas que se pueden inferir usando la ontología. Las reglas fueron escritas usando Owlready que es un lenguaje en python que permite crear programas orientados a una ontología, y luego ejecutarlos usando el razonador integrado Pellet. Las reglas fueron definidas para razonar sobre el conocimiento representado en los siguientes casos:

Tabla 4-10: Regla para buscar los lugares que tengan eventos en la fecha y hora del contexto temporal

Nombre: Regla de lugares según el contexto temporal
Regla:

```
rule1 = Imp()
rule1.set_as_rule("Temporal(?t),fecha(?t,?f),hora(?t,?h),"

"Location(?l),hasSchedule(?l,?s),date_schedule(?s,?f)," # la misma fecha
                  "opening_time(?s,?o),greaterThanOrEqual(?h,?o)," #
```

```
que ya este abierto
                    "closing_time(?s,?c),lessThanOrEqual(?h,?c) " #
que no este cerrado
                    "->
AdequateLocationTemporal(?l),AdequateSchedule(?s)")
```

Fuente: El autor

Tabla 4-11: Regla para buscar los lugares que cumplen con el contexto temporal y luego buscar que tengan el tópico del usuario con id: id

Nombre: Regla de lugares con el tópico de interés

Regla:

```
rule2 = Imp()
rule2.set_as_rule("User(?u),identity(?u," + idStr +
"),hasPreference(?u,?p),prefHasTopic(?p,?t),"

"AdequateLocationTemporal(?l),hasCharacteristic(?l,?c),hasTopic(?c,?
t)" # que tenga el mismo topico del usuario
                    "->AdequateLocationTopic(?l)")
```

Fuente: El autor

Tabla 4-12: Regla para buscar los lugares que correspondan con las preferencias genéricas del usuario: la hora de salida, de llegada, su presupuesto y su fecha propuesta para salir

Nombre: Regla para los lugares con las preferencias genéricas del usuario

Regla:

```
rule3 = Imp()
rule3.set_as_rule("Temporal(?t),fecha(?t,?f),"
                    "User(?u),identity(?u," + idStr +
"),hasGenericPreference(?u,?p),dateGenPref(?p,?f),"

"departure_time(?p,?d),arrival_time(?p,?a),budget(?p,?b),"

"AdequateLocationTopic(?l),hasSchedule(?l,?s),opening_time(?s,?o),"

"closing_time(?s,?c),cost(?s,?cst),greaterThan(?b,?cst)," # con el
presupuesto mayor que el costo
                    "greaterThanOrEqual(?d,?o),lessThanOrEqual(?a,?c)"
# que ya este abierto y que me alcance antes de que cierren
                    "->RecommendedUserLoc(?l)")
```

Fuente: El autor

Tabla 4-13: Regla para buscar los lugares que no tengan restricciones para menores de edad

Nombre: Reglas de lugares para menores de edad

Regla:
`rule4 = Imp()`
`rule4.set_as_rule("RecommendedUserLoc(?l),hasSchedule(?l,?s),underage(?s,false),"`
`"User(?u),identity(?u," + idStr + "),hasPreference(?u,?p),"`
`"hasEmotionalState(?p,normal_1),hasCompanion(?p,group_1),"`
`"typeOfTransportation(?p,private)"`
`"->AdequateLocationUnderAge(?l)")`

Fuente: El autor

- Definición de instancias

  Las definiciones de las instancias fueron creadas en la ontología tourist usando Owlready y se pueden ver en el anexo de las instancias.

4.2.2 Formalización e implementación de la ontología

A la representación del conocimiento del sistema propuesto se le dio el nombre de ontología tourist, y representa el conocimiento en detalle, para relacionarla con sus elementos definidos y razonar sobre este conocimiento, en este caso se utilizó Owlready y se crearon 2 clases para manejar el usuario, 3 clases para gestionar el historial del usuario. Se crearon 7 clases para gestionar los tipos de lugares turísticos. Se crearon 2 clases para gestionar la programación de los eventos de los lugares. Se crearon 2 clases para representar las preferencias del usuario y las características del lugar. Se crearon 8 clases para gestionar el estado emocional del usuario. Se crearon 4 clases para gestionar el perfil social del usuario. Se crearon 5 clases para gestionar las preferencias de transporte del usuario. Se crearon 13 clases para el tópico de Accomodation. Se crearon 12 clases para el tópico de Services. Se crearon 17 clases para el tópico de Transportation. Se crearon 9 clases para el tópico de Food. Se crearon 9 clases para el tópico de NationalCulture. Se crearon 9 clases para el tópico de Activity. Se crearon 40 clases para el tópico de Shopping. Se crearon 25 clases para el tópico de Entertaintment. Se crearon 21 clases para el tópico de NaturalLandscape. Se crearon 43 clases para el tópico de CulturalLandscape. Se crearon 5 clases para gestionar el contexto geográfico. Se crearon 2 clases para gestionar el contexto temporal. Se crearon 34 relaciones binarias entre todas las clases definidas. Se crearon 37 propiedades de tipo object-property y datatype-property. La implementación de la ontología la pueden ver en el anexo.

4.3 Sistemas de recomendación aplicados al modelo de recomendación para visitas guiadas, basado en computación ubicua y sensible al contexto

Un sistema de recomendación es una aplicación que indica objetos relevantes a un usuario, y dependiendo del dominio de la aplicación, el objeto podría referirse a una película, un producto, un restaurante, una ruta de viaje, etc. Pero lo que realmente los sistemas de recomendación intentan indicar o predecir es si un objeto recomendado será de valor para el usuario (Liang, 2022). En el modelo que se propone en este trabajo se implementan una serie de modelos de recomendación que arrojan recomendaciones, pero se van mejorando y avanzando hasta llegar a un modelo de recomendación híbrido que hace uso de alguno de los modelos previos para obtener mejores recomendaciones. En este capítulo veremos cada uno de los sistemas de recomendación abordados que hacen uso de los datos que fueron usados según el modelo desarrollado siguiendo la ontología tourist y que fueron implementados en pandas para poder aplicar las técnicas que se usan en el lenguaje de python para la construcción de los sistemas de recomendación adicionalmente hablaremos de las métricas empleadas para evaluar el desempeño de cada uno de estos sistemas.

4.3.1 Modelo de filtrado colaborativo

Para desarrollar un filtro colaborativo se necesitan los datos, la métrica de evaluación y una función que aplique dicha métrica a un modelo determinado. En este apartado se construye el modelo del filtro para visitas guiadas sensibles al contexto, y debemos considerar el modelo de la ontología *tourist* para modelar los datos necesarios. Los filtros colaborativos requieren datos sobre el usuario y su comportamiento, por esta razón necesitamos crear el conjunto de datos adecuado.

Para modelar los datos de los lugares según el modelo de la ontología *tourist*, se definen los siguientes archivos:

- locations.csv: Este archivo contiene los siguientes campos; id_location, name_location, address, image, char_id, sum_scores, vote_count, y overview. Con estos datos se representa un lugar y se hacen los cálculos relacionados en el modelo.
- schedules.csv: Este archivo se usa para representar los horarios e información de los eventos y actividades que se realicen en un lugar y tiene los siguientes datos: char_id, sche_id, date_schedule, opening_time, closing_time, event, cost, y underage.
- topicCharacteristics.csv: Este archivo se usa para indicar el tópico de un lugar y tiene los siguientes datos: char_id, hasTopic.
- transportationCharacteristics.csv: Este archivo se usa para indicar las formas de llegar a un lugar y contiene información sobre el tiempo que se toma para llegar y el costo de usar dicho medio de transporte. Tiene los siguientes datos: char_id, transp_id, required_time, y cost_transp.
- environmentalCharacteristics.csv: Este archivo contiene los datos para representar el contexto ambiental, geográfico y temporal del lugar y tiene los siguientes datos: char_id, environmental_id, weather, temperature, crowdedness, illumination, humidity, noise, geographical_id, inside_id, temporal_id, hora y fecha.

Para modelar los datos de los usuarios según el modelo de la ontología *tourist*, se definen los siguientes archivos:

- users.csv: Este archivo contiene la información del usuario: user_id, identity, current_location, name_user, gender, age, education, e income_level.
- transportationPreferences.csv: Este archivo se usa para indicar las formas preferidas de llegar a un lugar y contiene información sobre el tiempo que el usuario está dispuesto a gastar para llegar y el presupuesto que está dispuesto a usar por dicho medio de transporte. Tiene los siguientes datos: pref_id, transp_id, required_time, y cost_transp.
- topicPreferences.csv: Este archivo se usa para indicar el tópico de interés de un lugar por visitar, un usuario puede tener muchos tópicos de interés y tiene los siguientes datos: pref_id, prefHasTopic.
- ratings.csv: Este archivo contiene la calificación y la reseña de un usuario para un lugar, un usuario puede calificar muchos lugares. Tiene los siguientes datos: user_id, location_id, rating y review.

- preferences.csv: Este archivo se usa para representar el estado emocional y la forma de relacionarse socialmente en un viaje. Tiene los siguientes datos: id_user, pref_id, hasEmotionalState y hasCompanion.
- GenericPreference.csv: Este archivo se usa para representar las preferencias de un itinerario por día de un usuario y tiene los siguientes datos: gen_pref_id, departure_time, Budget, arrival_time, dateGenPref, y id_user.
- environmentalPreferences.csv: Este archivo contiene los datos para representar el contexto ambiental, geográfico y temporal del usuario y tiene los siguientes datos: pref_id, environmental_id, weather, temperature, crowdedness, illumination, humidity, noise, geographical_id, temporal_id, hora y fecha.

Para los modelos presentados en este capítulo, se usaron 50 usuarios, 100 lugares y 300 evaluaciones y reseñas de dichos usuarios para dichos lugares. Veamos los 5 primeros registros de los usuarios en Pandas DataFrame, ver figura 4-5:

Figura 4-15: Ejemplo de los 5 primeros registros de los usuarios.

	user_id	identity	current_location	name_user	gender	age	education	income_level
0	1	10008	“”	Maria	female	27	master	lower-middle
1	2	10009	“”	Juan	male	30	master	lower-middle
2	3	10010	“”	Adriana	female	22	upper secondary education	lower-middle
3	4	10011	“”	Angela	female	40	bachelor	high
4	5	10012	“”	Jose Luis	male	8	primary education	lower-middle

Fuente: El autor

Podemos ver que los registros contienen información demográfica de los usuarios. Ahora vemos el contenido del archivo de los lugares, el cual nos da información sobre los lugares que han sido calificados y reseñados por los usuarios, ver figura 4-6:

Figura 4-16: Ejemplo de los 2 primeros registros de los lugares.

```
0 id_location                                                  1
name_location                 El correo licor  y comidas 24 horas
address                  Cra 52 #3 Sur - 85, Medellín, El poblado
image                                                   urlimage
char_id                                                        1
sum_scores                                                   101
vote_count                                                    30
overview         Gran restaurante para cenar en una zona muy tr...
Name: 0, dtype: object
   ---

1 id_location                                                  2
name_location                              JPMorgan FIX Tattoo
address          Cra. 51 # 48 - 40 local 107, Itagui, Antioquia
image                                                   urlimage
char_id                                                        2
sum_scores                                                    94
vote_count                                                    39
overview              Tienda de tatuajes y piercings en Itagüi.
Name: 1, dtype: object
   ---
```

Fuente: El autor

Podemos ver que este archivo nos da información concerniente a los lugares. Como solo se pretende construir un filtro colaborativo, de estos datos solo necesitamos el nombre y el id que identifica el lugar. Ahora vemos algunos de los registros del archivo ratings.csv que nos muestra las calificaciones y reseñas de los lugares visitados por los usuarios ver figura 4-7. Esta figura nos muestra que por cada registro podemos ver el rating y el review dado por un usuario a un lugar particular.

Figura 4-17: Ejemplo de los 3 primeros registros de los ratings de los usuarios.

```
0 user_id                                                    1
location_id                                                  1
rating                                                       4
review         Excelente servicio. Los mejores licores de la ...
Name: 0, dtype: object
   ---

1 user_id                                                    1
location_id                                                  2
rating                                                       4
review         Muy buen trabajo, muy importante todo muy limp...
Name: 1, dtype: object
   ---

2 user_id                                                    2
location_id                                                 30
rating                                                       1
review         Resultados manipulados con computadora. Ningún...
Name: 2, dtype: object
   ---
```

Fuente: El autor

El DataFrame de los ratings contiene información de los ratings de los usuarios para cada lugar visitado que varían de 1 a 5. Por lo cual, este problema se modeló como un caso de aprendizaje supervisado donde es necesario predecir el rating, dado un usuario y un lugar, debido a esta circunstancia el problema se modeló como un problema de regresión.

Para construir un modelo de aprendizaje supervisado necesitamos construir un conjunto de datos de prueba y otro de entrenamiento. El modelo aprende usando los datos de entrenamiento y su potencia será evaluada usando los datos de prueba. Para tal fin se dividieron los datos de los ratings en 75% para entrenamiento y 25% para prueba.

Para obtener los conjuntos de datos usamos la función train_test_split de la librería scikit-learn, con los datos del archivo ratings.csc como variables predictoras y únicamente la columna user_id como la variable objetivo.

La métrica que se utilizó para medir el desempeño del modelo supervisado debido a que es un problema de regresión fue el error cuadrático medio RMSE (por sus siglas en inglés), está métrica se usó para evaluar el desempeño de los modelos de sistemas de

recomendación implementados. En la sección de evaluación de los modelos propuestos se mostrarán los resultados obtenidos de este modelo.

4.3.2 Modelo de filtrado colaborativo basado en la calificación del usuario

Los filtros basados en el usuario buscan usuarios similares a un usuario particular para recomendarle los lugares visitados que a ellos les gustaron. Para medir el desempeño de este modelo se usaron los mismos elementos que se usaron en el modelo de filtrado colaborativo.

Primero necesitamos construir una matriz de puntajes donde cada fila representa a un usuario y cada columna representa a un lugar. Por lo tanto, el valor en la entrada ij^{th} para la fila i y columna j representa un puntaje que el usuario i asignó a un lugar j. Se usó una función de pandas, llamado pivot_table, para construir esta matriz usando los datos del archivo ratings.csv ya mencionado al comienzo del capítulo.

Figura 4-18: Matriz de puntajes

location_id	1	2	3	4	8	9	...	81	85	87	89	90	91
user_id							...						
1	NaN	4.0	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN
2	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN
3	NaN	NaN	4.0	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN
4	NaN	NaN	NaN	NaN	2.0	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN
5	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN
6	NaN	NaN	NaN	NaN	NaN	NaN	...	2.0	4.0	1.0	NaN	3.0	NaN

Fuente: El autor

La figura 4-8, nos muestra como es la forma de la matriz para los 6 primeros usuarios. Vemos que el usuario 6 le dio un puntaje de 2 al lugar 81, de 4 al lugar 85, de 1 al lugar 87, y de 3 al lugar 90. Para los demás lugares no hay puntuación ya que el usuario no los ha visitado o no le interesa hacerlo. Además, la matriz tiene muchos lugares sin calificar con NaN, esto indica que la matriz es muy dispersa.

Vamos primero a trabajar con todos los usuarios y lugares, para calcular la media de los puntajes dados a cada lugar por parte de todos los usuarios que lo han visitado y dado un puntaje. No se hace ninguna distinción entre los usuarios, es decir que se les da igual peso a los puntajes de todos los usuarios. Mirar en los anexos la función cf_user_mean (user_id, locaton_id).

Ahora vamos a trabajar con la media ponderada debido a que no le da mayor preferencia a los usuarios que le dieron la misma calificación a los lugares que la que le daría el usuario de interés. Para esto se usó como función de similitud, la distancia del coseno. En este caso fue necesario construir una matriz de similitud del coseno para los usuarios del modelo. Está matriz nos da los puntajes escalados entre 1 y 5. En los anexos se puede ver una explicación de cómo funciona la distancia del coseno.

Figura 4-19: Matriz de similitud del coseno para los usuarios.

user_id	1	2	3	4	...	16	17	18	19
user_id					...				
1	1.0	0.0	0.000000	0.000000	...	0.000000	0.000000	0.000000	0.000000
2	0.0	1.0	0.000000	0.000000	...	0.000000	0.000000	0.624695	0.000000
3	0.0	0.0	1.000000	0.000000	...	0.000000	0.000000	0.000000	0.000000
4	0.0	0.0	0.000000	1.000000	...	0.000000	0.000000	0.000000	0.000000
5	0.0	0.0	0.000000	0.000000	...	0.000000	0.119523	0.000000	0.000000
6	0.0	0.0	0.000000	0.126773	...	0.000000	0.000000	0.000000	0.133892
7	0.0	0.0	0.000000	0.000000	...	0.305424	0.000000	0.269827	0.036290
8	0.0	0.0	0.486664	0.000000	...	0.000000	0.000000	0.076004	0.000000
9	0.0	0.0	0.000000	0.000000	...	0.000000	0.069007	0.000000	0.000000
10	0.0	0.0	0.000000	0.000000	...	0.000000	0.000000	0.000000	0.000000

Fuente: El autor

Con la matriz de similitud de distancias del coseno, podemos calcular los puntajes con la media ponderada, en los anexos pueden ver la implementación del código, solo demos tener en cuenta que solo se consideran los puntajes de distancia del coseno que no sean de valor igual a 0.0, es decir que se debe evitar a los usuarios que non han visitado ni calificado un lugar determinado.

4.3.3 Modelo de filtrado colaborativo usando datos demográficos del usuario

Los filtros pueden aprovechar la información demográfica del usuario. La idea básica es que en estos filtros se intenta aprovechar a los usuarios que tienen los mismos datos demográficos también suelen tener los mismos gustos para viajar a los mismos lugares. Una persona del mismo sexo, un adolescente o una persona de la misma zona pueden tener gustos parecidos por viajar a los mismos lugares.

Este modelo de filtrado no considera los puntajes dados por los usuarios a un lugar determinado. Sino, que en estos modelos se buscan a los usuarios que se ajustan a ciertas características demográficas. Este modelo que se presenta en esta sección es un filtro que identifica el género de un usuario, calcula la media ponderada de sus puntajes para un lugar determinado para dicho género y muestra como resultado dicho valor calculado. En este modelo se utilizó el archivo users.csc para poder construir el filtro.

En el anexo se puede ver como se obtuvo la media del puntaje de cada lugar por género. Y la función para identificar el género del usuario, y extraer el promedio de los puntajes dados a un lugar para dicho genero particular, y retorna dicho valor como resultado. Finalmente, se construye un modelo de recomendación usando el género y el nivel educativo. Y se puede observar que las métricas mejoran el nivel de desempeño de la medición, pero no superan las del modelo inicial de filtrado colaborativo básico. El código lo pueden encontrar en los anexos.

4.3.4 Modelo de filtrado colaborativo basado en clustering

Este tipo de modelado permite contar con una forma de agrupar a los usuarios con una métrica mucho más robusta que la de los datos demográficos. Es posible utilizar algoritmos de clustering, tales como k-means, para agrupar a los usuarios en un clúster y luego trabajar con los usuarios de un mismo clúster para hacer predicciones de puntajes de los sitios de interés.

En esta sección vamos a usar un algoritmo muy parecido al k-means, el KNN, para construir el filtro colaborativo basado en clustering, de la siguiente forma, dado un usuario u, y un lugar l, se siguen los siguientes pasos:

- Encontrar lo vecinos k más cercanos al usuario u, y que ya hayan visitado y calificado el lugar l.
- Generar el puntaje promedio de los k usuarios que ya hayan visitado y calificado el lugar l.

Para implementar este modelo se usó la librería *surprise*, del paquete scikit usada para construir sistemas de recomendación en python.

En los anexos se puede ver como se construyó y evaluó el modelo del filtro colaborativo basado en KNN. En este modelo se aplicó una técnica de validación cruzada de 5 pliegues y se identificaron para cada pliegue las métricas RMSE, precision, recall y F1-score. Y que finalmente nos sirven para buscar la mejor para usar en el conjunto de pruebas.

4.3.5 Modelo de filtrado colaborativo usando descomposición en valores singulares

Si miramos la matriz de rankings, tiene la forma de m x n, donde las filas representan m usuarios y las columnas representan uno n lugares. Si eliminamos alguna de las columnas, a hora nuestra matriz tendrá menor tamaño, y si consideramos esta nueva matriz como las variables predictoras y la columna que se eliminó como la columna objetivo, se pueden usar algoritmos supervisados para entrenar los valores que estén en la columna eliminada y predecir los valores que no existan en esta misma columna. Este proceso puede ser repetido n veces por cada columna hasta llegar a completar toda la matriz. Pero los datos faltantes son un problema y en nuestra matriz es una matriz dispersa con más del 90% de los datos faltantes. Para solucionar este problema se usa la técnica de descomposición en valores singulares que permite obtener la matriz original factorizada en otras dos matrices.

De la siguiente forma: $A = UV^T$

Donde la matriz U, son los componentes principales del usuario, y la matriz V es la matriz de los lugares. La matriz U se conoce como la matriz *user-embedding*. Y la matriz V como la matriz *place-embeding*. Estas dos matrices son densas, y nos dan las predicciones de todos los valores faltantes en la matriz original A.

En el anexo se puede ver como se implementa es filtro SVD usando la librería *surprise*.

4.3.6 Modelo de filtrado colaborativo basado en sistemas de recomendación híbridos

Los sistemas de recomendación híbridos son sistemas que combinan varios modelos para solucionar los problemas y deficiencias que cada uno presenta individualmente con el uso de las fortalezas de los otros. En esta sección se presenta un modelo de recomendación híbrido que combina los filtros basados en contenido y los filtros colaborativos.

Primero se construye un filtro colaborativo que hace predicciones de los puntajes de los lugares recomendados por otro modelo de filtrado basado en contenido y entrega como resultado los mejores lugares con las predicciones de los puntajes más altos.

Para realizar el modelo de filtrado híbrido se siguieron estos pasos:

- Seleccionar el nombre de un lugar y a un usuario.
- Usar un modelo de filtrado basado en contenido que calcule los 25 lugares más similares a lugar seleccionado.
- Calcular los puntajes que el usuario seleccionado podría darle a estos 25 lugares usando un filtrado colaborativo.
- Mostrar los 10 mejores lugares con los puntajes más altos que han sido predichos por el filtrado colaborativo.

Se usaron los mismos archivos con los datos que se indicaron al comienzo. Se requirió para el filtro colaborativo hacer la matriz de similitudes de la distancia del coseno, de esta matriz se requirió mapear cada índice de cada lugar del conjunto de datos existente. Para este modelo se utilizó un filtrado colaborativo usando la descomposición en valores singulares SVD. Del conjunto de datos de los lugares se necesitó construir una función que buscara el nombre de un lugar indicándole su id, y otra función que realizara lo inverso.

En el anexo se puede ver el código para construir la función para el filtrado híbrido con los pasos anteriores.

4.4 Análisis de sentimientos aplicado al modelo de recomendación para visitas guiadas, basado en computación ubicua y sensible al contexto

El análisis de sentimientos es una aplicación de aprendizaje de máquinas que permite clasificar documentos según su contenido. La aplicación de esta técnica en esta sección necesita acceder a ciertas características de un lugar que permitan determinar si dicha característica se encuentra presente o ausente en el lugar.

Como se trata de documentos, los lugares deben exhibir algún tipo de texto para poder trabajar sobre las características de las palabras que se encuentran en dicho texto. Al usar palabras como características, se parte de la idea de que algunas palabras tienen mayor probabilidad de aparecer, en este caso estamos hablando de un texto con el que vamos a trabajar en esta sección, se trata de los review o reseñas que un usuario ha realizado a algún lugar visitado, para esto se usará el archivo ratings.csv que tiene las puntuaciones y reseñas realizadas por los usuarios para cada lugar visitado.

Un texto necesita ser representado en forma de cantidades matemáticas, esto se logra representando los textos como vectores. Esta técnica necesita usar una función conocida como *vectorizer* para convertir los textos en un vector. En esta sección se usó la función TF-IDFVectorizer que significa, Term Frequency-Inverse Document Frequency, (por sus siglas en inglés), y considera el peso de todas las palabras presentes en un texto y no se centra únicamente en las más repetidas. Esta función asigna un peso a cada palabra presente en el texto según la siguiente formula. Para toda i en el texto j, aplica lo siguiente:

$$w_{i,j} = tf_{i,j}\, x\, \log(\frac{N}{df_i})$$

- En la formula anterior se cumple lo siguiente:
- $w_{i,j}$ es el peso de la palabra i en el documento j
- df_i es el número de documentos que contienen el termino i
- N es el total de documentos

En esta fórmula el peso de una palabra en un documento es más grande si esta palabra aparece más frecuentemente en ese documento y esta menos presente en otros documentos. El peso $w_{i,j}$ toma valores entre 0 y 1.

TF-IDFVectorizer se aplica en este modelo debido a que algunas palabras aparecen con mucha más frecuencia en algunas reseñas. Es mejor asignar pesos a cada palabra en un texto según la fórmula de esta función.

Veamos un ejemplo de algunas reseñas de 5 lugares que han sido visitadas.

Figura 4-20: Ejemplo de las 5 primeras reseñas a lugares visitados

```
0    Excelente servicio. Los mejores licores de la ...
1    Muy buen trabajo, muy importante todo muy limp...
2    Resultados manipulados con computadora. Ningún...
3    El museo es gigantesco. Si te gustan este tipo...
4    Para los amantes de los museos, es una opción ...
```

Fuente: El autor

Para cada uno de estos textos, es decir las reseñas, vamos a calcular una nueva característica que nos indica en una escala de 1 a 5, el grado de aceptación del lugar. Primero vamos definir los conjuntos de entrenamiento y pruebas. Luego definimos el vector TD-IDFVectorizer y removemos las palabras más comunes que no aportan valor. Y entrenamos el modelo.

Veamos un ejemplo del modelo, vamos a determinar la frecuencia de las palabras de un texto y ver su score IDF:

```
ejemplo = 'excelente lugar para realizar eventos como convenciones, es
muy amplio y cuenta con una plaza donde puedes encontrar restaurantes y
tiendas.'
```

Figura 4-21: Ejemplo de las frecuencias de las palabras en un texto de ejemplo

Palabra	IDF
excelente	3.067
lugar	2.425
para	00V
realizar	00V
eventos	00V
como	00V
convenciones	00V
,	00V
es	00V
muy	00V
amplio	00V
y	00V
cuenta	00V
con	00V
una	00V
plaza	00V
donde	00V
puedes	3.760
encontrar	00V
restaurantes	00V
y	00V
tiendas	00V
.	00V

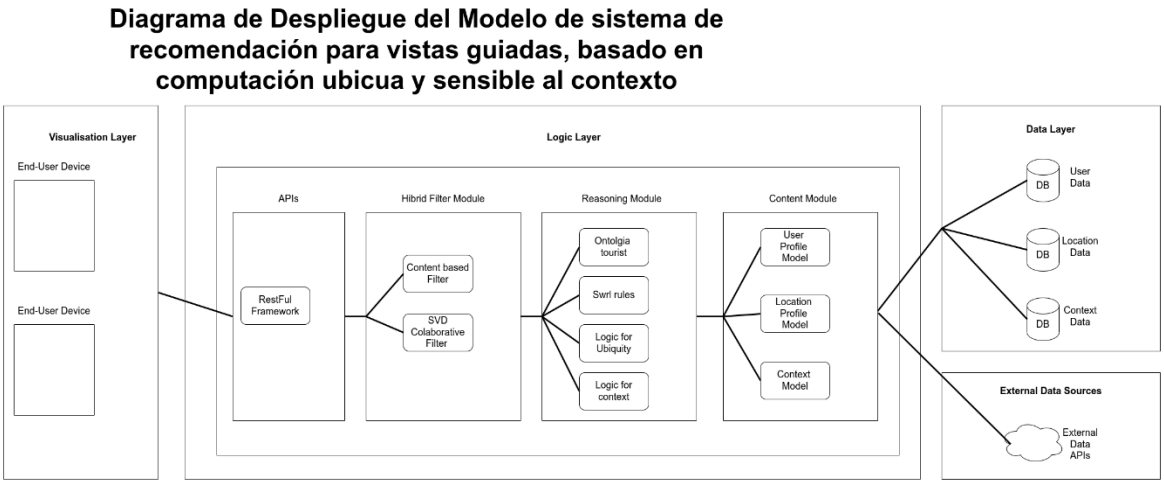
Fuente: El autor

Ya tenemos preprocesadas todas las palabras de los reviews realizados por los usuarios, y con el modelo ya creado se puede hacer el entrenamiento y realizar la correspondiente evaluación y ver los resultados de sus métricas. El código para generar este modelo se puede ver en los anexos.

4.5 Diseño de la arquitectura

A continuación se muestra el diseño de la arquitectura del modelo y sus elementos que lo conforman.

Figura 4-22: Diagrama de despliegue del modelo



4.6 Conclusiones del capitulo

En este capitulo, se presentó la caracterización de las entidades y conceptos que conforman el modelo del sistema de recomendación para visitas guiadas para el modelo del usuario y su personalización, para el modelo del lugar y su adaptación y para el modelo del contexto.

Para la ontología tourist, se presentó su conceptualización por medio de la elaboración de sus componentes y su formalización e implementación en python usando la librería Owlready.

Este capítulo permitió demostrar la ventaja de integrar los elementos del modelo propuesto como el contexto, los lugares y el usuario en una ontología que facilitó la integración posterior a los modelos estudiados de sistemas de recomendación y a la técnica de aprendizaje automático de análisis de sentimientos.

También se mostraron aplicaciones del modelo propuesto a distintos modelos de recomendación de filtrado colaborativo y para esto se demostró que se pudo integrar la información del modelo para ser manipulada en python con la librería pandas. Los modelos de recomendación de filtrado colaborativo con los que se trabajó fueron, basados en la

calificación del usuario, usando datos demográficos del usuario, basado en clustering, usando descomposición en valores singulares, y basado en sistemas de recomendación híbridos. Se utilizó la técnica de análisis de sentimientos usando la información representada en el modelo; las características de los lugares visitados y las evaluaciones de los usuarios.

En el siguiente capítulo, se verá cómo se realizaron cada una de las aplicaciones del modelo a los diferentes modelos de recomendación de filtrado colaborativo y al modelo de análisis de sentimientos, así como la interpretación de los resultados obtenidos con sus métricas de desempeño.

5. Evaluación de los modelos propuestos a través de métricas

Esta sección presenta la evaluación del desempeño de cada uno de los modelos de sistemas de recomendación que se desarrollaron aplicando el modelo propuesto y resaltando sus características presentadas. Para analizar los modelos de recomendación y medir que tan efectivos son para dar sugerencias relevantes a los usuarios, se usaron las siguientes métricas Precision@k y Recall@k.

La validación se aplica a los siguientes modelos:

- Filtrado colaborativo.
- Filtrado colaborativo basado en el usuario.
- Filtrado colaborativo usando datos demográficos del usuario.
- Filtrado colaborativo basado en Clustering.
- Filtrado colaborativo basado en reducción de dimensionalidad usando descomposición en valores singulares.
- Filtrado colaborativo basado en sistemas de recomendación híbridos.
- Análisis de sentimientos aplicado al modelo propuesto.

Para evaluar los modelos, vamos a usar estas dos métricas estándar, el error cuadrático medio (RMSE) y el error absoluto medio (MAE) (Hodson, 2022).

Para una muestra de n observaciones $y(y_i \; i = 1,2, \dots, n)$

Y n correspondiendo a la predicción del modelo $\hat{y}$ el MAE y el RMSE son:

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{n}(y_{i-\hat{y}_i})^2}{n}} \quad , \qquad (1)$$

$$\mathrm{MAE} = \frac{1}{\mathrm{n}} \sum_{i=1}^{n} |y_i - \hat{y}_i| \quad , \qquad (2)$$

Como la librería scikit-learn nos brinda una implementación para calcular el error cuadrático medio, solo se tuvo que hacer una función que retornará la raíz cuadrada del resultado de llamar esta función, mean_squared-error. En los anexos mirar el código implementado.

En lugar de mirar a todo el conjunto posible de recomendaciones, con las métricas Precision@k y Recall@k solo se consideran los k mejores lugares para visitar. Debemos aclarar que los lugares recomendados son los lugares que cada modelo sugiere a un usuario particular. Y un lugar relevante es un lugar que fue realmente seleccionado por un usuario particular. El termino @k, significa los k mejores, siendo esta k el número de lugares que nos interesan.

Ahora definamos Precision@k y Recall@k:

$$\mathrm{Precision@k} = \frac{\text{número de lugares recomendados que son relevantes @k}}{\text{número de lugares recomendados @k}} \qquad (1)$$

$$\mathrm{Recall@k} = \frac{\text{número de lugares recomendados que son relevantes @k}}{\text{número de todos los lugares relevantes}} \qquad (2)$$

La Precision@k y el Recall@k son métricas binarias, y solo pueden ser calculadas para lugares que un usuario ya halla realmente calificado. La Precision@k mide que porcentaje de recomendaciones fueron realmente seleccionadas por el usuario. Y, por el contrario, Recall@k mide la fracción de todos los lugares relevantes que fueron sugeridos por el modelo.

Un lugar es considerado relevante si su evaluación real r_{ui} es más grande que un umbral determinado. Un lugar es considerado recomendado si su evaluación estimada $\hat{r}_{ui}$ es mayor que el umbral, y si esta está entre los k rankings estimados más altos.

Adicionalmente, se considerará la métrica F1-score para decidir en conjunto con la RMSE, que modelo es mejor. El F1-score calcula la armónica media de la Precision@k y el Recall@k, de la siguiente forma:

$$\mathrm{F1-score} = 2 * \frac{\mathrm{Precision@k} * \mathrm{Recall@k}}{\mathrm{Precision@k} * \mathrm{Recall@k}}$$

En los anexos se puede ver el código para calcular estás métricas, ver anexo. Para el modelo de recomendación híbrido, se aplicó un método de validación cruzada de 5 pliegues y se identificaron las puntuaciones F1-score para cada pliegue, de las puntuaciones obtenidas se buscó la media de estos 5 pliegues y se eligió como la mejor la del promedio F1-score más alto.

5.1 Filtrado colaborativo

Después de construir el modelo de aprendizaje y usar los conjuntos de datos de prueba y entrenamiento. Vamos a evaluar la potencia usando los datos de prueba. Al calcular el RMSE, se obtiene para nuestro modelo de referencia la siguiente puntuación:

Figura 5-1: Evaluación del filtrado colaborativo modelo de referencia

```
The RMSE score obtained on the testing set by the model
Precision@k: 0.90, Recall@k: 0.44, F1 Score@k: 0.59 for k=20
RMSE: 1.2765578717786357
```

Fuente: El autor

De la figura podemos interpretar que, para un k de 20, se obtuvo un F1-score de 0.59 y un RMSE de 1.27. Estos resultados nos indican la relación existente entre los lugares, y sus puntuaciones con respecto los usuarios. Este resultado nos muestra la distancia promedio entre los valores predichos y los valores reales en el conjunto de datos. En este modelo se usan los lugares ya evaluados y sus puntajes para predecir un usuario particular qué puntaje asignarían a dicho lugar sino lo ha visitado en realidad. Mientras más bajo sea el valor del RMSE, mucho mejor es el modelo para ajustar los datos del modelo. En los siguientes modelos se trata de conseguir un RMSE mucho menor que el obtenido con filtrado colaborativo como modelo inicial.

5.2 Filtrado colaborativo basado en el usuario

Primero vamos a calcular las métricas de RMSE y F1-score para el modelo que filtra la media de los puntajes de todos los usuarios para un solo lugar. Y a continuación para el modelo que filtra la media ponderada de los puntajes de todos los usuarios para un solo lugar. El resultado se muestra en la siguiente figura 5-2. Primero para la media y luego tenemos los resultados de la media ponderada.

Figura 5-2: Evaluación del filtrado colaborativo basado en el usuario usando las medias

```
RMSE for the Mean model
Precision@k: 0.85, Recall@k: 0.41, F1 Score@k: 0.56 for k=20
RMSE: 1.8311995158001397

Using Weighted Mean Ratings
Precision@k: 0.85, Recall@k: 0.41, F1 Score@k: 0.56 for k=20
RMSE: 1.712421524717329
```

Fuente: El autor

Podemos ver que los puntajes de los F1-scores son más bajos que el obtenido en el filtrado colaborativo que fue de F1-score: 0.59, lo cual indica que los RMSE son más altos que el obtenido en el filtrado colaborativo que fue de 1.27. Estos modelos no ajustan mejor los datos que el modelo anterior, se debe seguir buscando con otros modelos de recomendación que si mejoren dicha capacidad de dar mejores recomendaciones al usuario. En conclusión, con este modelo basado en el usuario usando las medias no se mejora los puntajes del RMSE.

5.3 Filtrado colaborativo usando datos demográficos del usuario

Resultados obtenidos usando el filtrado con datos demográficos del usuario, usando primero el género y luego el género y el nivel educativo.

Figura 5-3: Resultados del F1-score y RMSE usando datos demográficos del usuario

```
Using Mean Ratings for gender
Precision@k: 0.80, Recall@k: 0.39, F1 Score@k: 0.52 for k=20
RMSE: 1.6591032551083464

Using Mean Ratings for Gender and Education Based Collaborati
Precision@k: 0.90, Recall@k: 0.44, F1 Score@k: 0.59 for k=20
RMSE: 1.353957163281025
```

Fuente: El autor

Podemos ver que este modelo tiene un mejor desempeño en el RMSE cuando se considera el género y el nivel educativo, pero no supera el desempeño del modelo de filtrado colaborativo básico, es decir que este modelo inicial da mejores recomendaciones que el enfoque con datos demográficos.

Estos resultados igualmente sugieren que trabajar con datos demográficos del usuario no es la mejor forma de mejorar las predicciones de los sistemas de recomendación y es mejor usar técnicas de predicción que utilicen datos de rankings y puntajes. Una de las mayores desventajas de estos filtros es que se basan en la idea de que los usuarios con un mismo valor demográfico piensan y evalúan los productos de la misma manera.

5.4 Filtrado colaborativo basado en clustering

Este filtrado se apoya en la agrupación de los usuarios en varios clústeres para luego trabajar con los usuarios de un mismo clúster y para hacer predicciones de sus puntajes a los sitios de interés.

Veamos los resultados que nos arrojó la técnica de validación cruzada de 5 pliegues para identificar las métricas RMSE, precision, recall y F1-score.

Figura 5-4: Resultado de la métrica RMSE usando clúster KNN para 5 pliegues

```
Evaluating RMSE of algorithm KNNBasic on 5 split(s).

                  Fold 1  Fold 2  Fold 3  Fold 4  Fold 5  Mean    Std
RMSE (testset)    1.3444  1.5993  1.2814  1.6277  1.4647  1.4635  0.1362
Fit time          0.00    0.00    0.00    0.00    0.00    0.00    0.00
Test time         0.00    0.00    0.00    0.00    0.00    0.00    0.00
```

Fuente: El autor

Figura 5-5: Métricas de desempeño usando clúster KNN para 5 pliegues

```
              Fold 1    Fold 2    Fold 3    Fold 4    Fold 5
Precision:  0.850000  0.847059  0.926316  0.880952  0.819048
Recall:     0.855556  0.808123  0.914474  0.926190  0.824405
F1 Score:   0.852769  0.827133  0.920357  0.903005  0.821717
```

Fuente: El autor

El resultado nos indica que los datos están divididos en 5 partes iguales. Esta técnica usa 4 partes para el entrenamiento y para la prueba usa la parte número 5. Esto se hace 5 veces, de tal forma que cada parte hace las veces de datos de prueba una vez. Para este modelo vemos que la media de RMSE es de 1.4635, la cual no supera la obtenida en el filtrado colaborativo del modelo inicial básico la cual fue de 1.2765.

5.5 Filtrado colaborativo basado en reducción de dimensionalidad usando descomposición en valores singulares.

El resultado de implementar el filtrado SVD se puede ver en la siguiente figura.

Figura 5-6: Resultado de implementar el modelo en descomposición en valores singulares

```
Evaluating RMSE of algorithm SVD on 5 split(s).

                  Fold 1  Fold 2  Fold 3  Fold 4  Fold 5  Mean    Std
RMSE (testset)    1.3694  1.2115  1.1715  1.0960  1.3965  1.2490  0.1158
Fit time          0.01    0.01    0.01    0.01    0.01    0.01    0.00
Test time         0.00    0.00    0.00    0.00    0.00    0.00    0.00

              Fold 1    Fold 2    Fold 3    Fold 4    Fold 5
Precision:  0.779487  0.947619  0.774444  0.893750  0.940000
Recall:     0.805128  0.969388  0.916667  0.914583  1.000000
F1 Score:   0.792100  0.958380  0.839575  0.904047  0.969072
```

Fuente: El Autor

Para este modelo vemos que la media de RMSE es de 1.2490, la cual supera la obtenida en el filtrado colaborativo del modelo inicial básico la cual fue de 1.2765. Y se puede concluir que con este modelo se pueden obtener mejores recomendaciones que con todos los modelos vistos previamente.

5.6 Filtrado colaborativo basado en sistemas de recomendación híbridos

Para evaluar el modelo se usaron estos dos casos de estudio, y se obtuvieron los resultados correspondientes que a continuación se muestran.

```
mostrar(hybrid(6, 'California Club'))
```

```
mostrar(hybrid(49, 'Cedar Point'))
```

En la figura 5-7, se presentan los resultados de aplicar la técnica de validación cruzada con 5 pliegues para el modelo híbrido compuesto de un sistema de recomendación basado en contenido y otro colaborativo usando descomposición en valores singulares, obtuvimos como media para RMSE un valor de 1.2478, que supera la obtenida en el modelo que usa la descomposición en valores singulares con un RMSE de 1.2490, y la cual había sido la mejor hasta el momento.

Además, en el pliegue número 3 se obtuvo un F1-score de 0.96, el mejor de todos y que corresponde a un RMSE de 1.1936, de esta forma se puede concluir que el modelo de filtrado basado en sistemas de recomendación híbridos supera en desempeño a todos los modelos vistos previamente y por esta razón es uno de los más usados en la industria. Los resultados de las recomendaciones de este modelo se pueden ver en el anexo.

Figura 5-7: Resultados al aplicar el modelo híbrido usando la técnica de validación cruzada con 5 pliegues

```
Evaluating RMSE, MAE, MSE, FCP of algorithm SVD on 5 split(s).

                  Fold 1  Fold 2  Fold 3  Fold 4  Fold 5  Mean    Std
RMSE (testset)    1.2616  1.2970  1.1936  1.3086  1.1783  1.2478  0.0531
MAE (testset)     1.0295  1.0605  1.0078  1.1129  0.9279  1.0277  0.0612
MSE (testset)     1.5916  1.6821  1.4246  1.7125  1.3884  1.5598  0.1318
FCP (testset)     0.4912  0.4667  0.3883  0.5238  0.5576  0.4855  0.0574
Fit time          0.01    0.01    0.01    0.01    0.01    0.01    0.00
Test time         0.00    0.00    0.00    0.00    0.00    0.00    0.00

               Fold 1    Fold 2    Fold 3    Fold 4    Fold 5
Precision:  0.815556  0.844792  0.947619  0.904762  0.816667
Recall:     0.970000  0.916667  0.985714  0.908163  0.760417
F1 Score:   0.886098  0.879263  0.966291  0.906459  0.787539
```

Fuente: El autor

5.7 Análisis de sentimientos aplicado al modelo propuesto

El modelo se somete a un proceso de entrenamiento con el fin de obtener los parámetros idóneos que permitan obtener la mejor precisión, de esto se obtienen los siguientes resultados:

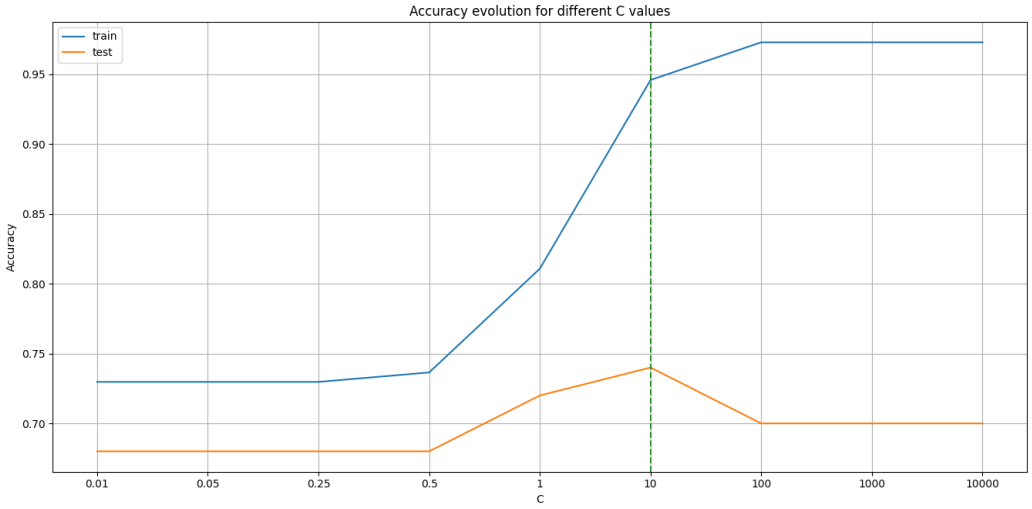
Figura 5-8: Ajuste del Accuracy del modelo

```
Accuracy for C = 0.01: 0.68
Accuracy for C = 0.05: 0.68
Accuracy for C = 0.25: 0.68
Accuracy for C = 0.5: 0.68
Accuracy for C = 1: 0.72
Accuracy for C = 10: 0.74
Accuracy for C = 100: 0.7
Accuracy for C = 1000: 0.7
Accuracy for C = 10000: 0.7
```

Fuente: El autor.

Podemos apreciar que para el valor del parámetro C con 10000, que se usa para controlar el inverso de la fuerza de regularización de la función de regresión logística, se obtiene un accuracy de 0.7, como lo muestra la figura 5-8. Pero el parámetro que nos da el mejor accuracy es para un C de 10 que corresponde con un accuracy de 0.74. Por lo cual este es el mejor parámetro para el modelo y con el cual vamos a ajustar el modelo definitivo, como se puede ver en la figura 5.9, donde el mejor accuracy ocurre para C igual a 10.

Figura 5-9: Ajuste del accuracy para el parámetro C.



Fuente: El autor.

Después de ajustar el modelo se obtiene la siguiente matriz de confusión:

Figura 5-10: Resultados matriz de confusión

```
Confussion matrix:
[[ 5 11]
 [ 2 32]]
```

Estos son los resultados obtenidos de las métricas:

Figura 5-11: Resultados de las métricas después del ajuste.

```
Classification report:
              precision    recall  f1-score   support

           0       0.71      0.31      0.43        16
           1       0.74      0.94      0.83        34

    accuracy                           0.74        50
   macro avg       0.73      0.63      0.63        50
weighted avg       0.73      0.74      0.70        50

Accuracy score: 0.74
```

Fuente: El autor

Ejemplos de resultados de algunas predicciones, figura 5-12. Tenemos 4 casos de reseñas, donde se han hecho predicciones con una precisión acertada, de esta forma se ha probado que el modelo de análisis de sentimientos para un accuracy de 0.74 funciona bien.

Figura 5-12: Resultados de la predicción de análisis de sentimientos para 4 reseñas.

```
Review no. 125
Actual sentiment: 1
Prediction: [1]

Review no. 194
Actual sentiment: 1
Prediction: [1]

Review no. 46
Actual sentiment: 0
Prediction: [0]

Review no. 182
Actual sentiment: 1
Prediction: [1]
```

Fuente: El autor

Los resultados nos indican que todas las predicciones fueron acertadas para los 4 usuarios que hicieron un review a un sitio visitado.

5.8 Conclusiones del capitulo

En este capítulo, se presentó la evaluación del desempeño en la recomendación de lugares a visitar por un usuario para cada uno de los modelos de sistemas de recomendación colaborativos. Para la evaluación de los modelos de recomendación se utilizó la métrica RMSE y para calificar el desempeño se utilizaron las siguientes métricas, precision@k y recall@k, siendo k el número de los mejores k lugares a recomendar.

Los modelos fueron elaborados usando el filtrado colaborativo basado en el usuario, ya que permite buscar productos y servicios según los gustos de adquisiciones previas de usuarios similares. Las validaciones se aplicaron a modelos de recomendación de filtrado colaborativo basado en el usuario, el primer modelo fue un modelo de filtrado que consideró solo las evaluaciones de los usuarios que visitaron sitios similares, y se consideró la media de las evaluaciones y luego la media ponderada de las evaluaciones,

mostrando que se obtuvo una mejor medida del RMSE con el modelo que consideró la media.

Para el modelo de sistema de recomendación usando datos demográficos se usó primero el género del usuario, y después el generó y el nivel educativo y se evidenció que el modelo tuvo un mejor desempeño usando ambos datos demográficos.

Para los modelos de sistema de recomendación basado en clustering y basado en reducción de dimensionalidad usando descomposición en valores singulares, los resultados que se obtuvieron fueron mucho mejor que los anteriores. Y finalmente el modelo de sistema de recomendación basado en sistemas de recomendación híbridos que combinó un sistema de recomendación basado en contenido y uno con descomposición en valores singulares, se obtuvieron los mejores resultados en la recomendación y en las métricas de estudio.

Para el modelo de análisis de sentimientos aplicado al modelo, después de entrenar el modelo y ajustarlo se obtuvieron resultados con una precisión de 0.74 y se mostraron algunas predicciones acertadas en el análisis de textos de recomendaciones realizadas por los usuarios a los lugares visitados. Se observa que cuando se obtienen más datos para entrenar el modelo, los resultados mejoran en la precisión y en las recomendaciones.

En el capítulo que sigue a continuación, se presentan las conclusiones y el trabajo futuro.

6. Conclusiones y trabajo futuro

6.1 Conclusiones y aportes de la tesis

En este trabajo de tesis se propone un modelo de recomendación que se puede aplicar en el ámbito del turismo para visitas guiadas, que facilita ser utilizado en recorridos para visitas en centros turísticos con lugares abiertos al exterior, lugares cerrados como museos, y en recorridos en lugares que ofrezcan varias ubicaciones con servicios que impliquen la búsqueda y toma de decisiones por parte del usuario, ejemplo un centro comercial.

El modelo permite hacer uso de la información y preferencias del usuario, la información y las características del lugar, así como de la información del contexto en el momento del recorrido de manera que el modelo hace uso de la computación ubicua, además se puede adaptar en modelos de sistemas de recomendación que trabajan con el filtrado colaborativo para recomendar lugares de interés.

El modelo propuesto en esta tesis presenta como uno de los aportes principales el modelado basado en ontologías, y propone la ontología tourist para visitas guiadas que permite representar el conocimiento referente a los perfiles y preferencias del usuario los perfiles y características del lugar e información contextual. En la ontología se permite considerar el estado emocional, las preferencias sociales y de transporte del usuario, así como los tópicos de interés para etiquetar un lugar a visitar. Igualmente, la ontología considera las características, los eventos y las preferencias de transporte del lugar. La ontología también considera la información del contexto tanto para el usuario como para el lugar; el contexto temporal, geográfico y ambiental.

Adicionalmente, el modelo permite trabajar con toda esta información del contexto y adaptarla a diferentes modelos de sistemas de recomendación, todo esto constituye capacidades y posibilidades para trabajar con modelos de recomendación que utilizan

comportamiento dinámico, y que se pueden utilizar en nuevos campos de aplicación. Estas aplicaciones podrían ser: la creación y recomendación de recorridos e itinerarios, la generación de listas personalizadas de planes de viaje, con diferentes modos de transporte y donde cada sugerencia este acompañada de una explicación detallada del plan de viaje propuesto. Al igual que hacer sugerencias dependientes del contexto.

La computación ubicua permite liberar a los usuarios de las limitaciones del tiempo y el espacio, brindando oportunidades para que los usuarios puedan acceder y disfrutar de los servicios que se ofrecen en los lugares sin importar en donde se encuentren físicamente al acceder a las tecnologías que permiten los dispositivos móviles, todas estas posibilidades pueden ser aprovechadas con las ideas presentadas en el modelo propuesto de tesis para desarrollar sistemas que dependiendo de la ubicación actual del usuario, automáticamente se muestre la información sobre los distintos lugares cercanos, sus servicios y eventos. Así como recomendar otros lugares de interés según los tópicos seleccionados por el usuario y al tener la ventaja de conocer la información de su contexto obtener mejores recomendaciones.

Como técnica de inteligencia artificial para personalizar las recomendaciones partiendo de las preferencias del contexto, el perfil y las preferencias del usuario se ha utilizado un modelo de sistema de recomendación basado en conocimiento que permite obtener recomendaciones intermedias de los lugares por medio de consultas a los conjuntos de datos que contienen la información relevante obtenida del usuario ya sea a modo de consulta o usando una interfaz de una aplicación. De esta manera el sistema busca a través de los datos, la información de las preferencias y su perfil para retornar los resultados que concuerden con una serie de reglas o directrices sobre cómo deberían de ser los resultados, o un ejemplo de un lugar recomendable.

Existen tres métodos para evaluar los sistemas de recomendación: (1) fuera de línea o sin conexión, (2) experimentos realizados en línea o conectados y (3) los estudios de usuarios.

El enfoque de los modelos de sistemas de recomendación de filtrado colaborativo presentados en esta tesis de maestría fue el de la evaluación usando el método fuera de línea, en donde se usaron datos recolectados previamente en forma simulada, en conjunto con los comportamientos simulados de los usuarios. Con el fin de examinar el desempeño de los distintos modelos de sistemas de recomendación que se presentaron en esta propuesta de tesis, se consideraron la precision y los resultados del RMSE. Los cuales

mostraron ser mejor cada vez mejores a medida que se experimentaba con modelos más sofisticados, ejemplo, usando clústeres, descomposición en valores singulares o modelos híbridos.

Se elaboraron modelos de sistemas de recomendación de filtrado colaborativo, primero usando un modelo de referencia, luego un modelo de filtrado colaborativo basado en el usuario usando la media y la media ponderada de las calificaciones de los usuarios a los lugares. También se han elaborado dos modelos de filtrado colaborativo usando datos demográficos del usuario, primero el género, y luego el género y el nivel educativo. Igualmente se han elaborado dos modelos de filtrado colaborativo con los que se busca mejorar la precisión objetiva de la clasificación de las métricas y, para esto se utilizó una técnica de validación cruzada de cinco pliegues, estos dos modelos fueron el primero basado en clustering y el segundo usando descomposición en valores singulares. Finalmente se elaboró un modelo de filtrado colaborativo basado en un sistema de recomendación hibrido.

Entre los modelos de sistemas de recomendación con filtrado colaborativo, el modelo hibrido obtuvo los mejores resultados para la RMSE y las demás métricas de accuracy, recall, y F1-score. Por esta razón es uno de los modelos de recomendación más usados en la industria.

Finalmente, se puede determinar que los objetivos propuestos en esta tesis se cumplieron a cabalidad, de la siguiente forma:

- Objetivo 1: Se realizó la caracterización de todas las clases con sus propiedades en un diagrama de clases que ofrece una perspectiva conceptual para representar los conceptos del dominio, considerando su aplicabilidad a las propiedades de ubicuidad y sensibilidad al contexto. Se elaboraron estos diagramas para el modelo del sistema de recomendación para visitas guiadas, el modelo del usuario y su personalización, el modelo del lugar y su adaptación y el modelo del contexto.
- Objetivo 2: Se desarrolló un sistema de recomendación basado en conocimiento el cual permite obtener recomendaciones intermedias de los lugares por medio de consultas a los datasets que contienen la información relevante obtenida del usuario ya sea por medio de consultas o por medio de una interfaz de usuario, para

retornar los resultados que concuerden con una serio de reglas o directrices sobre cómo deberían de los resultados o un lugar que cumpla con las restricciones.

- Objetivo 3: Se empleó una metodología que consiste en el modelado usando ontologías para representar el conocimiento del sistema propuesto y se le dio el nombre de ontología tourist, y representa el conocimiento en detalle, para relacionarlo con los elementos definidos y razonar sobre este conocimiento, para este modelo se utilizó la herramienta Owlready.
- Objetivo 4: Se implementaron aplicaciones del modelo propuesto para los casos de estudio a distintos modelos de recomendación de filtrado colaborativo y para esto se demostró que se pudo integrar la información del modelo para ser manipulada con la librería pandas. Se utilizó la técnica de aprendizaje de máquinas de análisis de sentimientos usando la información representada en el modelo; las características de los lugares visitados y las evaluaciones de los usuarios.
- Objetivo 5: Se realizó la evaluación del desempeño en la recomendación de lugares a visitar por un usuario para cada uno de los modelos de sistemas de recomendación colaborativos y para el modelo de análisis de sentimientos. Para la evaluación de los modelos de recomendación se utilizó la métrica RMSE y para calificar el desempeño se utilizaron las siguientes métricas, precision@k y recall@k, siendo k el número de los mejores k lugares a recomendar.

Adicionalmente, se realizó un aporte, como valor agregado, a los objetivos propuestos para el desarrollo de esta tesis, que consistió en la implementación de un modelo de aprendizaje de máquina con análisis de sentimientos que analiza los textos de los comentarios de los lugares visitados por los usuarios para predecir la precepción de un usuario particular acerca de un lugar que no ha visitado y de esta forma enriquece el proceso de recomendación.

6.2 Trabajo Futuro

Como trabajo futuro, se plantea usar el modelo de sistema de recomendación de filtrado colaborativo del sistema propuesto en esta tesis en el diseño de una arquitectura para una aplicación que se pueda desarrollar en plataformas móviles como IOS o Android.

También se plantea usar el modelo propuesto en este trabajo de tesis con otras técnicas de inteligencia artificial, como el uso de métodos de Deep Learning que permitan experimentar en aras de mejorar el proceso de recomendación.

Aunque el uso de datos de prueba en los distintos conjuntos de datos de los comportamientos simulados de los usuarios que se usaron mostraron tener una buena adherencia con el modelo de visitas guiadas, basado en computación ubicua y sensible al contexto es necesario examinar esta adherencia de los comportamientos del usuario basándose en una aplicación móvil que permita validar los patrones de uso o la tasa de abandono durante una prueba.

Incluir el desarrollo del sistema propuesto en un espacio turístico o cultural real y realizar su evaluación con usuarios reales en términos de experiencia de usuario y mejoras de la interacción con el sistema.

Elaborar la documentación técnica de la aplicación y participar en la publicación de artículos científicos con los resultados obtenidos.

A. Anexo: Código implementado en Python para los sistemas de recomendación aplicados al modelo

Función que retorna la raíz cuadrada del valor retornado por mean_squared_error. Esta función se usa para generar el RMSE de los modelos implementados y poder mirar su desempeño.

```
#Import the mean_squared_error function
from sklearn.metrics import mean_squared_error
import numpy as np
#Function that computes the root mean squared error (or RMSE)
def rmse(y_true, y_pred):
    return np.sqrt(mean_squared_error(y_true, y_pred))
```

En este código, se define un filtro básico que toma como parámetros el id del usuario y el id del lugar y generará un número decimal entre 1 y 5. El modelo fue construido de manera que devuelve 3 independientemente de usuario y el lugar ingresado.

```
def baseline(user_id, location_id):
    return 3
```

Para probar la potencia del modelo calculamos la RMSE que se obtiene para cada modelo particular y por cada pareja de usuario y lugar en los datos de prueba, esto se logra por medio de la siguiente función:

```
def score(cf_model):

    #Construct a list of user-movie tuples from the testing dataset
    id_pairs = zip(X_test['user_id'], X_test['location_id'])
    #Predict the rating for every location id tuple
    y_pred = np.array([cf_model(user, location) for (user, location)
    in id_pairs])
    #Extract the actual ratings given by the users in the test data
    y_true = np.array(X_test['rating'])

    df = {
        'y_actual': []
        , 'y_recommended': []
    };
```

```
    for x in y_true:
        df['y_actual'].append(x)

    for x in y_pred:
        df['y_recommended'].append(x)

    df = pd.DataFrame(df)

    df.sort_values(by='y_recommended', ascending=False, inplace=True)

    threshold = 2

    df = df >= threshold

    # print(df)

    k = 20
    pre = precision at k(df, k)
    rec = recall_at_k(df, k)
    f1score = 2 * ((pre * rec) / (pre + rec))
    print('Precision@k: {:.2f}, Recall@k: {:.2f},
           F1 Score@k: {:.2f} for   k={}'.format(pre, rec, f1score, k))

    #Return the final RMSE score
    return 'RMSE: '+ str(rmse(y_true, y_pred))+'\n'
```

- Código para calcular las métricas de Precision@k, Recall@k y F1-score:

```
def precision_at_k(df: pd.DataFrame, k: int = 3, y_test: str =
'y_actual', y_pred: str = 'y_recommended') -> float:
    """
    Function to compute precision@k for an input boolean dataframe

    Inputs:
        df     -> pandas dataframe containing boolean columns y_test &
y_pred
        k      -> integer number of items to consider
        y test -> string name of column containing actual user input
        y-pred -> string name of column containing recommendation output

    Output:
        Floating-point number of precision value for k items
    """
    # check we have a valid entry for k
    if k <= 0:
        raise ValueError('Value of k should be greater than 1, read in
as: {}'.format(k))
    # check y test & y pred columns are in df
    if y_test not in df.columns:
        raise ValueError('Input dataframe does not have a column named:
{}'.format(y_test))
    if y_pred not in df.columns:
        raise ValueError('Input dataframe does not have a column named:
{}'.format(y_pred))
```

```
    # extract the k rows
    dfK = df.head(k)
    # compute number of recommended items @k
    denominator = dfK[y_pred].sum()
    # compute number of recommended items that are relevant @k
    numerator = dfK[dfK[y_pred] & dfK[y_test]].shape[0]
    # return result
    if denominator > 0:
        return numerator / denominator
    else:
        return None


def recall_at_k(df: pd.DataFrame, k: int = 3, y_test: str = 'y_actual',
y_pred: str = 'y_recommended') -> float:
    """
    Function to compute recall@k for an input boolean dataframe

    Inputs:
        df     -> pandas dataframe containing boolean columns y_test &
y pred
        k      -> integer number of items to consider
        y_test -> string name of column containing actual user input
        y-pred -> string name of column containing recommendation output

    Output:
        Floating-point number of recall value for k items
    """
    # check we have a valid entry for k
    if k <= 0:
        raise ValueError('Value of k should be greater than 1, read in
as: {}'.format(k))
    # check y_test & y_pred columns are in df
    if y_test not in df.columns:
        raise ValueError('Input dataframe does not have a column named:
{}'.format(y test))
    if y_pred not in df.columns:
        raise ValueError('Input dataframe does not have a column named:
{}'.format(y_pred))

    # extract the k rows
    dfK = df.head(k)
    # compute number of all relevant items
    denominator = df[y_test].sum()
    # compute number of recommended items that are relevant @k
    numerator = dfK[dfK[y_pred] & dfK[y_test]].shape[0]
    # return result
    if denominator > 0:
        return numerator / denominator
    else:
        return None
```

- La siguiente función se usa para obtener la media de los puntajes de todos los usuarios dado a un solo lugar y se le calcula la media, a todos los puntajes se les asigna igual peso:

```
def cf_user_mean(user_id, location_id):
    #Check if location_id exists in r_matrix
    if location_id in r_matrix:
        #Compute the mean of all the ratings given to the location
        mean_rating = r_matrix[location_id].mean()
    else:
        #Default to a rating of 3.0 in the absence of any information
        mean_rating = 3
    return mean_rating
```

- Código para generar la matriz de similitud de distancia del coseno para los usuarios y la función para trabajar con la media ponderada.

```
# Weighted mean: Media ponderada

#Create a dummy ratings matrix with all null values
# imputed to 0
r_matrix_dummy = r_matrix.copy().fillna(0)

# Import cosine score
from sklearn.metrics.pairwise import cosine_similarity

#Compute the cosine similarity matrix using the dummy ratings matrix
cosine_sim = cosine_similarity(r_matrix_dummy, r_matrix_dummy)

#Convert into pandas dataframe
cosine_sim = pd.DataFrame(cosine_sim,
index=r_matrix.index,columns=r_matrix.index)

print(cosine_sim.head(10))

#User Based Collaborative Filter using Weighted Mean Ratings
def cf_user_wmean(user_id, movie_id):
    #Check if movie id exists in r matrix
    if movie_id in r_matrix:
        #Get the similarity scores for the user in question with every
other user
        sim_scores = cosine_sim[user_id]
        #Get the user ratings for the movie in question
        m_ratings = r_matrix[movie_id]
        #Extract the indices containing NaN in the m_ratings series
        idx = m_ratings[m_ratings.isnull()].index
        #Drop the NaN values from the m ratings Series
        m_ratings = m_ratings.dropna()
        #Drop the corresponding cosine scores from the sim_scores series
        sim_scores = sim_scores.drop(idx)
        #Compute the final weighted mean
        if sim_scores.sum() != 0:
            wmean_rating = np.dot(sim_scores, m_ratings)/
sim_scores.sum()
```

```
        else:
            wmean_rating = 3
    else:#Default to a rating of 3.0 in the absence of any information
        wmean_rating = 3
    return wmean_rating
```

- El puntaje de similitud del coseno

El puntaje de similitud del coseno es una métrica muy utilizada para construir modelos de recomendación. El puntaje del coseno es extremadamente robusto y fácil de calcular y se usa en conjunto con la función TF-IDFVectorizer.

El puntaje de similitud del coseno entre dos textos, x y, y es como se indica a continuación:

$$coseno(x, y) = \frac{x.y^T}{\|x\|.\|y\|}$$

El puntaje del coseno puede tomar cualquier valor entre -1 y 1. Mientras más alto sea el puntaje del coseno, más parecidos son los documentos.

- Cálculo de la media del puntaje de cada lugar por género:

```
merged = pd.merge(X_train,users,on='user_id')
```

- Función para calcular la media del puntaje por género:

```
def cf_gender(user_id, location_id):
    #Check if location id exists in r matrix (or training set)
    if location_id in r_matrix:
        #Identify the gender of the user
        gender = users.loc[user_id]['gender']
        #Check if the gender has rated the movie
        if gender in gender_mean[location_id]:
            #Compute the mean rating given by that gender to the
location
            gender_rating = gender_mean[location_id][gender]
        else:
            gender_rating = 3
    else:
        #Default to a rating of 3.0 in the absence of any information
        gender_rating = 3
    return gender_rating
```

- Código para generar el modelo de recomendación usando los datos demográficos del género y el nivel educativo:

```
#Compute the mean rating by gender and education
gen_edu_mean = merged[['gender', 'rating', 'location_id',
'education']].pivot_table(
values='rating', index='location_id', columns=['education', 'gender'],
aggfunc='mean')

#Gender and Eduation Based Collaborative Filter using Mean Ratings
def cf_gen_edu(user_id, location_id):
    #Check if location id exists in gen edu mean
    if location_id in gen_edu_mean.index:
        #Identify the user
        user = users.loc[user_id]
        #Identify the gender and education
        gender = user['gender']
        edu = user['education']
        #Check if the education has rated the movie
        if edu in gen_edu_mean.loc[location_id]:
            #Check if the gender has rated the movie
            if gender in gen edu mean.loc[location id][edu]:
                #Extract the required rating
                rating = gen_edu_mean.loc[location_id][edu][gender]
                #Default to 3.0 if the rating is null
                if np.isnan(rating):
                    rating = 3
                return rating
    #Return the default rating
    return 3
```

- Construcción y evaluación del modelo del filtro colaborativo basado en KNN.

```
#Define a Reader object
#The Reader object helps in parsing the file or dataframe containing
ratings
reader = Reader()

ratings = pd.read_csv('Userdata/ratings.csv')

#Create the dataset to be used for building the filter
# ratings is the .csv file
data = Dataset.load_from_df(ratings, reader)

#Provides train/test indices to split data in train/test sets.
kf = KFold(n_splits=5)

#Define the algorithm object; in this case kNN
knn = KNNBasic()

#Evaluate the performance in terms of RMSE
# the filter is making use of a technique known as
# fivefold cross-validation
cross validate(knn, data, measures=['RMSE'], verbose=True)
```

- Código que implementa el filtro SVD usando la librería surprise:

```
from surprise import SVD
#Provides train/test indices to split data in train/test sets.
```

```
kf = KFold(n_splits=5)
#Define the SVD algorithm object

#Provides train/test indices to split data in train/test sets.
kf = KFold(n_splits=5)

svd = SVD()
print()
#Evaluate the performance in terms of RMSE
cross_validate(svd, data, measures=['RMSE'], verbose=True)
```

- Función para construir el filtrado híbrido:

```
def hybrid(userId, name):
    # Extract the id of the location
    idx = indices[name]

    # Extract the similarity scores and their corresponding index for
every location from
    # the cosine_sim matrix
    sim_scores = list(enumerate(cosine_sim[idx]))

    # Sort the (index, score) tuples in decreasing order of similarity
scores
    sim_scores = sorted(sim_scores, key=lambda x: x[1], reverse=True)

    # Select the top 10 tuples, excluding the first
    # (as it is the similarity score of the location with itself)
    sim_scores = sim_scores[1:11]

    # Store the cosine_sim indices of the top 10 loctions in a list
    location_indices = [i[0] for i in sim_scores]

    # Extract the metadata of the aforementioned locations
    loc = locations.iloc[location_indices][['id_location',
'name_location', 'sum_scores', 'vote_count', 'overview']]



    # Compute the predicted ratings using the SVD filter
    #  est: estimation
    loc['est'] = loc['id_location'].apply(lambda x: svd.predict(userId,
id_to_title.loc[x]
    ['name_location']).est)

    # Sort the locations in decreasing order of predicted rating
    loc = loc.sort_values('est', ascending=False)

    # Return the top 5 locations as recommendations
    return loc.head(5)
```

B. Resultados de la recomendación usando el sistema de recomendación híbrido para los casos de estudio

Caso 1:

```
mostrar(hybrid(6, 'California Club'))
```

29 id_location 30
name_location Casino Platinium Club
sum_scores 4623
vote_count 1200
overview El lugar es muy fresco. La decoración es excel...
est 2.875074
Name: 29, dtype: object
---

21 id_location 22
name_location Mystic Club
sum_scores 4300
vote_count 1200
overview Discoteca y club nocturno. Buen ambiente y muy...
est 2.875074
Name: 21, dtype: object
---

17 id_location 18
name_location Fase II
sum_scores 3004
vote_count 700
overview Discoteca con las mujeres más hermosas y el me...
est 2.875074
Name: 17, dtype: object
---

27 id_location 28
name_location Porto
sum_scores 10230
vote_count 2600
overview Los tragos son deliciosos, música ambiental ta...
est 2.875074
Name: 27, dtype: object

---

38 id_location 38
name_location Connolly's Pub & Restaurant
sum_scores 30000
vote_count 6700
overview Buenas y variadas cenas, amabilidad y profesio...
est 2.875074
Name: 38, dtype: object

Caso 2

```
mostrar(hybrid(49, 'Cedar Point'))
```

0 id_location 1
name_location El correo licor y comidas 24 horas
sum_scores 101
vote_count 30
overview Gran restaurante para cenar en una zona muy tr...
est 3.291195
Name: 0, dtype: object
---

1 id_location 2
name_location JPMorgan FIX Tattoo
sum_scores 94
vote_count 39
overview Tienda de tatuajes y piercings en Itagüi.
est 3.291195
Name: 1, dtype: object
---

2 id_location 3
name_location AMERICAN MUSEUM OF NATURAL HISTORY
sum_scores 5300
vote_count 1520
overview El Museo Americano de Historia Natural es uno ...

```
est                                                    3.291195
Name:                    2,                  dtype:    object
                                  ---

3  id_location                                                4
name_location                                           Beavers
sum_scores                                                  780
vote_count                                                  250
overview        El castor no es el roedor típico. Es el más gr...
est                                                    3.291195
Name:                    3,                  dtype:    object
                                  ---

4  id_location                                                5
name_location                                           Sharks
sum_scores                                                  960
vote_count                                                  300
overview        La nueva exposición Tiburones presenta docenas...
est                                                    3.291195
Name:                    4,                  dtype:    object
  ---
```

C. Código para construir el modelo de análisis de sentimientos

```
import pandas as pd
import numpy as np
import random
import nltk
#nltk.download('punkt')
import nltk.data

#nltk.download('stopwords')

from nltk.corpus import stopwords
spanish_stopwords = stopwords.words('spanish')

for sw in ['eramos', 'estabamos', 'estais', 'estan', 'estara',
'estaran', 'estaras', 'estare', 'estareis', 'estaria', 'estariais',
'estariamos', 'estarian', 'estarias', 'esteis', 'esten', 'estes',
'estuvieramos', 'estuviesemos', 'fueramos', 'fuesemos', 'habeis',
'habia', 'habiais', 'habiamos', 'habian', 'habias', 'habra', 'habran',
'habras', 'habre', 'habreis', 'habria', 'habriais', 'habriamos',
'habrian', 'habrias', 'hayais', 'hubieramos', 'hubiesemos', 'mas',
'mia', 'mias', 'mio', 'mios', 'seais', 'sera', 'seran', 'seras', 'sere',
'sereis', 'seria', 'seriais', 'seriamos', 'serian', 'serias', 'si',
'tambien', 'tendra', 'tendran', 'tendras', 'tendre', 'tendreis',
'tendria', 'tendriais', 'tendriamos', 'tendrian', 'tendrias', 'teneis',
'tengais', 'tenia', 'teniais', 'teniamos', 'tenian', 'tenias',
'tuvieramos', 'tuviesemos']:
    spanish_stopwords.append(sw)

# print(spanish_stopwords)

from sklearn.feature_extraction.text import TfidfVectorizer,
CountVectorizer
from sklearn.model_selection import train_test_split
from sklearn.pipeline import Pipeline
from sklearn.feature_selection import chi2
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report, roc_curve

import matplotlib
from matplotlib import pyplot as plt


# Read in data:
ratings = pd.read_csv('Userdata/ratings.csv')

#Replace NaN with an empty string
ratings['review'] = ratings['review'].fillna('')
```

```
#print(ratings['review'])

# Construct the required TF-IDF matrix by applying the fit_transform
method on the
# review feature
# tfidf_matrix = tfidf.fit_transform(ratings['review'])

#Output the shape of tfidf_matrix
# print(tfidf matrix.shape)

# print(ratings['review'][33])

tokenizer = nltk.data.load('tokenizers/punkt/spanish.pickle')


def my_word_tokenizer(mydata):
    my_tokens = []
    for token in tokenizer.tokenize(mydata):
        for t in nltk.word tokenize(token):
            my_tokens.append(t)
    return my_tokens


# print(my_word_tokenizer(ratings['review'][33]))

# print(ratings[['review','rating']])

# El estado neutro sería 3, más de 3 sería positivo y
# menos de 3 sería un sentimiento negativo.
set(ratings['rating'])
# print(set(ratings['rating']))


def label_sentiment(row):
    if int(row['rating']) < 3:
        return 0
    else:
        return 1


ratings['sentiment_label'] = ratings.apply (lambda row: label_sentiment
(row), axis = 1)

# print(ratings.head (5))

X_train, X_test, y_train, y_test = train_test_split(
       ratings['review'],
       ratings['sentiment_label'],
       train_size = 0.75,
       test_size = 0.25,
       random_state = 42,
       shuffle = True
)

# print(X_train.iloc [:10])
```

```
#Define a TF-IDF Vectorizer Object. Remove all spanish stopwords
cv = TfidfVectorizer (
        ngram_range = (1, 3),
        strip_accents = 'ascii',
        max_df = 0.95,
        min_df = 3,
        max_features = 3000,
        stop_words = spanish_stopwords
)

cv.fit (X_train)

# print(len(cv.vocabulary .items ()))
X_train_ = cv.transform (X_train)
X_test_ = cv.transform (X_test)

# print(X_train.iloc [0])

ejemplo = 'excelente lugar para realizar eventos como convenciones, es
muy amplio y cuenta con una plaza donde puedes encontrar restaurantes y
tiendas.'

tokenized_ejemplo = my_word_tokenizer(ejemplo)

# TF-IDFVectorizer: TERM FREQUENCY-INVERSE DOCUMENT FREQUENCY
# some words occur much more frequenly in reviews than others.

vocab_idf = dict (zip (cv.get_feature_names_out (), cv.idf_))

#
print ('{0:20} {1:20}'.format ('Palabra', 'IDF'))
for word in tokenized ejemplo:
      if word not in vocab_idf:
            print ('{0:20} {1:20}'.format (word, '00V'))
      else:
            print ('{0:20} {1:2.3f}'.format (word, vocab_idf [word]))

# Entrenamiento de ejercicio de análisis de sentimientos

# Logistic Regression
# Logistic regression aims to solve classification problems.
# It does this by predicting categorical outcomes,
# In this case there are two outcomes, which is called binomial,
# an example of which is predicting if a review has a negative or
positive sentiment.

# Here we will be using basic logistic regression to predict a binomial
variable.
# This means it has only two possible outcomes.


c_params = [0.01, 0.05, 0.25, 0.5, 1, 10, 100, 1000, 10000]

train_acc = list ()
test_acc = list ()
for c in c_params:
```

```
    lr = LogisticRegression (C = c, solver = 'lbfgs', max_iter = 500)
    lr.fit (X_train_, y_train)

    train_predict = lr.predict (X_train_)
    test_predict = lr.predict (X_test_)

    print ("Accuracy for C = {}: {}".format (c, accuracy_score (y_test,
test_predict)))

    train_acc.append (accuracy_score (y_train, train_predict))
    test_acc.append (accuracy_score (y_test, test_predict))

# Algunas métricas
# Para cada uno de los algoritmos que hemos entrenado, vamos a ver las
medias que obtenemos:

#Entrenamiento de ejercicio de análisis de sentimientos
print ('Confussion matrix: \n{}'.format (confusion_matrix (y_test,
test predict)))
print ('\nClassification report: \n{}'.format (classification_report
(y_test, test_predict)))
print ('Accuracy score: {}'.format (accuracy_score (y_test,
test_predict)))

# Podemos inferir de la tabla anterior que el algoritmo es bastante
flojo
# cuando se trata de reviews negativas, ya que la precisión es bastante
baja.

plt.figure (figsize = (12, 8))
plt.plot (train_acc, label = 'train')
plt.plot (test_acc, label = 'test')
plt.axvline (np.argmax (test_acc), c = 'g', ls = '--', alpha = 0.8)
plt.title ('Accuracy evolution for different C values')
plt.xlabel ('C')
plt.ylabel ('Accuracy')
plt.legend ()
plt.grid ()
plt.xticks (list (range (len (c_params))), c_params)
plt.tight_layout ()
plt.show ()

# Aquí vemos cómo evoluciona el performance en función del parámetro
# que hemos propuesto inicialmente. Vemos, por ejemplo, que en el punto
# 10 es cuando más se estabiliza; este podría considerarse el valor
óptimo.

# Predict
# Ahora haremos la predicción:
def predict_review_sentiment(review_index, model):
    print('Actual sentiment:
{}'.format(ratings.iloc[review_index]['sentiment_label']))
    r = ratings.iloc[review_index]['review']
    print('Prediction: {}'.format(lr.predict(cv.transform([r]))))

for i in random.sample (range (0, len (ratings)), 5):
```

```
        print ('\nReview no. {}'.format (i))
        predict_review_sentiment (i, lr)
```

D. Definición de las instancias de la ontología tourist

```
#  -- - - - Instancias de la ontologia tourist-----------------

# Se crean las instancias o individuos de tipo EmotionalState:

happy = tourist.Happy("happ1_1")
sad = tourist.Sad("sad_1")
angry = tourist.Angry("angry_1")
calm = tourist.Calm("calm_1")
surprised = tourist.Surprised("surprised_1")
tired = tourist.Tired("tired_1")
normal = tourist.Normal("normal_1")
AllDifferent([happy, sad, angry, calm, surprised, tired, normal])

# Se crean las instancias o individuos para gestionar el perfil solcial
del usuario Companion:
single = tourist.Single("single_1")
group = tourist.Group("group_1")
couple = tourist.Couple("couple_1")
AllDifferent([single, group, couple])

# Se crean las instancias o individuos para gestionar el topico
Accomodation:
apartment = tourist.Apartment("apartment_1")
bungalow = tourist.Bungalow("bungalow_1")
cabin = tourist.Cabin("cabin")
dormitory = tourist.Dormitory("dormitory_1")
farm = tourist.Farm("farm_1")
hotel = tourist.Hotel("hotel_1")
house = tourist.House("house")
houseBoat = tourist.HouseBoat("houseBoat_1")
treeHouse = tourist.TreeHouse("treeHouse_1")
chalet = tourist.Chalet("chalet_1")
gestHouse = tourist.GuestHouse("gestHouse_1")
hostel = tourist.Hostel("hostel_1")
AllDifferent([apartment, bungalow,
cabin,dormitory,farm,hotel,house,houseBoat,treeHouse,chalet,gestHouse,ho
stel])
```

```
# Se crean las instancias o individuos para gestionar el topico
Services:
atm = tourist.Atm("atm_1")
bank = tourist.Bank("bank")
currencyExchange = tourist.CurrencyExchange("currencyExchange_1")
internetCafe = tourist.InternetCafe("internetCafe_1")
vendingMachine = tourist.VendingMachine("vendingMachine_1")
commercial = tourist.Commercial("commercial_1")
kiosk = tourist.Kiosk("kiosk_1")
oficce = tourist.Oficce("oficce_1")
supermarket = tourist.Supermarket("supermarket_1")
coworking = tourist.Coworking("coworking_1")
clinic = tourist.Clinic("clinic_1")
AllDifferent([atm,bank,currencyExchange,internetCafe,vendingMachine,comm
ercial,kiosk,oficce,supermarket,coworking,clinic])

# Se crean las instancias o individuos para gestionar el topico Food:
bar = tourist.Bar("bar_1")
biergarten = tourist.Biergarten("biergarten 1")
cafe = tourist.Cafe("cafe_1")
fastFood = tourist.FastFood("fastFood_1")
foodCourt = tourist.FoodCourt("foodCourt_1")
iceCream = tourist.IceCream("iceCream_1")
pub = tourist.Pub("pub_1")
restaurant = tourist.Restaurant("restaurant_1")
AllDifferent([bar,biergarten,cafe,fastFood,foodCourt,iceCream,pub,restau
rant])

# Se crean las instancias o individuos para gestionar el topico
Transportation:
bicycleParking = tourist.BicycleParking("bicycleParking_1")
bicycleRepairStation =
tourist.BicycleRepairStation("bicycleRepairStation_1")
bicycleRental = tourist.BicycleRental("bicycleRental_1")
boatRental = tourist.BoatRental("boatRental_1")
busStation = tourist.BusStation("busStation_1")
carRental = tourist.CarRental("carRental_1")
carWash = tourist.CarWash("carWash_1")
compressedAir = tourist.CompressedAir("compressedAir_1")
chargingStation = tourist.ChargingStation("chargingStation_1")
ferryTerminal = tourist.FerryTerminal("ferryTerminal_1")
fuelStation = tourist.FuelStation("fuelStation_1")
motorCycleParking = tourist.MotorCycleParking("motorCycleParking_1")
parking = tourist.Parking("parking_1")
parkingSpace = tourist.ParkingSpace("parkingSpace_1")
taxiWait = tourist.TaxiWait("taxiWait_1")
station = tourist.Station("station_1")
AllDifferent([bicycleParking,bicycleRepairStation,bicycleRental,boatRent
al,busStation,carRental,carWash,compressedAir,chargingStation,ferryTermi
nal,fuelStation,motorCycleParking,parking,parkingSpace,taxiWait,station]
)

# Se crean las instancias o individuos para gestionar el topico
NationalCulture:
artsCentre = tourist.ArtsCentre("artsCentre_1")
communityCentre = tourist.CommunityCentre("communityCentre_1")
```

```
conferenceCentre = tourist.ConferenceCentre("conferenceCentre_1")
eventsVenue = tourist.EventsVenue("eventsVenue_1")
fountain = tourist.Fountain("fountain_1")
socialCentre = tourist.SocialCentre("socialCentre_1")
townHall = tourist.TownHall("townHall_1")
publicBath = tourist.PublicBath("publicBath_1")
AllDifferent([artsCentre,communityCentre,conferenceCentre,eventsVenue,fo
untain,socialCentre,townHall,publicBath])

# Se crean las instancias o individuos para gestionar el topico
Activity:
bbq = tourist.Bbq("bbq_1")
diveCentre = tourist.DiveCentre("diveCentre_1")
hunting = tourist.Hunting("hunting_1")
caravanSite = tourist.CaravanSite("caravanSite_1")
picnicSite = tourist.PicnicSite("picnicSite_1")
themePark = tourist.ThemePark("themePark_1")
wildernessHut = tourist.WildernessHut("wildernessHut_1")
zoo = tourist.Zoo("zoo_1")
AllDifferent([bbq,diveCentre,hunting,caravanSite,picnicSite,themePark,wi
ldernessHut,zoo])

# Se crean las instancias o individuos para gestionar el topico
Shopping:
marketPlace = tourist.MarketPlace("marketPlace_1")
alcohol = tourist.Alcohol("alcohol_1")
bakery = tourist.Bakery("bakery_1")
beverages = tourist.Beverages("beverages_1")
butcher = tourist.Butcher("butcher_1")
cheese = tourist.Cheese("cheese_1")
chocolate = tourist.Chocolate("chocolate_1")
coffee = tourist.Coffee("coffee 1")
confectionery = tourist.Confectionery("confectionery_1")
convinience = tourist.Convinience("convinience_1")
deli = tourist.Deli("deli_1")
dairy = tourist.Dairy("dairy_1")
frozenFood = tourist.FrozenFood("frozenFood 1")
greenGrocer = tourist.GreenGrocer("greenGrocer_1")
healthFood = tourist.HealthFood("healthFood_1")
pastry = tourist.Pastry("pastry_1")
seaFood = tourist.SeaFood("seaFood_1")
spices = tourist.Spices("spices_1")
tea = tourist.Tea("tea_1")
wine = tourist.Wine("wine_1")
water = tourist.Water("water_1")
boutique = tourist.Boutique("boutique_1")
clothes = tourist.Clothes("clothes_1")
jewelry = tourist.Jewelry("jewelry_1")
leather = tourist.Leather("leather_1")
shoes = tourist.Shoes("shoes_1")
tailor = tourist.Tailor("tailor_1")
beauty = tourist.Beauty("beauty_1")
cosmetics = tourist.Cosmetics("cosmetics_1")
hairdresser = tourist.Hairdresser("hairdresser_1")
tattoo = tourist.Tattoo("tattoo_1")
antiques = tourist.Antiques("antiques_1")
```

```
art = tourist.Art("art_1")
craft = tourist.Craft("craft_1")
gift = tourist.Gift("gift_1")
stationery = tourist.Stationery("stationery_1")
pet = tourist.Pet("pet_1")
petGrooming = tourist.PetGrooming("petGrooming_1")
toys = tourist.Toys("toys_1")
AllDifferent([marketPlace,alcohol,bakery,beverages,butcher,cheese,chocol
ate,coffee,confectionery,convinience,deli,

dairy,frozenFood,greenGrocer,healthFood,pastry,seaFood,spices,tea,wine,w
ater,boutique,clothes,

jewelry,leather,shoes,tailor,beauty,cosmetics,hairdresser,tattoo,antique
s,art,craft,gift,
              stationery,pet,petGrooming,toys])

# Se crean las instancias o individuos para gestionar el topico
Entertaintment:
casino = tourist.Casino("casino_1")
cinema = tourist.Cinema("cinema_1")
loveHotel = tourist.LoveHotel("loveHotel_1")
nightClub = tourist.NightClub("nightClub_1")
planetarium = tourist.Planetarium("planetarium_1")
theatre = tourist.Theatre("theatre_1")
sportsHall = tourist.SportsHall("sportsHall_1")
stadium = tourist.Stadium("stadium_1")
recreationGround = tourist.RecreationGround("recreationGround_1")
adultGamingCentre = tourist.AdultGamingCentre("adultGamingCentre_1")
amusementArcade = tourist.AmusementArcade("amusementArcade_1")
beachResort = tourist.BeachResort("beachResort_1")
bandStand = tourist.BandStand("bandStand_1")
dogPark = tourist.DogPark("dogPark_1")
campSite = tourist.CampSite("campSite_1")
fishing = tourist.Fishing("fishing_1")
fitnessCentre = tourist.FitnessCentre("fitnessCentre_1")
miniatureGolf = tourist.MiniatureGolf("miniatureGolf_1")
playground = tourist.Playground("playground_1")
sportsCentre = tourist.SportsCentre("sportsCentre_1")
summerCamp = tourist.SummerCamp("summerCamp_1")
swimmingPool = tourist.SwimmingPool("swimmingPool_1")
waterPark = tourist.WaterPark("waterPark_1")
aquarium = tourist.Aquarium("aquarium_1")
AllDifferent([casino,cinema,loveHotel,nightClub,planetarium,theatre,spor
tsHall,stadium,recreationGround,

adultGamingCentre,amusementArcade,beachResort,bandStand,dogPark,campSite
,fishing,fitnessCentre,

miniatureGolf,playground,sportsCentre,summerCamp,swimmingPool,waterPark,
aquarium])

# Se crean las instancias o individuos para gestionar el topico
NaturalLandscape:
aboriginalLand = tourist.AboriginalLand("aboriginalLand_1")
nationalPark = tourist.NationalPark("nationalPark_1")
```

```
villageGreen = tourist.VillageGreen("villageGreen_1")
garden = tourist.Garden("garden_1")
natureReserve = tourist.NatureReserve("natureReserve_1")
park = tourist.Park("park_1")
pier = tourist.Pier("pier_1")
bay = tourist.Bay("bay_1")
beach = tourist.Beach("beach_1")
geyser = tourist.Geyser("geyser_1")
glacier = tourist.Glacier("glacier_1")
isthmus = tourist.Isthmus("isthmus_1")
reef = tourist.Reef("reef_1")
caveEntrance = tourist.CaveEntrance("caveEntrance_1")
cliff = tourist.Cliff("cliff_1")
peak = tourist.Peak("peak_1")
river = tourist.River("river_1")
lake = tourist.Lake("lake_1")
stream = tourist.Stream("stream_1")
waterFall = tourist.WaterFall("waterFall_1")
AllDifferent([aboriginalLand,nationalPark,villageGreen,garden,natureRese
rve,park,pier,bay,beach,geyser,glacier,isthmus,
              reef,caveEntrance,cliff,peak,river,lake,stream,waterFall])

# Se crean las instancias o individuos para gestionar el topico
CulturalLandscape:
monastery = tourist.Monastery("monastery_1",name_topic="monastery")
church = tourist.Church("church_1")
mosque = tourist.Mosque("mosque_1")
temple = tourist.Temple("temple_1")
cathedral = tourist.Cathedral("cathedral_1")
chapel = tourist.Chapel("chapel_1")
synagogue = tourist.Synagogue("synagogue_1")
bridge = tourist.Bridge("bridge 1")
castle = tourist.Castle("castle_1")
ruin = tourist.Ruin("ruin_1")
aqueduct = tourist.Aqueduct("aqueduct_1")
archaeologicalSite = tourist.ArchaeologicalSite("archaeologicalSite_1")
battleField = tourist.BattleField("battleField 1")
historicBuilding = tourist.HistoricBuilding("historicBuilding_1")
castleWall = tourist.CastleWall("castleWall_1")
cityGate = tourist.CityGate("cityGate_1")
cityWall = tourist.CityWall("cityWall_1")
historicDistrict = tourist.HistoricDistrict("historicDistrict_1")
historicalFarm = tourist.HistoricalFarm("historicalFarm_1")
militaryFort = tourist.MilitaryFort("militaryFort_1")
locomotive = tourist.Locomotive("locomotive_1")
historicMansion = tourist.HistoricMansion("historicMansion_1")
memorial = tourist.Memorial("memorial_1")
monument = tourist.Monument("monument_1")
railwayCar = tourist.RailwayCar("CulturalLandscape_1")
ship = tourist.Ship("ship_1")
tank = tourist.Tank("tank_1")
tomb = tourist.Tomb("tomb_1")
tower = tourist.Tower("tower_1")
vehicle = tourist.Vehicle("vehicle_1")
cemetery = tourist.Cemetery("cemetery_1")
cross = tourist.Cross("cross_1")
```

```
obelisk = tourist.Obelisk("obelisk_1")
windMill = tourist.WindMill("windMill_1")
town = tourist.Town("town_1")
village = tourist.Village("village_1")
square = tourist.Square("square_1")
artWork = tourist.ArtWork("artWork_1")
attraction = tourist.Attraction("attraction_1")
gallery = tourist.Gallery("gallery_1")
museum = tourist.Museum("museum_1")
exhibition = tourist.Exhibition("exhibition_1")
AllDifferent([monastery,church,mosque,temple,cathedral,chapel,synagogue,
bridge,castle,ruin,

aqueduct,archaeologicalSite,battleField,historicBuilding,castleWall,city
Gate,

cityWall,historicDistrict,historicalFarm,militaryFort,locomotive,histori
cMansion,memorial,

monument,railwayCar,ship,tank,tomb,tower,vehicle,cemetery,cross,obelisk,
windMill,town,village,
              square,artWork,attraction,gallery,museum,exhibition])

# Se crean las instancias o individuos para gestionar el contexto
Geographical:
inside = tourist.Inside("inside_1")
outside = tourist.Outside("outside_1")
service = tourist.Service("service_1")
AllDifferent([inside,outside,service])

animals = tourist.KeyWord("animals")
ecology = tourist.KeyWord("ecology")
forests = tourist.KeyWord("forests")
planetarium = tourist.KeyWord("planetarium")
space = tourist.KeyWord("space")
solar_system = tourist.KeyWord("solar_system")
salud = tourist.KeyWord("salud")
consulta = tourist.KeyWord("consulta")
pacientes = tourist.KeyWord("pacientes")
urgencias = tourist.KeyWord("urgencias")
hospitalizacion = tourist.KeyWord("hospitalizacion")
muestras = tourist.KeyWord("muestras")
laboratorio = tourist.KeyWord("laboratorio")
night = tourist.KeyWord("night")
AllDifferent([animals,ecology,forests,planetarium,space,solar_system,sal
ud,consulta,pacientes,urgencias,
              hospitalizacion,muestras,laboratorio,night])

walking = tourist.Walking("walking")
public = tourist.Public("public")
bike = tourist.Bike("bike")
private = tourist.Private("private")
AllDisjoint([walking, public, bike, private])
```

E. Formalización e implementación de la ontología tourist

```
from owlready2 import *

""" ONTOLOGIA tourist: 21 de Octubre de 2022"""
""" AUTHOR: Juan Carlos Gil Vera"""

tourist = get_ontology("http://test.org/tourist.owl·")

with tourist:

    class GenericPreference(Thing): pass
    class User(Thing): pass

    # Clases para gestionar el historial del usuario
    class Record(Thing): pass
    class CheckIn(Record): pass
    class PastLocation(Record): pass
    AllDisjoint([CheckIn, PastLocation])

    # Clases para gestionar los tipos de Ubicación turistica
    class Location(Thing): pass
    class InsideLocation(Location): pass
    class OutsideLocation(Location): pass
    class ServiceLocation(Location): pass
    class AdequateLocationTemporal(Location): pass  # Location que
concuerda con el contexto temporal.
    class AdequateLocationTopic(Location): pass  # Location que
concuerda con el topico.
    class AdequateLocationUnderAge(Location): pass  # Location que
permite a menores de edad.
    AllDisjoint([InsideLocation, OutsideLocation, ServiceLocation])

    class Schedule(Thing): pass
    class AdequateSchedule(Schedule): pass # Schedule de las Location
que concuerda con el contexto temporal.

    # Clases para gestionar los perfiles, carateristicas
    # y preferencias del  Usuario y la Ubicación
    class Characteristic(Thing): pass
    class Preference(Thing): pass

    # Clases para gestionar el estado emocional del usuario
    class EmotionalState(Thing): pass
    class Happy(EmotionalState): pass
    class Sad(EmotionalState): pass
    class Angry(EmotionalState): pass
    class Calm(EmotionalState): pass
    class Surprised(EmotionalState): pass
    class Tired(EmotionalState): pass
    class Normal(EmotionalState): pass
    AllDisjoint([Happy, Sad, Angry, Calm, Surprised, Tired, Normal])
```

```
    # Clases para gestionar el perfil solcial del usuario
    class Companion(Thing): pass
    class Single(Companion): pass
    class Group(Companion): pass
    class Couple(Companion): pass
    AllDisjoint([Single, Group, Couple])

    # Clases para gestionar los medios de transporte
    class TransportationPref(Thing): pass
    class Walking(TransportationPref): pass
    class Public(TransportationPref): pass
    class Bike(TransportationPref): pass
    class Private(TransportationPref): pass
    AllDisjoint([Walking, Public, Bike, Private])

    # Clases para gestionar los tipos de lugares turisticos por Topico
    class Topic(Thing): pass

    class Accomodation(Topic): pass  # TOPICO: Accomodation
    class Apartment(Accomodation): pass
    class Bungalow(Accomodation): pass
    class Cabin(Accomodation): pass
    class Dormitory(Accomodation): pass
    class Farm(Accomodation): pass
    class Hotel(Accomodation): pass
    class House(Accomodation): pass
    class HouseBoat(Accomodation): pass
    class TreeHouse(Accomodation): pass
    class Chalet(Accomodation): pass
    class GuestHouse(Accomodation): pass
    class Hostel(Accomodation): pass
    AllDisjoint(
        [Apartment, Bungalow, Cabin, Dormitory, Farm, Hotel, House,
HouseBoat, TreeHouse, Chalet, GuestHouse, Hostel])


    class Services(Topic): pass  # TOPICO: Services
    class Atm(Services): pass
    class Bank(Services): pass
    class CurrencyExchange(Services): pass
    class InternetCafe(Services): pass
    class VendingMachine(Services): pass
    class Commercial(Services): pass
    class Kiosk(Services): pass
    class Oficce(Services): pass
    class Supermarket(Services): pass
    class Coworking(Services): pass
    class Clinic(Services): pass
    AllDisjoint(
        [Atm, Bank, CurrencyExchange, InternetCafe, VendingMachine,
Commercial, Kiosk, Oficce, Supermarket, Coworking, Clinic])


    class Transportation(Topic): pass  # TOPICO: Transportation
    class BicycleParking(Transportation): pass
```

```
    class BicycleRepairStation(Transportation): pass
    class BicycleRental(Transportation): pass
    class BoatRental(Transportation): pass
    class BusStation(Transportation): pass
    class CarRental(Transportation): pass
    class CarWash(Transportation): pass
    class CompressedAir(Transportation): pass
    class ChargingStation(Transportation): pass
    class FerryTerminal(Transportation): pass
    class FuelStation(Transportation): pass
    class MotorCycleParking(Transportation): pass
    class Parking(Transportation): pass
    class ParkingSpace(Transportation): pass
    class TaxiWait(Transportation): pass
    class Station(Transportation): pass
    AllDisjoint([BicycleParking, BicycleRepairStation,
                 BicycleRental, BoatRental, BusStation, CarRental,
                 CarWash, CompressedAir, ChargingStation, FerryTerminal,
Station,
                 FuelStation, MotorCycleParking, Parking, ParkingSpace,
TaxiWait])


    class Food(Topic): pass  # TOPICO: Food
    class Bar(Food): pass
    class Biergarten(Food): pass
    class Cafe(Food): pass
    class FastFood(Food): pass
    class FoodCourt(Food): pass
    class IceCream(Food): pass
    class Pub(Food): pass
    class Restaurant(Food): pass
    AllDisjoint([Bar, Biergarten, Cafe, FastFood, FoodCourt, IceCream,
Pub, Restaurant])


    class NationalCulture(Topic): pass  # TOPICO: NationalCulture
    class ArtsCentre(NationalCulture): pass
    class CommunityCentre(NationalCulture): pass
    class ConferenceCentre(NationalCulture): pass
    class EventsVenue(NationalCulture): pass
    class Fountain(NationalCulture): pass
    class SocialCentre(NationalCulture): pass
    class TownHall(NationalCulture): pass
    class PublicBath(NationalCulture): pass
    AllDisjoint(
        [ArtsCentre, CommunityCentre, ConferenceCentre, EventsVenue,
Fountain, SocialCentre, TownHall, PublicBath])


    class Activity(Topic): pass  # TOPICO: Activity
    class Bbq(Activity): pass
    class DiveCentre(Activity): pass
    class Hunting(Activity): pass
    class CaravanSite(Activity): pass
    class PicnicSite(Activity): pass
```

```
    class ThemePark(Activity): pass
    class WildernessHut(Activity): pass
    class Zoo(Activity): pass
    AllDisjoint([Bbq, DiveCentre, Hunting, CaravanSite, PicnicSite,
ThemePark, WildernessHut, Zoo])


    class Shopping(Topic): pass  # TOPICO: Shopping
    class MarketPlace(Shopping): pass
    class Alcohol(Shopping): pass
    class Bakery(Shopping): pass
    class Beverages(Shopping): pass
    class Butcher(Shopping): pass
    class Cheese(Shopping): pass
    class Chocolate(Shopping): pass
    class Coffee(Shopping): pass
    class Confectionery(Shopping): pass
    class Convinience(Shopping): pass
    class Deli(Shopping): pass
    class Dairy(Shopping): pass
    class FrozenFood(Shopping): pass
    class GreenGrocer(Shopping): pass
    class HealthFood(Shopping): pass
    class Pastry(Shopping): pass
    class SeaFood(Shopping): pass
    class Spices(Shopping): pass
    class Tea(Shopping): pass
    class Wine(Shopping): pass
    class Water(Shopping): pass
    class Boutique(Shopping): pass
    class Clothes(Shopping): pass
    class Jewelry(Shopping): pass
    class Leather(Shopping): pass
    class Shoes(Shopping): pass
    class Tailor(Shopping): pass
    class Beauty(Shopping): pass
    class Cosmetics(Shopping): pass
    class Hairdresser(Shopping): pass
    class Tattoo(Shopping): pass
    class Antiques(Shopping): pass
    class Art(Shopping): pass
    class Craft(Shopping): pass
    class Gift(Shopping): pass
    class Stationery(Shopping): pass
    class Pet(Shopping): pass
    class PetGrooming(Shopping): pass
    class Toys(Shopping): pass
    AllDisjoint(
        [MarketPlace, Alcohol, Bakery, Beverages, Butcher, Cheese,
Chocolate, Coffee, Confectionery, Convinience,
         Deli, Dairy, FrozenFood, GreenGrocer, HealthFood, Pastry,
SeaFood, Spices, Tea, Wine, Water, Boutique,
         Clothes, Jewelry, Leather, Shoes, Tailor, Beauty, Cosmetics,
Hairdresser, Tattoo, Antiques, Art, Craft,
         Gift, Stationery, Pet, PetGrooming, Toys])
```

```
    class Entertaintment(Topic): pass  # TOPICO: Entertaintment
    class Casino(Entertaintment): pass
    class Cinema(Entertaintment): pass
    class LoveHotel(Entertaintment): pass
    class NightClub(Entertaintment): pass
    class Planetarium(Entertaintment): pass
    class Theatre(Entertaintment): pass
    class SportsHall(Entertaintment): pass
    class Stadium(Entertaintment): pass
    class RecreationGround(Entertaintment): pass
    class AdultGamingCentre(Entertaintment): pass
    class AmusementArcade(Entertaintment): pass
    class BeachResort(Entertaintment): pass
    class BandStand(Entertaintment): pass
    class DogPark(Entertaintment): pass
    class CampSite(Entertaintment): pass
    class Fishing(Entertaintment): pass
    class FitnessCentre(Entertaintment): pass
    class MiniatureGolf(Entertaintment): pass
    class Playground(Entertaintment): pass
    class SportsCentre(Entertaintment): pass
    class SummerCamp(Entertaintment): pass
    class SwimmingPool(Entertaintment): pass
    class WaterPark(Entertaintment): pass
    class Aquarium(Entertaintment): pass
    AllDisjoint([Casino, Cinema, LoveHotel, NightClub, Planetarium,
Theatre, SportsHall, Stadium, RecreationGround,
                 AdultGamingCentre,
                 AmusementArcade, BeachResort, BandStand, DogPark,
CampSite, PicnicSite, Fishing, FitnessCentre,
                 MiniatureGolf, Playground,
                 SportsCentre, SummerCamp, SwimmingPool, WaterPark,
Aquarium])


    class NaturalLandscape(Topic): pass  # TOPICO: NaturalLandscape
    class AboriginalLand(NaturalLandscape): pass
    class NationalPark(NaturalLandscape): pass
    class VillageGreen(NaturalLandscape): pass
    class Garden(NaturalLandscape): pass
    class NatureReserve(NaturalLandscape): pass
    class Park(NaturalLandscape): pass
    class Pier(NaturalLandscape): pass
    class Bay(NaturalLandscape): pass
    class Beach(NaturalLandscape): pass
    class Geyser(NaturalLandscape): pass
    class Glacier(NaturalLandscape): pass
    class Isthmus(NaturalLandscape): pass
    class Reef(NaturalLandscape): pass
    class CaveEntrance(NaturalLandscape): pass
    class Cliff(NaturalLandscape): pass
    class Peak(NaturalLandscape): pass
    class River(NaturalLandscape): pass
    class Lake(NaturalLandscape): pass
    class Stream(NaturalLandscape): pass
```

```
    class WaterFall(NaturalLandscape): pass
    AllDisjoint([AboriginalLand, NationalPark, VillageGreen, Garden,
NatureReserve, Park, Pier, Bay, Beach,
                 Geyser, Glacier, Isthmus, Reef, CaveEntrance, Cliff,
Peak, River, Lake, Stream, WaterFall])


    class CulturalLandscape(Topic): pass  # TOPICO: CulturalLandscape
    class Monastery(CulturalLandscape): pass
    class Church(CulturalLandscape): pass
    class Mosque(CulturalLandscape): pass
    class Temple(CulturalLandscape): pass
    class Cathedral(CulturalLandscape): pass
    class Chapel(CulturalLandscape): pass
    class Synagogue(CulturalLandscape): pass
    class Bridge(CulturalLandscape): pass
    class Castle(CulturalLandscape): pass
    class Ruin(CulturalLandscape): pass
    class Aqueduct(CulturalLandscape): pass
    class ArchaeologicalSite(CulturalLandscape): pass
    class BattleField(CulturalLandscape): pass
    class HistoricBuilding(CulturalLandscape): pass
    class CastleWall(CulturalLandscape): pass
    class CityGate(CulturalLandscape): pass
    class CityWall(CulturalLandscape): pass
    class HistoricDistrict(CulturalLandscape): pass
    class HistoricalFarm(CulturalLandscape): pass
    class MilitaryFort(CulturalLandscape): pass
    class Locomotive(CulturalLandscape): pass
    class HistoricMansion(CulturalLandscape): pass
    class Memorial(CulturalLandscape): pass
    class Monument(CulturalLandscape): pass
    class RailwayCar(CulturalLandscape): pass
    class Ship(CulturalLandscape): pass
    class Tank(CulturalLandscape): pass
    class Tomb(CulturalLandscape): pass
    class Tower(CulturalLandscape): pass
    class Vehicle(CulturalLandscape): pass
    class Cemetery(CulturalLandscape): pass
    class Cross(CulturalLandscape): pass
    class Obelisk(CulturalLandscape): pass
    class WindMill(CulturalLandscape): pass
    class Town(CulturalLandscape): pass
    class Village(CulturalLandscape): pass
    class Square(CulturalLandscape): pass
    class ArtWork(CulturalLandscape): pass
    class Attraction(CulturalLandscape): pass
    class Gallery(CulturalLandscape): pass
    class Museum(CulturalLandscape): pass
    class Exhibition(CulturalLandscape): pass
    AllDisjoint([Monastery, Church, Mosque, Temple, Cathedral, Chapel,
Synagogue, Bridge, Castle, Ruin, Aqueduct,
                 ArchaeologicalSite,
                 BattleField, HistoricBuilding, CastleWall, CityGate,
CityWall, HistoricDistrict, HistoricalFarm,
                 MilitaryFort,
```

```
                Locomotive, HistoricMansion, Memorial, Monument,
RailwayCar, Ship, Tank, Tomb, Tower, Vehicle,
                Cemetery,
                Cross, Obelisk, WindMill, Town, Village, Square,
ArtWork, Attraction, Gallery, Museum, Exhibition])

    AllDisjoint([Accomodation, Services, Transportation, Food,
NationalCulture,
                Activity, Shopping, Entertaintment, NaturalLandscape,
CulturalLandscape])


    # Clases para gestionar el Contexto
    class Context(Thing): pass
    class Geographical(Context): pass
    class Inside(Geographical): pass
    class Outside(Geographical): pass
    class Service(Geographical): pass
    AllDisjoint([Inside, Outside, Service])

    class Temporal(Context): pass
    class Environmental(Context): pass
    AllDisjoint([Geographical, Temporal, Environmental])


    class KeyWord(Thing): pass

    AllDisjoint([GenericPreference, User, Record,
                Location, Schedule, Characteristic,
                Preference, EmotionalState, Companion,
                TransportationPref, Topic, Context, KeyWord])


    # Relaciones Binarias entre User y GenericPreference:
    class hasGenericPreference(User >> GenericPreference): pass
    class isGenericPreferenceOf(GenericPreference >> User):
        owl_inverse_property = hasGenericPreference

    # Relaciones Binarias entre User y Record:
    class hasRecord(User >> Record):
        pass
    class isRecordOf(Record >> User):
        owl_inverse_property = hasRecord

    # Relaciones Binarias entre Record y Location:
    class hasLocation(Record >> Location, FunctionalProperty): pass
    class isLocationOf(Location >> Record, FunctionalProperty):
        owl_inverse_property = hasLocation

    # Relaciones Binarias entre Location y Schedule:
    class hasSchedule(Location >> Schedule): pass
    class isScheduleOf(Schedule >> Location):
        owl_inverse_property = hasSchedule

    # Relaciones Binarias entre Location y Characteristic
    class hasCharacteristic(Location >> Characteristic,
```

```
FunctionalProperty): pass
    class isCharacteristicOf(Characteristic >> Location,
FunctionalProperty):
        owl_inverse_property = hasCharacteristic

    # Relaciones Binarias entre Characteristic y TransportationPref:
    class hasTransportation(Characteristic >> TransportationPref): pass
    class isTransportationOf(TransportationPref >> Characteristic):
        owl_inverse_property = hasTransportation

    # Relaciones Binarias entre Characteristic y Context:
    class hasContext(Characteristic >> Context): pass
    class isContextOf(Context >> Characteristic):
        owl_inverse_property = hasContext

    # Relaciones Binarias entre Characteristic y Topic:
    class hasTopic(Characteristic >> Topic): pass
    class isTopicOf(Topic >> Characteristic):
        owl inverse property = hasTopic

    # Relaciones Binarias entre User y Preference:
    class hasPreference(User >> Preference, FunctionalProperty): pass
    class isPreferenceOf(Preference >> User, FunctionalProperty):
        owl_inverse_property = hasPreference

    # Relaciones Binarias entre Preference y EmotionalState:
    class hasEmotionalState(Preference >> EmotionalState): pass
    class isEmotionalStateOf(EmotionalState >> Preference):
        owl_inverse_property = hasEmotionalState

    # Relaciones Binarias entre Preference y Companion:
    class hasCompanion(Preference >> Companion): pass
    class isCompanionOf(Companion >> Preference):
        owl_inverse_property = hasCompanion

    # Relaciones Binarias entre Preference y TransportationPref:
    class prefHasTransportation(Preference >> TransportationPref): pass
    class isTransportationOfPref(TransportationPref >> Preference):
        owl_inverse_property = prefHasTransportation

    # Relaciones Binarias para ver el tipo de transporte del usuario:
    class typeOfTransportation(Preference >> TransportationPref): pass
    class isTypeOfTransportationOf(TransportationPref >> Preference):
        owl_inverse_property = typeOfTransportation

    # Relaciones Binarias entre Preference y Context:
    class prefHasContext(Preference >> Context): pass
    class isContextOfPref(Context >> Preference):
        owl_inverse_property = prefHasContext

    # Relaciones Binarias entre Preference y Topic:
    class prefHasTopic(Preference >> Topic): pass
    class isTopicOfPref(Topic >> Preference):
        owl_inverse_property = prefHasTopic

    # Relaciones Binarias entre OutsideLocation y InsideLocation
```

```
    class isInsideOf(InsideLocation >> OutsideLocation): pass
    class isConteinerOf(OutsideLocation >> InsideLocation):
        owl_inverse_property = isInsideOf

    # Relaciones Binarias entre OutsideLocation y ServiceLocation
    class isInsideOfServiceLocation(ServiceLocation >> OutsideLocation):
pass
    class isConteinerOfServiceLocation(OutsideLocation >>
ServiceLocation):
        owl_inverse_property = isInsideOfServiceLocation

    # Propiedades de GenericPreference:
    class departure_time(GenericPreference >> datetime.time,
FunctionalProperty): pass
    class budget(GenericPreference >> float, FunctionalProperty): pass
    class arrival_time(GenericPreference >> datetime.time,
FunctionalProperty): pass
    class dateGenPref(GenericPreference >> datetime.date,
FunctionalProperty): pass
    AllDifferent([departure_time, budget, arrival_time, dateGenPref])

    # Propiedades de User
    class identity(User >> int, FunctionalProperty): pass
    class current_location(User >> Location, FunctionalProperty): pass
    class name_user(User >> str, FunctionalProperty): pass
    class gender(User >> str, FunctionalProperty): pass
    class age(User >> int, FunctionalProperty): pass
    class education(User >> str, FunctionalProperty): pass
    class income_level(User >> str, FunctionalProperty): pass
    # AllDifferent([identity, current_location, name_user, gender, age,
education, income_level])

    # Propiedades de Record
    class date_record(Record >> datetime.date, FunctionalProperty): pass
    # Propiedades de CheckIn:
    class time_checkin(CheckIn >> datetime.time, FunctionalProperty):
pass

    # Propiedades de PastLocation:
    class rating(PastLocation >> int): pass
    class content_review(PastLocation >> str): pass
    AllDifferent([rating, content_review])

    # Propiedades de Location:
    class identity_location(Location >> int, FunctionalProperty): pass
    class name_location(Location >> str, FunctionalProperty): pass
    class address(Location >> str, FunctionalProperty): pass
    class staff(Location >> str): pass
    class image(Location >> str, FunctionalProperty): pass
    AllDifferent([identity_location, name_location, address, staff,
                  image])

    # Relaciòn entre Location y keyWord: keywords
    class keywords(Location >> KeyWord): pass

    # Propiedades de Schedule:
```

```
    class date_schedule(Schedule >> datetime.date, FunctionalProperty):
pass
    class opening_time(Schedule >> datetime.time, FunctionalProperty):
pass
    class closing_time(Schedule >> datetime.time, FunctionalProperty):
pass
    class event(Schedule >> str, FunctionalProperty): pass
    class cost(Schedule >> float, FunctionalProperty): pass
    class underage(Schedule >> bool, FunctionalProperty): pass
    AllDifferent([date_schedule, opening_time, closing_time, event,
cost, underage])

    # Propiedades de TransportationPref:
    class required_time(TransportationPref >> float,
FunctionalProperty): pass
    class cost_transp(TransportationPref >> float, FunctionalProperty):
pass
    AllDifferent([required_time, cost_transp])


    # Propiedades de Temporal:
    class hora(Temporal >> datetime.time, FunctionalProperty): pass
    class fecha(Temporal >> datetime.date, FunctionalProperty): pass
    AllDifferent([hora, fecha])


    # Propiedades de Environmental:
    class weather(Environmental >> str, FunctionalProperty): pass
    class temperature(Environmental >> str, FunctionalProperty): pass
    class crowdedness(Environmental >> str, FunctionalProperty): pass
    class illumination(Environmental >> int, FunctionalProperty): pass
    class humidity(Environmental >> str, FunctionalProperty): pass
    class noise(Environmental >> str, FunctionalProperty): pass
    AllDifferent([weather, temperature, crowdedness, illumination,
humidity, noise])

    class RecommendedUserLoc(Location):
        pass

    # Relaciones Binarias entre User y RecommendedUserLoc:
    class hasRecommendedLoc(User >> Location):
        pass
    class isRecommendedLocOf(Location >> User):
        owl_inverse_property = hasRecommendedLoc


    class Test(Thing):
        pass

    tourist.save("tourist.owl")
```

Bibliografía

Abbasi-Moud, Z., Vahdat-Nejad, H., & Sadri, J. (2021). Tourism recommendation system based on semantic clustering and sentiment analysis. *Expert Systems with Applications*, *167*, 114324. https://doi.org/10.1016/j.eswa.2020.114324

Abowd, G. D., Ebling, M., Hunt, G., Hui, L., & Gellersen, H. W. (2002). *Context-aware computing* (Vol. 1, Número 3, p. 23). https://doi.org/10.1109/MPRV.2002.1037718

Alaei, A. R., Becken, S., & Stantic, B. (2019). Sentiment Analysis in Tourism: Capitalizing on Big Data. *Journal of Travel Research*, *58*(2), 175–191. https://doi.org/10.1177/0047287517747753

Aldayel, M., Al-Nafjan, A., Al-Nuwaiser, W. M., Alrehaili, G., & Alyahya, G. (2023). Collaborative Filtering-Based Recommendation Systems for Touristic Businesses, Attractions, and Destinations. *Electronics*, *12*(19), 4047. https://doi.org/10.3390/electronics12194047

Alkhafaji, A., Fallahkhair, S., & Haig, E. (2020). A theoretical framework for designing smart and ubiquitous learning environments for outdoor cultural heritage. *Journal of Cultural Heritage*, *46*, 244–258. https://doi.org/10.1016/j.culher.2020.08.006

Ansari, S. A. (s/f). *Building a recomendation engine with Scala* (p. 2).

Bickerton, E. (2017). Out of context. *Apollo*, *2017-July*(July-August), 58–62. https://doi.org/10.5840/philtheol20186792

Brusilovsky, P. (1996). Methods and techniques of adaptive hypermedia. *User Modeling and User-Adapted Interaction*, *6*(2–3), 87–129. https://doi.org/10.1007/BF00143964

Buhalis, D., & Law, R. (2008). Progress in information technology and tourism management: 20 years on and 10 years after the Internet—The state of eTourism research. *Tourism Management*, *29*(4), 609–623. https://doi.org/10.1016/j.tourman.2008.01.005

Cena, F., Likavec, S., & Rapp, A. (2019). Real World User Model: Evolution of User Modeling Triggered by Advances in Wearable and Ubiquitous Computing: State of the Art and Future Directions. *Information Systems Frontiers*, *21*(5), 1085–1110. https://doi.org/10.1007/s10796-017-9818-3

Chalmers, M. (2004). A historical view of context. *Computer Supported Cooperative Work: CSCW: An International Journal*, *13*(3–4), 223–247. https://doi.org/10.1007/s10606-004-2802-8

Chang, G., Healey, M. J., McHugh, J. A. M., & Wang, J. T. L. (2001). *Web Mining*. *43*(8), 93–104. https://doi.org/10.1007/978-1-4615-1639-2_7

Chen, C.-C., & Tsai, J.-L. (2019). Determinants of behavioral intention to use the Personalized Location-based Mobile Tourism Application: An empirical study by integrating TAM with ISSM. *Future Generation Computer Systems*, *96*, 628–638. https://doi.org/10.1016/j.future.2017.02.028

Dey, Akd., A. (2001). Understanding and using context. Personal and Ubiquitous Computing. Retrieved from http://dl. acm. org/citation. cfm?id=593572. (2001). Understanding and using context. *Personal and ubiquitous computing*, 4–7. https://doi.org/10.1016/j.healthplace.2012.01.006

Dourish, P., & Damasceno, C. S. (2016). Ubiquitous computing. *Dialogues on Mobile Communication*, *804*, 67–86. https://doi.org/10.4324/9781315534619

Esmaeili, L., Mardani, S., Golpayegani, S. A. H., & Madar, Z. Z. (2020). A novel tourism recommender system in the context of social commerce. *Expert Systems with Applications*, *149*, 113301. https://doi.org/10.1016/j.eswa.2020.113301

Etaiwi, W., & Naymat, G. (2017). The Impact of applying Different Preprocessing Steps on Review Spam Detection. *Procedia Computer Science*, *113*, 273–279. https://doi.org/10.1016/j.procs.2017.08.368

Feng, L., Apers, P. M. G., & Jonker, W. (2004). Towards context-aware data management for ambient intelligence. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *3180*, 422–431. https://doi.org/10.1007/978-3-540-30075-5_41

Ghafoori, H. R., Sadeghi-Niaraki, A., Alesheikh, A. A., & Choi, S.-M. (2022). Ubiquitous GIS based outdoor evacuation assistance: An effective response to earthquake disasters. *International Journal of Disaster Risk Reduction*, *81*, 103232. https://doi.org/10.1016/j.ijdrr.2022.103232

Hodson, T. O. (2022). Root-mean-square error (RMSE) or mean absolute error (MAE): When to use them or not. *Geoscientific Model Development*, *15*(14), 5481–5487. https://doi.org/10.5194/gmd-15-5481-2022

Höpken, W., Fuchs, M., Zanker, M., & Beer, T. (2010). Context-Based Adaptation of Mobile Applications in Tourism. *Information Technology & Tourism*, *12*(2), 175–195. https://doi.org/10.3727/109830510X12887971002783

Hu, Y., Gao, S., Janowicz, K., Yu, B., Li, W., & Prasad, S. (2015). Extracting and understanding urban areas of interest using geotagged photos. *Computers, Environment and Urban Systems*, *54*, 240–254. https://doi.org/10.1016/j.compenvurbsys.2015.09.001

Islam, M. F., & Fonzone, A. (2021). Bus passenger path choices after consulting ubiquitous real-time information. *Travel Behaviour and Society*, *23*(August 2020), 226–239. https://doi.org/10.1016/j.tbs.2021.01.001

Jeon, N. J., Leem, C. S., Kim, M. H., & Shin, H. G. (2007). A taxonomy of ubiquitous computing applications. *Wireless Personal Communications*, *43*(4), 1229–1239. https://doi.org/10.1007/s11277-007-9297-9

Jiao, X., Xiao, Y., Zheng, W., Wang, H., & Hsu, C. H. (2019). A novel next new point-of-interest recommendation system based on simulated user travel decision-making process. *Future Generation Computer Systems*, *100*, 982–993. https://doi.org/10.1016/j.future.2019.05.065

Kano, Y., & Nakajima, T. (2018). International Journal of Pervasive Computing and Communications Article information : *International Journal of Pervasive Computing and Communications*, *14*(1), 15–32.

Li, J., Yang, Y., Gong, X., Jiang, J., Lu, Y., Lu, J., & Xie, S. (2023). Point-of-Interest Recommendations Based on Immediate User Preferences and Contextual Influences. *Electronics*, *12*(20), 4199. https://doi.org/10.3390/electronics12204199

Liang, Z. (2022). Context-Aware Sleep Health Recommender Systems (CASHRS): A Narrative Review. *Electronics*, *11*(20), 3384. https://doi.org/10.3390/electronics11203384

Madeira, R. N. (2012). Personalization in pervasive spaces towards smart interactions design. *2012 IEEE International Conference on Pervasive Computing and Communications Workshops, PERCOM Workshops 2012*, *March*, 548–549. https://doi.org/10.1109/PerComW.2012.6197568

Mcdonald, D. W. (s/f). *Systems*. 111–112.

McKenzie, G., Janowicz, K., Gao, S., & Gong, L. (2015). How where is when? On the regional variability and resolution of geosocial temporal signatures for points of interest. *Computers, Environment and Urban Systems*, *54*, 336–346. https://doi.org/10.1016/j.compenvurbsys.2015.10.002

Palomino, P. T., Toda, A. M., Rodrigues, L., Oliveira, W., Nacke, L., & Isotani, S. (2022). An ontology for modelling user' profiles and activities in gamified education. *Research and Practice in Technology Enhanced Learning*, *18*, 018. https://doi.org/10.58459/rptel.2023.18018

Park, H., Kwon, S., & Kwon, H.-C. (2009). Ontology-based Approach to Intelligent Ubiquitous Tourist Information System. *Proceedings of the 4th International Conference on Ubiquitous Information Technologies & Applications*, 1–6. https://doi.org/10.1109/ICUT.2009.5405697

Potonniée, O. (2002). Ubiquitous Personalization: A Smart Card Based Approach. *Proc. of 4th Gemplus Developer Conference*.

Raza, S., & Ding, C. (2019). Progress in context-aware recommender systems—An overview. *Computer Science Review*, *31*, 84–97. https://doi.org/10.1016/j.cosrev.2019.01.001

Restrepo Medina, S. E. (2012). Modelo de Inteligencia Ambiental basado en la integración de Redes de Sensores Inalámbricas y Agentes Inteligentes. *Bdigital.Unal.Edu.Co*, 126–126.

Salur, M. U., Aydin, I., & Alghrsi, S. A. (2019). SmartSenti: A Twitter-Based Sentiment Analysis System for the Smart Tourism in Turkey. *2019 International Artificial Intelligence and Data Processing Symposium (IDAP)*, 1–5. https://doi.org/10.1109/IDAP.2019.8875922

Santos, V. (2013). Use of Social Paradigms in Mobile Context-aware Computing. *Procedia Technology*, *9*, 100–113. https://doi.org/10.1016/j.protcy.2013.12.011

Schürholz, D., Kubler, S., & Zaslavsky, A. (2020). Artificial intelligence-enabled context-aware air quality prediction for smart cities. *Journal of Cleaner Production*, *271*. https://doi.org/10.1016/j.jclepro.2020.121941

Villegas, N. M., Sánchez, C., Díaz-Cely, J., & Tamura, G. (2018). Characterizing context-aware recommender systems: A systematic literature review. *Knowledge-Based Systems*, *140*, 173–200. https://doi.org/10.1016/j.knosys.2017.11.003

Zhang, B., Yin, C., David, B., Xiong, Z., & Niu, W. (2016). Facilitating professionals' work-based learning with context-aware mobile system. *Science of Computer Programming*, *129*, 3–19. https://doi.org/10.1016/j.scico.2016.01.008

Zheng, W., Liao, Z., & Lin, Z. (2020). Navigating through the complex transport system: A heuristic approach for city tourism recommendation. *Tourism Management*, *81*, 104162. https://doi.org/10.1016/j.tourman.2020.104162