FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Towards a Comprehensive Tourism Recommender System

Igor Liberato de Castro

Mestrado em Engenharia de Software

Supervisor: Prof. Rosaldo José Fernandes Rossetti

September 23, 2025

Towards a Comprehensive Tourism Recommender System

Igor Liberato de Castro

Mestrado em Engenharia de Software

September 23, 2025

Resumo

O rápido crescimento da indústria do turismo resultou numa quantidade sufocante de opções para turistas ao planear viagens, criando uma grande necessidade de sistemas inteligentes capazes de auxiliar na criação de itinerários turísticos personalizados e realistas. Embora Sistemas de Recomendação Turística (TRS) existentes consigam sugerir atrações individuais, eles raramente abordam a recomendação de roteiros completos, e, quando o fazem, não consideram a complexidade de restrições e preferências que caracterizam cenários reais de planeamento turístico. Esta dissertação responde à lacuna através do desenvolvimento e validação de um TRS abrangente para a geração de roteiros realistas e personalizados para viagens de vários dias.

A solução proposta adota uma arquitetura de duas camadas, combinando a previsão *data-driven* de avaliações pessoais de Pontos de Interesse (POI) com a otimização de itinerários, sujeita a restrições. O sistema tem em conta características demográficas do utilizador bem como as suas preferências quanto a cinco dimensões de caracterização dos POIs (História, Natureza, Arte, Cultura e Lazer), familiaridade com o destino e ritmo de visita. Os dados dos POIs são importados automaticamente do OpenStreetMap e enriquecidos por um Modelo de Linguagem de Larga Escala (LLM) com estimativas das suas características. As preferências dos utilizadores foram recolhidas através de um inquérito online e usadas para treinar modelos de filtragem híbrida para prever opiniões sobre POIs.

Para a geração dos roteiros, o sistema emprega um algoritmo inovador que combina *Constrained K-Means clustering* e otimização com *Branch-and-Bound* iterativo, formulando o problema como um *Orienteering Problem* de Equipa com Janelas Temporais (TOPTW). Essa abordagem agrupa os POIs candidatos em *clusters* diários geograficamente coerentes e determina sequências ótimas de visita para cada *cluster* que respeitam horários de abertura de POIs, tempos de deslocação e restrições temporais definidas pelo utilizador.

O sistema foi validado com dados reais de três grandes cidades europeias: Porto, Lisboa e Paris. Durante a avaliação, múltiplas abordagens algorítmicas foram testadas com dados recolhidos através de um inquérito que envolveu 184 participantes, fornecendo um total de 8.667 avaliações de POIs. Os resultados demonstram que o sistema proposto supera consistentemente a referência (*baseline*), gerando planos de melhor qualidade sem comprometer a completude.

Esta dissertação contribui para a área com um modelo matemático para a avaliação de itinerários turísticos, uma definição formal de uma versão do Problema do Planeamento de Viagens Turísticas (TTDP), uma solução algorítmica inovadora para o problema e *datasets* turísticos abrangentes. Os resultados validam a hipótese de que a incorporação de preferências e restrições temporais dos utilizadores proporciona uma melhoria significativa na qualidade e utilidade prática das recomendações, avançando, portanto, o estado da arte em sistemas de planeamento turístico personalizado.

Abstract

The tourism industry's rapid growth has created an overwhelming array of choices for travelers when planning trips, leading to a significant need for intelligent systems that can assist in creating personalized and feasible travel itineraries. While existing Tourism Recommender Systems (TRS) can suggest individual attractions, they rarely address the recommendation of complete itineraries, and when they do, typically fail to integrate the complex constraints and preferences that characterize real travel planning scenarios. This dissertation addresses the gap by developing and evaluating a comprehensive TRS capable of generating realistic and personalized multi-day travel plans.

The proposed solution adopts a two-layered architecture that combines data-driven personalized rating prediction with constraint-aware itinerary optimization. The system takes into account user demographics, preferences regarding five POI feature dimensions (History, Nature, Art, Culture, and Leisure), familiarity with destinations, and individual pacing preferences. POI data is automatically imported from OpenStreetMap and enriched using Large Language Model-based feature estimation. User preferences are captured through a custom web survey and used to train hybrid filtering models to predict personalized POI ratings.

For itinerary generation, the system employs a novel algorithm combining Constrained K-Means clustering with Iterative Branch-and-Bound optimization, formulating the problem as a Team Orienteering Problem with Time Windows (TOPTW). This approach partitions candidate POIs into geographically coherent daily clusters and determines optimal visit sequences for each cluster which respect POI opening hours, travel times, and user-defined temporal constraints.

The system was validated using real-world data from three major European cities: Porto, Lisbon, and Paris. During evaluation multiple algorithmic approaches were tested with data collected through a survey involving 184 participants, who provided 8,667 POI ratings. Results demonstrate that the proposed system consistently outperforms baseline methods, achieving superior itinerary quality without compromising completeness.

This dissertation contributes to the field with a mathematical framework for evaluating tourism itineraries, a formal definition of a version of the Tourist Trip Design Problem (TTDP), an innovative algorithmic solution to the problem and comprehensive tourism datasets. The findings validate the hypothesis that incorporating user preferences and temporal constraints significantly improves recommendation quality and practical feasibility, advancing the state of the art in personalized tourism planning systems.

Acknowledgements

First and foremost I must thank those without whom I literally could not have existed: My parents. To my dad, a formidable systems analyst with more years of experience than I have on earth, for guiding me through this often confusing field, as well as life itself, and giving me unconditional love and support on my journey. And to my mom, the best role model one could ever wish for on how to live an authentic life, who gave me the strength to continue even in my darkest moments. Love you both from the bottom of my heart.

I would also like to thank my advisor, Rosaldo Rossetti, who took a chance on me by approving my crazy self-authored thesis proposal, an action for which I will forever be grateful and hope to prove it by my work. His vast expertise was also vital in bringing this project to fruition.

Likewise, I could never be where I am without my amazing group of friends, who have always encouraged me to pursue my dream of giving life to my project. To all of you I give my sincerest thanks.

Praise goes also to the beautiful city of Sevilla and to my wonderful cousin Carina, who received me there with open arms. It was the observations I drew from my trip to this gorgeous city in 2022 that spawned the idea for this project.

In the same vein, I want to show my appreciation for all researchers who have done work in the field of Tourism Recommender Systems before me, you are the giants on whose shoulders I stand. Special praise goes to my colleagues at the University of Vilnius in Lithuania, Remigijus Paulavičius, Linas Stripinis, Simona Sutavičiūtė, Dmitrij Kočegarov and Ernestas Filatovas and my Tunisian colleagues Takwa Tlili and Saoussen Krichen, authors of [30] and [45] respectively, the two most influential papers on the design of my solution. Though you don't know me, your work has been invaluable in the development of this project and for this I extend my sincerest regards.

Lastly, I would be a fool not to acknowledge two special organizations which were indispensable for this study. Firstly, to the Student Branch of IEEE at the University of Porto, for providing me with a community of engineers and friends, as well as the physical space in which this dissertation was primarily developed. And secondly, but no less importantly, to OpenStreetMap, without whose free and open geographic data this project could never have been realized.

Igor Liberato

"The world is a book, and those who do not travel read only one page"

St. Augustine

Contents

List of Figures

List of Tables

Listings

List of Acronyms

BB	*Branch-and-Bound*
CBF	*Content-based Filtering*
CF	*Collaborative Filtering*
CKM	*Constrained K-Means*
CKMR	*Constrained K-Means with Refinement*
DG	*Dynamic Greedy*
HF	*Hybrid Filtering*
IBB	*Iterative Branch-and-Bound*
ILP	*Integer Linear Programming*
ILS	*Iterated Local Search*
KM	*K-Means*
LLM	*Large Language Model*
MAE	*Mean Absolute Error*
MAUT	*Multi-Attribute Utility Theory*
ML	*Machine Learning*
MSE	*Mean Squared Error*
NG	*Naive Greedy*
OP	*Orienteering Problem*
OPTW	*Orienteering Problem with Time Windows*
OQL	*Overpass Query Language*
OSM	*OpenStreetMap*
POI	*Point of Interest*
RMSE	*Root Mean Squared Error*
SQ	*Search Query*
SA	*Simulated Annealing*
SVD	*Singular Value Decomposition*
TOP	*Team Orienteering Problem*
TOPTW	*Team Orienteering Problem with Time Windows*
TRS	*Tourism Recommender System*
TSP	*Traveling Salesman Problem*
TTDP	*Tourist Trip Design Problem*

Chapter 1

Introduction

1.1 Context and motivation

The tourism industry has experienced unprecedented growth over the past decades, becoming one of the world's largest economic sectors. As travel becomes increasingly accessible and destinations more diverse, tourists are confronted with an ever more overwhelming array of choices when planning their trips, exponentially growing the challenge of selecting appropriate Points of Interest (POIs) and organizing them into feasible, satisfying itineraries. This complexity has created a significant opportunity for intelligent systems that can assist travelers in making informed decisions while maximizing their satisfaction and trip efficiency.

Traditional travel planning methods, whether through guidebooks, travel agents, or generic online platforms, often fail to account for the highly personalized nature of tourist preferences and the intricate spatio-temporal constraints that govern real-world travel scenarios. Tourists differ significantly in their preferences for historical sites, natural attractions, cultural experiences, artistic venues, and leisure activities. Furthermore, practical considerations such as opening hours, travel times between locations, daily time budgets, and individual pacing preferences add layers of complexity that generic recommendations cannot fully address.

While the emergence of Tourism Recommender Systems (TRS) offers a promising response to the growing complexity of travel planning, current implementations fall short of addressing the real challenges involved. Existing platforms are usually able to recommend individual POIs based on popularity or basic user ratings, but they rarely account for the multifaceted nature of tourist preferences or the spatio-temporal constraints that shape real-world itineraries. As a result, their recommendations often lack practical feasibility and coherence to be actually usable in practice. This gap is the main motivation for the present research, which seeks to move beyond isolated POI suggestions toward the generation of complete, personalized multi-day travel plans.

1.2 Goals

The primary goal of this project is to develop a comprehensive TRS capable of generating realistic and personalized multi-day travel itineraries by considering rich user preferences and conforming to robust spatio-temporal constraints. Achieving this goal requires addressing two complementary challenges: accurately predicting the degree of interest a unique user would have in each available POI, and organizing the most suitable POIs into efficient and usable daily routes. To this end, the project employs a two-layered architecture inspired by state-of-the-art research, that integrates a data-driven module for personalized rating prediction, with a heuristic itinerary generation module. Because this project is concerned with the realism and real-world usability of the generated itineraries, an essential aspect involves validating results using real user data and assessing their subjective quality.

1.3 Contributions

This dissertation contributes to the field of TRS design by proposing and validating a complete system capable of generating personalized, realistic and comprehensive multi-day itineraries. The work advances the state of the art in three main ways:

- It implements a two-layer recommendation architecture that combines personalized POI selection with constraint-aware itinerary planning.
- It introduces a formal evaluation framework for generated itineraries and uses it to compare the performance of a diverse set of optimization algorithms.
- It delivers curated datasets, detailed analyses and new methodological insights that together provide a foundation for future research on personalized tourism planning.

1.4 Document structure

The remainder of this document is structured into five further chapters. Chapter 2 explains the background theoretical knowledge that grounds this study and explores related work in the field of TRS design and associated fields. Chapter 3 outlines the methodological approach, detailing the research question, hypotheses, evaluation metrics, problem formalization, and overall system design. Chapter 4 explains the implementation details of each of the system's modules. Chapter 5 reports on the results of validating each module and analyzes them with a critical lens. Finally, Chapter 6 summarizes the main contributions of the study and suggests directions for future research.

Chapter 2

Literature review

This chapter summarizes and discusses previous work done in the field of tourism recommender system design, with a particular focus on systems which integrate a high degree of personalization or rich spatio-temporal constraints. Section 2.1 explains some important background concepts for understanding the study as a whole. Section 2.2 outlines the protocol used to select articles for inclusion in this review. Section 2.3 explores tourist typologies and preferences, the factors that influence them, and the relationship between touristic preferences and satisfaction in travelers. Section 2.4 gives an overview of the different ways researchers have conceptualized the problem of automating touristic recommendations and their diverse approaches to solve it. This section is further divided into two subsections based on the type of method proposed, either data-driven or heuristic. Finally, section 2.5 summarizes the literature review and discusses the current state of the art in TRS design. It also presents a tabular classification of the solutions explored.

2.1 Background

2.1.1 Core concepts and entities

- **Point of Interest (POI):** A real-life visitable attraction such as a museum, a park, or a landmark, with a name, defined geographic location, category, and opening hours.
- **Itinerary / Route / Schedule:** When referring to the output of the proposed solution, which takes into account both the spatial distribution of POIs (characteristic of a route) and the commute and visitation times of POIs (characteristic of a schedule), these terms will be used interchangeably to mean an ordered, time-feasible sequence of POIs to visit for one or more days of a trip.
- **Tourism Recommender System (TRS):** A system that assists travelers in selecting POIs and organizing them into usable itineraries, aiming to maximize alignment with user preferences while satisfying spatio-temporal constraints such as opening hours, travel times, and daily time budgets.

- **Tourist Trip Design Problem (TTDP):** The computational optimization problem underlying a TRS. It consists of selecting and sequencing a subset of candidate POIs into one or more daily routes such that:
  - The total utility of the chosen POIs is maximized.
  - All spatio-temporal constraints are satisfied.

  The TTDP is inherently NP-hard and is often modelled using well-established combinatorial optimization templates such as the Travelling Salesman Problem (TSP) or Orienteering Problems (OPs).

2.1.2 Method families

The literature on designing a TRS largely clusters into two complementary families, which structure the review in Section 2.4:

- **Data-driven methods:** Focus on estimating what the user is likely to enjoy by analyzing patterns in existing data. Often rely on predictive models to estimate the ratings a user would give to candidate POIs. They are further divided into three approaches:
  - **Collaborative Filtering (CF):** Infers preferences from behavioral similarity between users (e.g., ratings, check-ins, past routes).
  - **Content-Based Filtering (CBF):** Uses item attributes (e.g., POI category, tags, features) to recommend similar items to those the user already likes.
  - **Hybrid Filtering (HF):** Combines CF and CBF, often by concatenating latent factors from CF with explicit features in a regression or neural model.
- **Heuristic/optimization methods:** Focus on finding a feasible and efficient route that maximizes total utility under real-world constraints. Models the TTDP as an optimization problem, with the following well-known frameworks being the most popular:
  - **Travelling Salesman Problem (TSP):** A routing problem where a traveler must visit all locations in a set exactly once while minimizing total travel cost or distance.
  - **Orienteering Problem (OP):** A routing problem where a traveler must visit a subset of locations, each with its own reward value and service time, aiming to collect the highest possible total reward within a limited time budget. It and its many variations are widely used in TRS because they directly address the balancing act between POI value and visit duration.
  - **Team Orienteering with Time Windows (TOPTW):** An extension of the OP that introduces:
    - **Multiple routes (team members):** Representing the different days of a trip.

- **Time windows:** Each location (POI) has specific opening and closing times and visits must occur within these.

TOPTW is currently considered the most faithful formulation of the real-world itinerary planning problem faced by TRS.

Because data-driven methods focus on predicting user preferences while heuristic methods focus on enforcing spatio-temporal feasibility, many modern TRS adopt a two-layer architecture that first predicts personalized POI scores and then schedules the highest-scoring POIs into feasible itineraries.

2.1.3 Machine Learning techniques

In addition to the broader method families, two core machine learning techniques were widely applied in this work and as such are crucial to understand:

- **Clustering:** An unsupervised machine learning technique that groups data points based on their similarity, without requiring labeled targets. In TRS and other geographically bound systems, it is often used to partition geolocated items into spatially coherent groups
- **Regression:** A supervised machine learning technique where a model learns to predict a continuous numerical target variable from a set of input features, using labeled data for training. In TRS, regression models are typically used to estimate user ratings of POIs from user profile data and POI characteristics.

2.2 Review protocol

2.2.1 Research question and search query

The scope of this review is research on tourism recommender systems, with a particular focus on recent developments and solutions that integrate rich time and space restrictions or personalize recommendations based on user preferences. In other words, it seeks to answer the following Research Question (RQ): “How can a system recommend complete and high-quality touristic itineraries for users that conform to user preferences and travel time restrictions?". From the RQ, the Search Query (SQ) in Listing 2.1 was devised.

Listing 2.1: Search query.

```
1  T1 (tour* OR travel* OR poi OR pois) AND
2  T2 recommend* AND
3  T3 (schedul* OR rout*) AND
4  T4 (personal* OR preference*) AND
5  T5 time*
```

The first term of the SQ establishes the application domain of tourism and the objects of interest, touristic POIs. The second term specifies recommendation algorithms as the topic of search. The third term determines the intended output of the system: Itineraries that are either time-restricted (schedule) or space-restricted (route). Finally, the fourth and fifth terms narrow the scope of the search to studies that address personalized and time-restricted recommendations, ensuring alignment with the RQ.

2.2.2 Paper selection

To identify and select studies for inclusion in this review, the method outlined in Figure 2.1 was followed. This method was designed following the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) guidelines[1] as a template. It consists of three distinct phases, the Preliminary phase, Phase 1 and Phase 2.

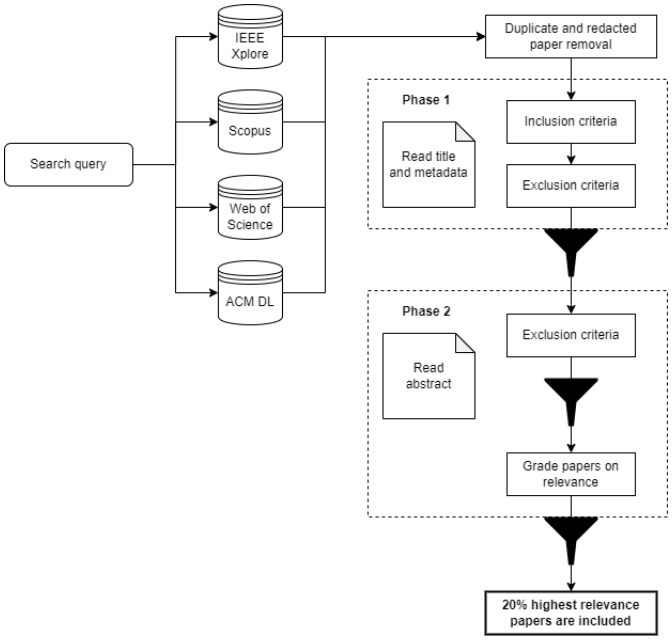

Figure 2.1: Paper selection method.

2.2.2.1 Preliminary phase

During the preliminary phase, the SQ presented in Listing 2.1 was submitted to four scientific databases: ACM Digital Library[2], IEEE Xplore[3], Scopus[4], and Web of Science[5]. To better capture the most recent developments in the field, the search was restricted exclusively to studies published between 2019 and 2024. A total of 496 works were retrieved across the four databases.

[1] https://www.prisma-statement.org
[2] https://dl.acm.org
[3] https://ieeexplore.ieee.org
[4] https://www.scopus.com
[5] https://www.webofscience.com

After compiling the results, 186 duplicate entries and 6 redacted papers were removed, leaving 304 unique and valid studies for further analysis.

2.2.2.2 Phase 1

In this phase, the titles of the remaining papers were read, their abstracts were briefly skimmed, and their metadata were analyzed. Based on the information acquired, the inclusion and exclusion criteria listed below were applied. The number of studies eliminated by each criterion is provided in parentheses.

- Inclusion criteria:
  1. Study type must be conference paper, journal article, or review *(Eliminated 18 studies)*.
  2. Study must be at least 6 pages long *(Eliminated 36 studies)*.
  3. Title and brief skim of the abstract indicate that:
     - Study is about tourism.
     - Study relates to the realm of computer science or engineering.
     - Study deals with recommending POIs or modeling touristic itineraries.

     (Eliminated 137 studies).
- Exclusion criteria:
  1. Title and brief skim of the abstract indicate that the study's application domain is excessively specific *(Eliminated 8 studies)*.

2.2.2.3 Phase 2

In this phase, the full abstracts of the remaining 105 papers were carefully reviewed. Based on the information therein, the exclusion criteria listed below were applied. The number of studies eliminated by each criterion is provided in parentheses.

- Exclusion criteria:
  1. Abstract indicates that the study's application domain does not align with this review's objectives *(Eliminated 16 studies)*.
  2. Abstract is unintelligible *(Eliminated 3 studies)*.

Next, the Relevance of the remaining 86 works was assessed. To do this, each of them was graded on two distinct dimensions, namely, Prominence and Comprehensiveness, explained below.

1. **Prominence:** Reflects the relative prominence of a study in the field. It is calculated as the average of two metrics: Unified Impact and Unified Citations per Year.

(a) **Unified Impact (UI):** Measures the importance of the venue (journal or conference) where the work was published. It is represented by the Journal Impact Factor (JIF) of the journal or conference proceedings where the paper was published, as estimated by Clarivate. If no JIF was available, UI was set to 0.3, matching the lowest UI among all other studies.

(b) **Unified Citations per Year (UCY):** Measures the relative influence of a paper to subsequent research. For works published before 2024, this was calculated as the amount of times they were cited, divided by the number of years elapsed since publication. For studies published in 2024, a default value of 3.0 was established to mitigate the disadvantage they would otherwise incur due to the recency of publication. This default value corresponds to the mean score across all other papers.

Figure 2.2 shows the distribution of Prominence scores across the 86 studies.

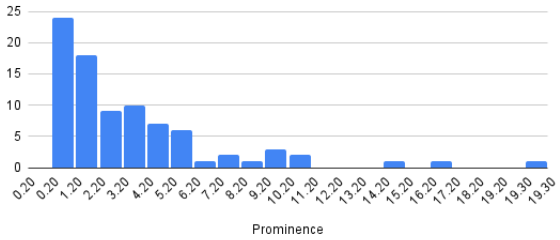

Figure 2.2: Distribution of Prominence.

2. **Comprehensiveness:** Reflects the number of key dimensions of restriction expected to be addressed in the work, based on the content of its abstract. The following five dimensions were identified as particularly pertinent and therefore considered for the Comprehensiveness score:

   D1. Personalization of recommendations based on users the individual tastes of users.

   D2. Consideration of geographic placement and distribution of POIs.

   D3. Factoring of commute time between POIs as well as POI visit durations.

   D4. Support for multi-day itinerary recommendations.

   D5. Development of methods to accurately assess the subjective fitness of a POI to a user.

   Each study was assigned one point for every dimension explicitly mentioned in the abstract. Table 2.1 presents the frequency with which each dimension was mentioned across the 86 works.

   The Comprehensiveness of a study is defined as the sum of points across the five dimensions. Figure 2.3 illustrates the distribution of Comprehensiveness scores for the 86 studies.

Table 2.1: Mentioning frequencies for the dimensions of restriction.

D1	D2	D3	D4	D5
52/86 (60%)	64/86 (75%)	21/86 (24%)	17/86 (20%)	15/86 (17%)

To determine the overall Relevance (R) of each paper, the scores for Prominence (P) and Comprehensiveness (C) were combined using the formula in 2.1.

$$R = \frac{P}{10} + C \tag{2.1}$$

Figure 2.4 shows the distribution of Relevance scores across the 86 works.

The top 20% of studies by Relevance, corresponding to 17 papers, with a minimum Relevance score of 3.42, were considered for this review.

2.2.2.4 Additional papers

In addition to the 17 papers selected for inclusion using the methodology defined in Section 2.2.2, an additional set of studies has been considered. These works fall into one of two categories, described below.

1. **Related informal research:** This category includes studies discovered through snowballing, which are concerned with the same problem outlined in the RQ. The following papers belong to this category:

   - **Supanich and Kulkarineetham [42] and Wang [50]:** Important works on the use of CF for TRSs.
   - **Bai, Pamula, and Jain [5]:** Employ HF to address the Cold Start Problem in TRS.
   - **Zhong et al. [55]:** One of the few works that integrates continual improvement in TRS.

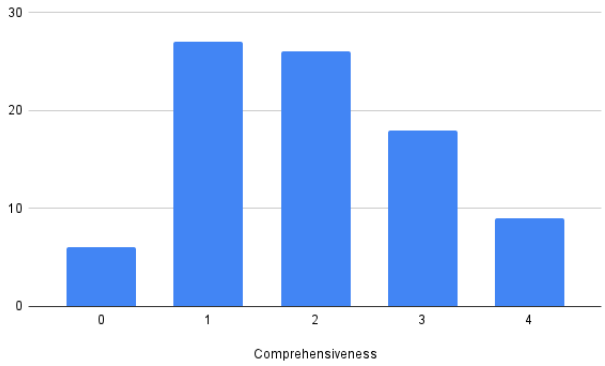

Figure 2.3: Distribution of Comprehensiveness.

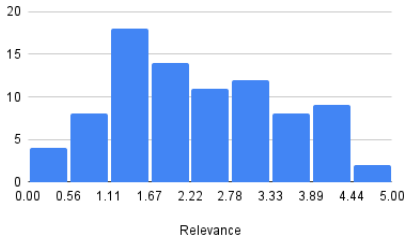

Figure 2.4: Distribution of Relevance.

- **Mahardika and Baizal [28]:** Innovative in their use of SA to solve the TTDP.

2. **Tourist typologies:** This category includes works that attempt to answer an important dependent question of the RQ: “What aspects of users and POIs determine the enjoyment a person derives from visiting a POI?". The following studies belong to this category:

   - **Brida, Disegna, and Scuderi [10], Iseh et al. [20], Kutlu et al. [22], Poria, Butler, and Airey [32], Richards [37], and Vengesayi, Mavondo, and Reisinger [48]:** Examine the aspects of POIs that give them value in the eyes of tourists.
   - **Alén, Losada, and Domínguez Vila [1], Andruliene, Macerinskiene, and Urbonavicius [4], Debski [15], and Meng and Uysal [29]:** Investigate the influence of demographic factors in tourist preferences.
   - **Zhang et al. [54]**: Explore the relationship between personal values and tourist preferences.
   - **Cena et al. [12]**: Probe into the connection between personality and tourist preferences.

2.3 Tourist typologies

This review is particularly focused on TRS with a high degree of personalization. To achieve personalization, it is crucial to understand which features of a POI are considered desirable by different types of tourists and which aspects determine each tourist’s individual preferences.

2.3.1 POI features

Many studies have dealt with the question of what aspects of a POI determine its desirability for tourists. Vengesayi et al. [48] for example, considered six overlapping attraction types as possible draw factors for tourists visiting Zimbabwe. According to their findings, "Unique" and "Man-made" were the most desirable categories, with "Historical" and "Natural" also playing a part. Interestingly, "Recreation" seemed to be a significant negative factor, though the authors attribute this to the underdeveloped nature of recreational tourism in the country. In Nigeria, a similar scenario exists according to Iseh et al. [20], who discovered cultural offerings and natural beauty

show a strong correlation with increased tourism.

Greg Richards' extensive work on cultural tourism, such as his 2000 paper [37], has served to show the important place of local culture, as well as historical heritage, in attracting tourists. This is also supported by Kutlu et al. [22], whose findings suggest that a 1% increase in the number of UNESCO World Heritage Sites in a country (an important indicator of cultural and historical richness) increases international tourism by an average of 0.22%.

Poria et al. [32] applied factor analysis to interviews with tourists to discover three distinct clusters with different motivations and preferences. The first cluster was mostly interested in connecting with the cultural heritage of the attraction, the second, in acquiring knowledge, and the third was more concerned with leisure and fun.

Finally, Brida et al. [10] used bagged clustering to categorize visitors to two different museums, one archaeological and one artistic. Through this method, the authors were able to define art museum visitors as a unique group with distinct motivations and characteristics, illustrating preference for art as an important differentiating factor between tourist types.

2.3.2 Demographics and tourist preferences

Among the characteristics that influence tourist preferences is their demographic profile. M. Debski [15] examined nationality as a possible factor shaping tourist preferences for different attraction categories. Through a survey of 233 students in higher education from three different countries, Austria, Ukraine, and Poland, Debski identified both commonalities and differences among the national groups.

For similarities, it was found that trying out local foods and visiting historical monuments were always major interests, regardless of nationality. Relaxing on a beach, though particularly high-ranking for Poles, with 77% of them reporting affinity, was greatly enjoyed by all groups.

As for differences, Ukranians showed the greatest appetite for both museums and nightlife with 67% and 60% respectively reporting interest, significantly higher than other groups. As mentioned previously, Poles expressed a higher-than-average preference for beach relaxation, but also for wildlife exploration, with 41% interest. Austrians, by contrast, were the least enthusiastic about wildlife exploration, with only 20% of those surveyed reporting affinity, and were generally more inclined towards shopping for souvenirs with 64% interested.

Meng et al. [29] explored the impact of gender on POI category preference, using a similar survey and analysis method to Debski [15]. Their study found that women place higher value on scenic beauty and prefer relaxing, educational, or shopping-related activities. Men, by contrast, were shown to yearn for a sense of adventure and favor physically demanding activities and sports.

With regards to age, Alén et al. [1] discovered a positive correlation between seniority and a preference for leisurely and educational activities. These results were corroborated by Andruliene et al. [4], who additionally detected a negative relationship between age and enjoyment of entertainment-related activities like bars, nightclubs, or theaters and nature recreation such as hiking. Additionally, older tourists were significantly more likely to partake in luxury experiences and stay in higher-end accommodation [1].

2.3.3 Personality and tourist preferences

Beyond demographics, personality traits and personal values profoundly influence tourist preferences. To demonstrate the impact of values, Zhang et al. [54] administered structured laddering interviews, a qualitative research technique designed to uncover underlying cognitive structures. This approach facilitated the collection of data on participants' preferred activities, the benefits they associated with these, and the personal values influencing their perceptions. K-means clustering was performed to classify them into four different clusters based on their values. For each group, Hierarchical Value Maps were applied to visualize the relationships between values, benefits, and activities. Trough discriminant analysis and other techniques, the clusters were found to be meaningful and contain unique and interesting patterns of relation, reflecting distinct travel motivations and behaviors.

The first group, nicknamed Pleasure Seeking Travelers, has pleasure as their main motivation, and so prefer entertainment and nightlife activities at higher rates. Cluster two, the Social Travelers, are driven by a desire to share their journeys online, and are therefore inclined towards more social and cultural attractions, as well as breathtaking, photo-worthy views. Knowledge and discovery are the motivating factors for group three, the Exploratory Travelers, who tend to enjoy local folk experiences, natural landscapes, and museums. Lastly, the Escape-Oriented Travelers seek a break from their daily lives for solitude and self reflection, they are the most nature-loving of the clusters and have a dislike of social activities.

- **Openness to experience:** *Open to new ideas and experiences* vs. *Resistant to change*.
- **Conscientiousness:** *Organized and responsible* vs. *Disorganized and careless*.
- **Extraversion:** *Sociable and assertive* vs. *Introverted and reserved*.
- **Agreeableness:** *Cooperative and compassionate* vs. *Competitive and inconsiderate*.
- **Emotional Stability:** *Calm and stable* vs. *Anxious and sensitive to stress*.

The itinerary characteristics considered were divided into three groups:

1. **POI-related**
    - **Variety:** Diversity in the types of POIs.
    - **Uniformity:** Preference for uniformity among POI categories.

- **Breadth:** Seeing many POIs.
- **Depth:** Spending substantial time at each POI.

2. **Time-related**

   - **Total available time:** Overall time budget allocated for the trip.
   - **Travel time:** Time spent commuting between POIs.
   - **Efficiency:** Efficient time allocation, considering distances, opening times, and peak hours.
   - **Free time:** Willingness to include unstructured time.
   - **Avoiding busy hours:** Preference for visiting attractions during less crowded times.

3. **Plan-related**

   - **Careful planning:** Creating detailed itineraries in advance.
   - **Unpremeditated choices:** Making spontaneous decisions.
   - **Autonomous decision making:** Planning without external help.
   - **Expert advice:** Relying on recommendations from experts.

A survey was conducted to collect data on participants' personality traits, as well as their itinerary preferences and data analysis techniques were applied to identify correlations between the two. Among many notable results, Openness was associated with a propensity towards efficiency and expert advice, Conscientiousness was connected with careful planning, efficient time use and total available time, Extraversion was negatively correlated with detailed planning and uniformity, and Emotional Stability was linked to unpremeditated choices.

The exploration of tourist typologies highlights the multifaceted nature of tourist behavior. History, Nature, Art, Culture and Leisure all stand out as important dimensions, subject to touristic preference [10, 20, 22, 32, 37, 48]. Demographic factors such as nationality, gender, and age have significant influence on not only preferences, but also leisure choices and spending tendencies [1, 4, 15, 29]. Simultaneously, personality traits and personal values reveal deeper motivational patterns for trip organization decisions [12, 54]. By understanding the part these factors play and taking them into account, future TRS can deliver more nuanced recommendations tailored to diverse tourist preferences, ultimately enhancing travel experiences.

2.4 Tourism Recommender System approaches

This section reviews the various methods explored by researchers between 2019 and 2024 to address the problem of creating a comprehensive TRS. These systems can broadly be categorized into data-driven and heuristic methods, depending on whether data patterns or heuristic algorithms primarily guide decision-making.

2.4.1 Data-driven methods

2.4.1.1 Simple recommendation algorithms

A significant amount of the data-driven systems explored are more concerned with the process of leveraging large volumes of data to calculate user preferences than itinerary planning. Therefore they often feature only simple recommendation algorithms to transform those preferences into POI recommendations.

For example, Tsai et al. [46] utilize data from Flickr to model and recommend tourist attractions. Their proposed method begins by applying location-based clustering of geotagged photos uploaded to the platform and then employing Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN) on the clusters to identify possible landmarks. To characterize the discovered landmarks, thematic topics are extracted from the hashtags associated with the photos through Latent Dirichlet Allocation (LDA) topic modeling.

The paper also proposes a Long Short-Term Memory (LSTM) prediction model to leverage the characterized POIs, alongside either the personal preferences of users or their previous visitation record, to recommend respectively an ideal personalized first attraction to visit or the most suitable next attraction in the visiting sequence. However, a limitation of the proposed algorithm is its focus on suggesting individual POIs rather than a complete itinerary.

Zhou et al. [56] take the concept further by proposing a data-driven algorithm capable of recommending full-day itineraries. The system extracts attributes from textual descriptions of POIs and clusters them based on attribute similarity. Users can specify the desired number of attractions to select from each cluster and the system responds by identifying the POIs within each group that best align with the user's preferences. To generate optimized routes, the algorithm accounts for travel time between POIs, accommodating three different modes of personal transport: walking, biking, and driving.

Liang et al. [25] introduce a robust algorithm designed to match users with their most suitable attractions. The process begins by breaking down POI reviews into component words and applying TextRank to calculate their relative importance. LDA topic modeling is then used to extract topics and tags that characterize both users and attractions. POI fitness scores, which are personalized for each user, are computed by combining user-attraction, user-user, and attraction-attraction similarity metrics, yielding a comprehensive and nuanced fitness score.

While the proposed algorithm is capable of recommending complete multi-day itineraries, this is done through a simple iterative selection of the highest-scoring POI available. This method does not account for the spatial proximity of attractions, which can result in impractical or inefficient travel plans.

A more sophisticated procedure for route recommendation is proposed by Zhang et al. [53]. Their process begins by constructing a matrix representing user visits to POIs. Due to the inherent

sparsity of the matrix, Funk Singular Value Decomposition (Funk-SVD) is employed to fill in the missing values. Next, Dual Bi-Directional LSTM Encoders combine the extrapolated user interests and spatial requirements into a complete user representation. This is then passed through an LSTM Decoder, which generates a route recommendation using an iterative process. A Grid Beam Search (GBS) algorithm is also integrated, to ensure the resulting itinerary adheres to user-defined restrictions.

A standout feature of Zhang et al.'s [53] approach is its support for real-time, user-driven route recalculations. This capability is enabled by a sophisticated GBS-based method that allows users to freely customize their itineraries. Furthermore, any modifications made by users are treated as feedback and used to fine-tune the Encoder-Decoder model, enhancing its adaptability and accuracy.

2.4.1.2 Collaborative Filtering

In the broader recommender systems field, perhaps the most popular data-driven recommendation method is collaborative filtering.

Collaborative Filtering (CF) is a recommendation technique that predicts user preferences by identifying patterns in historical user-item interactions. It is typically categorized into two main approaches: user-based CF, which generates recommendations by analyzing the similarity between users, and item-based CF, which focuses on the similarity between items.

A user-based implementation of CF is applied to the problem of recommending individual attractions to users by Supanich et al. [42]. To avoid the cold-start problem, the authors adopt an explicit approach to collecting users' past POI ratings and demographic information. Using these data, a neighborhood of 25 people with similar rating histories and demographic profiles is identified. Recommendations are then generated by applying CF to select POIs from among the most popular attractions within the identified neighborhood, ensuring relevance and personalization.

For all its advantages, in the specific context of tourism, CF often struggles due to its notable vulnerability to the cold start problem, as well as the inherent sparseness of data on user ratings of POIs. These limitations were highlighted by Wang [50]. To address these, the author introduces a novel similarity metric called Weighted J-M (WJM), derived from the Jeffries-Matusita (J-M) distance. This innovative measure accounts for variations in users' specific rating patterns, effectively compensating for the impact of data sparsity.

Both Bai et al. [5] and Zhong et al. [55] address the shortcomings of pure CF algorithms by mixing them with content-based techniques, a strategy known as Hybrid Filtering (HF). For Bai et al. [5], the primary focus is on overcoming the cold-start problem, with different solutions being used depending on whether it affects a new user or a new item.

- For new users, Demographic Filtering (DF) is employed to identify users with similar demographic profiles and Content-Based Filtering (CBF) is used to recommend appropriate POIs.
- For new items, item-based DF is applied to locate similar preexisting items and the new item is recommended to users who also enjoyed those counterparts.
- For established users and items, recommendations will increasingly be driven by both user-based and item-based CF.

In contrast, Zhong et al. [55] aim to recommend the most suitable POI within a target user's immediate surroundings. This is achieved by combing CF with CBF to match nearby attractions to users' known preferences. Additionally, their HF approach integrates a Machine Learning (ML) layer that fine-tunes recommendations based on more nuanced user preferences. This ML layer adapts over time through user feedback, enabling continuous improvement in recommendation accuracy and relevance.

2.4.1.3 Route patterns

For data-driven systems aiming to provide more robust route recommendations, a popular and efficient approach is leveraging route patterns, which are structured representations of partial historical travel routes. Systems taking this approach build itineraries by combining frequent patterns, which reflect routes commonly followed by real tourists.

Huang et al. [19] exemplify this approach with a three-stage process:

1. In the first stage, historical travel data is used to construct a heterogeneous graph that connects users and POIs. From this graph, two critical types of information are extracted: User preferences for POI categories and route patterns.
2. In the second stage, a Multi-layer Perceptron is employed to predict the probabilities of the next POI, given the current state of an elapsed route.
3. Lastly, to generate complete route recommendations, a Beam Search-based method, similar to the one in Zhang et al.[53], is adapted to focus on combining route patterns.

A more complex method for personalized route generation is proposed by Bin et al. [9] that incorporates not only route patterns, but also a multitude of user constraints such as the season of visit, type of companion (e.g., spouse, child), and traffic times between POIs. This system integrates data from multiple heterogeneous sources, creating a robust database of historical routes and respective user ratings, enriched with traffic times and tagged with the season of visit and companion type.

The database is processed using a PrefixSpan data mining algorithm to identify route patterns, which are used to generate candidate routes. These candidates are then ranked based on how well they align with user-specific restrictions and preferences, such as visit duration, preferred POI categories, companion type, and season of visit, to finally be presented to the user.

Instead of relying on individual route patterns, Xu et al. [51] compose multi-day itineraries by combining historical single-day routes. This ensures viability and realism in the recommendations, as each daily route has already been followed in the past.

A standout feature of Xu et al.'s work is their novel algorithm for extracting tourist preferences. This is done by analyzing POI reviews from a target user's historical travelogues to cluster POIs around keywords that represent POI attributes, and applying sentiment analysis to determine the user's feelings toward each attraction. These clusters are used to form a word set that encapsulates the user's preferences, which is subsequently compared with POI descriptions to calculate a personalized fitness score for each attraction.

2.4.2 Heuristic methods

Another widely adopted approach to recommending tourist itineraries, involves framing it as an optimization problem, called the Tourist Trip Design Problem (TTDP), and using heuristic, rule-based methods to approximate optimal solutions. The TTDP can draw on several well-established optimization problems as foundational blueprints to guide possible solutions.

One way to model the TTDP is as a Traveling Salesman Problem (TSP), which is the framework explored by Mahardika et al. [28]. In their study, the TSP is addressed using a Simulated Annealing (SA) approach, with a utility function that optimizes for three attributes simultaneously: attraction popularity, total travel time, and total cost. This method, known as Multi-Attribute Utility Theory (MAUT), is widely applied to handle multifaceted user preferences in the TTDP. To improve personalization of recommendations, the weight assigned to each attribute is adjusted based on each user's stated preferences. The algorithm is specifically designed to recommend multi-day itineraries by treating each day as a distinct TSP.

The TSP also serves as the foundation for the work of Maejima et al. [27]. Their proposed solution follows a greedy approach, leveraging MAUT to simultaneously optimize for POI proximity and alignment with user preferences. As in the paper by Mahardika et al.[28], each day is treated as a separate TSP. What sets this system apart however, is the possibility for real-time, user-driven route recalculation. This feature is designed as a quick solution to maintain itinerary coherence when faced with unexpected events.

Ananda et al. [2], on the other hand, highlight the shortcomings of applying the TSP model to multi-day itineraries, particularly in its lack of daily optimization. Alternatively, the authors

propose framing the TTDP as a Vehicle-Routing Problem (VRP), a well known logistical challenge, with each day of a trip represented as a separate vehicle within the problem. To solve the VRP, the authors developed a Whale-Optimization Algorithm (WOA), enhanced with Variable Neighborhood Search (VNS) elements, to avoid getting trapped in local optima, which they call WOA-VNS. They also integrate MAUT into the fitness function, optimizing for user preferences across attributes such as popularity, ratings, cost, and time.

In Li et al. [24], the multi-day TTDP is approached as a combination of the TSP and the VRP. To solve it, the authors employ a novel Contextual Multi-Armed Bandits (CMAB) algorithm, where the context includes user preferences, time and budget constraints, and even the season of visit. CMAB is specifically chosen for its ability to seamlessly incorporate contextual, user-dependent information and, equally, for its effectiveness in balancing the exploration-exploitation trade-off. Similar to the other algorithms discussed in this section, the reward function for the CMAB algorithm is defined using MAUT.

Although both the TSP and the VRP have been used to model the TTDP in specific contexts, most research suggests that the TTDP is best represented as an Orienteering Problem (OP).

Paulavičius et al. [30] define the TTDP as an Orienteering Problem with particularly rich constraints. While the study highlights a wide range of potential restrictions, only a select few are actually incorporated into the proposed algorithm. Among the most distinctive constraints considered are mandatory POIs, mutually exclusive POIs, and a novel tier system for POIs based on popularity, which limits tours from including an excess of POIs from a single category, such as Landmarks or Hidden Gems. These unique constraints, along with standard spatio-temporal restrictions, are integrated into the Greedy Genetic Algorithm (GGA) proposed as a solution to the OP.

Also accounted for is each user's familiarity with a city. Through this system, itineraries for first-time visitors prioritize landmarks and must-see attractions, whereas those for locals feature a greater proportion of lesser-known sites. Despite the range of attributes considered, the reward function for the GGA does not apply MAUT, instead, the system follows a two-module design. In the first module, personalized POI fitness is computed. The second module then employs the GGA to maximize the total precalculated fitness of the recommended itinerary, while ensuring adherence to time and space constraints.

Though Paulavičius et al. [30] don't account for multi-day routes, other similar OP-based approaches do. Notable examples of this are the 2023 study by Cena et al. [11] and its 2024 companion [13]. In both papers, particular focus is placed on time constraints affecting the OP. Dimensions considered include: POI opening and closing times, user preferences for free time, for POI visit times, and for density of visits in a day, and even lunch breaks. To solve the OP, a similar system to the one by Paulavičius et al.[30] is proposed, featuring an initial module for calculating personalized POI fitness and a second module that uses a Genetic Algorithm (GA) with greedy

characteristics to generate and select routes. In this case however, MAUT is an essential part of the GA's reward function.

Rusu et al. [38] also utilize a GA to solve a TTDP modeled as an OP. Unlike the previous studies, all computations, including fitness evaluation, are undertaken by a single GA-based module. The proposed algorithm is executed separately for each day of a multi-day trip, supports must-see attractions, and is even capable of recommending restaurants.

As demonstrated by the previous studies, an OP-based framework is certainly capable of handling multi-day itinerary recommendations. However, a more fitting analogy may lie in the Team Orienteering Problem (TOP). A sub-problem of OP, the TOP involves optimizing orienteering for an entire team of subjects. For a multi-day TTDP, each day can be equated to a member of the team, in a similar way to the VRP analogy in the paper by Ananda et al. [2].

This is the approach taken by Tarantino et al. [44]. In their study, the TTDP is modelled as a TOP and solved using a Multi-Objective Evolutionary Algorithm (MOEA). Instead of offering a single route recommendation to the user, the system presents an entire Pareto front of solutions, representing the best itinerary found for each of the attributes considered in the reward function of the MOEA. The algorithm is particularly innovative in its inclusion of users' accessibility requirements and its ability to respond to weather events and user delays with real-time recalculation.

Gavalas et al.[17] address a novel extension of the TTDP called the Vacation Planning Problem (VPP). While the objective in the TTDP is to recommend an itinerary within a known city, the VPP expands the scope to a broader region. This entails first selecting an ordered sequence of destinations within the region, dividing the available days among these, and finally, solving a potentially multi-day TTDP for each individual destination. In this context, the destination-specific TTDPs are modeled as Team Orienteering Problems with Time Windows (TOPTWs), which incorporate the opening and closing times of POIs. Each TOPTW is then solved using a heuristic derived from Iterated Local Search (ILS). The proposed algorithm is also capable of recommending hotels.

In the paper by Tlili et al. [45], a standard multi-day TTDP is modeled as a TOP. To solve it, the authors propose a novel two-part algorithm known as the KSA. The first part of the KSA involves applying K-Means clustering to group POIs by location, with the number of clusters matching the number of days in the trip. This step ensures that each day's recommended itinerary is geographically coherent and logistically feasible. After clustering, Simulated Annealing (SA) is employed to optimize the route within each cluster, identifying the most efficient order in which to visit the POIs for that day.

2.5 Discussion

The growing body of research in TRS design reflects the increasing prominence of the tourism industry [47] and the public desire for a truly comprehensive TRS to aid travel planning. Despite significant progress, the absence of such a TRS on the market highlights the complexity of the challenge and the need for continued investigation. Particularly important for a realistic and usable TRS, is a high degree of personalization, as well as integration of rich time and space constraints, to deliver tailored and practical recommendations that align with users' diverse tastes and logistical necessities [30]. This direction is being increasingly explored by recent studies. [2, 9, 11, 13, 24, 30]

As discussed previously, TRS research can broadly be categorized into data-driven and heuristic approaches, which deal with the problem in fundamentally different ways.

Data-driven methods leverage extensive user data to infer preferences and recommend POIs [5, 9, 19, 25, 42, 46, 50, 51, 53, 55, 56]. They generally excel at recommending individual attractions tailored to a user's particular tastes. However, they often fall short when generating complete itineraries, particularly when time and spatial constraints must be accounted for. This limitation is compounded by the inherent sparsity of tourism data, as most users visit only a limited number of POIs and rate even fewer, leading to challenges in training robust models [9].

Collaborative Filtering (CF), one of the most popular data-driven techniques in general recommender systems, is particularly inefficient in the tourism domain due to these sparsity issues [50]. Although some researchers have successfully mitigated this, often by integrating CF with other methods [5, 55], CF is yet to be adapted effectively for generating full itineraries. This demonstrates the limitations of relying solely on data-driven approaches in TRS.

For researchers determined to build personalized and comprehensive TRS using data-driven methods, combining route patterns derived from historical user itineraries into larger travel planes, emerges as the most reliable strategy [9, 19]. Nonetheless, even this approach has its limitations when users deviate significantly from established travel patterns or when new POIs are added.

One notable characteristic of data-driven methods is their emphasis on extracting detailed user preferences from large datasets. However, considering how effectively preferences can be predicted using demographic information and personality [1, 4, 12, 15, 29, 54], an alternative, potentially more efficient approach, might involve incorporating these factors directly. While this avenue remains underexplored, it offers a promising direction for future research.

Heuristic methods address the construction of TRS by framing it as a generic optimization problem, called the Tourist Trip Design Problem (TTDP), and applying algorithms to approximate optimal solutions. Many specific and established optimization problems may be used to model the

TTDP, these include the Traveling Salesman Problem (TSP) [24, 27, 28], Vehicle Routing Problem (VRP) [2, 24], and Orienteering Problem (OP) [11, 13, 17, 30, 38, 44, 45]. Among these, the OP is the most popular, due to its closer alignment with the requirements of the TTDP.

For more comprehensive multi-day TRS with a high degree of personalization and integration of time and space constraints, the Team Orienteering Problem with Time Windows (TOPTW) and rich constraints appears to be the most appropriate model. Its structure provides in-built support for multi-day routes by associating each day to a member of the team, time windows account for POI opening and closing times, and the constraints integrate user-specific restrictions and personalization. [13, 17, 30]

To solve the TTDP, Genetic Algorithms (GAs) emerge as a popular and promising path, due to their ease in incorporating constraints, scalability, and global search capability [11, 13, 30, 38]. To integrate user preferences, two strategies have been effectively used: a two-stage solution where personalized fitness scores are calculated for each POI before applying the GA [11, 13, 30] and a one-module design where everything is done by the GA [38]. Multi-Attribute Utility Theory (MAUT) may also be applied to ensure that the reward function of the GA considers multifaceted preferences [11, 13].

Finally, a few studies have proposed unique approaches that could be promising with further exploration. These include the possibility for hotel [17] and restaurant recommendations [38], the consideration of weather conditions [44] and season of visit [9, 24], and the application of user feedback for model fine-tuning [53, 55]. By broadening the scope of TRS research to encompass these and more additional dimensions, future systems could provide a more holistic and context-aware travel experience, further enhancing user satisfaction and system utility.

Below are two tables providing a classification of the different solutions proposed by the works included in this review. Table 2.2 highlights the characteristics and capabilities of each software solution and Table 2.3 lists the types of constraints supported by each.

Table 2.2: Characteristics of the solution.

Abbreviations: Class. = classification, Problem model = problem modeled as, DP alg. = base data-processing algorithm, Rec. alg. = base recommendation algorithm, MAUT = applies multi-attribute utility theory, MD = supports multi-day schedules, FB imp. = integrates feedback-based improvement, RTR custom. = capable of real-time route customization, Hot. = capable of hotel recommendation, Res. = capable of restaurant recommendation.

Study	Class.	Problem model	DP alg.	Rec. alg.	MAUT	MD	FB imp.	RTR custom.	Hot.	Res.
Tsai et al.[46]	Data-driven		LDA	LSTM						
Zhou et al. [56]	Data-driven		Text mining	novel						
Liang et al. [25]	Data-driven		LDA	novel		yes				
Zhang et al. [53]	Data-driven			LSTM/GBS			yes	yes		
Supanich et al. [42]	Data-driven			user-based CF						
Wang [50]	Data-driven			user-based CF						
Bai et al. [5]	Data-driven			HF						
Zhong et al. [55]	Data-driven			HF			yes			
Huang et al. [19]	Data-driven	RP combination	novel	MLP/BS						
Bin et al. [9]	Data-driven	RP combination	PrefixSpan	novel						
Xu et al. [51]	Data-driven	RP combination	novel	Greedy		yes				
Mahardika et al. [28]	Heuristic	TSP		SA	yes	yes				
Maejima et al. [27]	Heuristic	TSP		Greedy	yes	yes		yes		
Ananda et al. [2]	Heuristic	VRP		WOA	yes	yes				
Li et al. [24]	Heuristic	TSP/VRP		CMAB	yes	yes				
Paulavičius et al. [30]	Hybrid	OP		GGA						
Cena et al. [11, 13]	Hybrid	OP		GGA	yes	yes				
Rusu et al. [38]	Heuristic	OP		GA		yes				yes
Tarantino et al. [44]	Heuristic	TOP		MOEA	yes	yes		yes		
Gavalas et al. [17]	Heuristic	TOPTWs		ILS		yes			yes	
Tlili et al. [45]	Heuristic	TOP		SA		yes				
Proposed	Hybrid	TOPTWs		Clust. + It. BB		yes	yes			

Table 2.3: Constraints considered in the solution.

Abbreviations: Sp. dist. = considers spatial distribution of POIs, Mand. = allows for mandatory POIs, O/C tim. = considers POI opening/closing times, L. br. = makes room for lunch breaks, T. pref. = integrates POI visit time preferences of users, Budg. = integrates budgetary preferences of users, Fam. = considers users' familiarity with the location, Access. = considers accessibility requirements of users, Mul. MT = considers multiple modes of transport, Traff. = considers traffic conditions, Comp. = considers users' companion type, Seas. = considers season of visit, Wea. = considers weather conditions.

Study	Sp. dist.	Mand.	O/C tim.	L. br.	T. pref.	Budg.	Fam.	Access.	Mul. MT	Traff.	Comp.	Seas.	Wea.
Tsai et al. [46]													
Zhou et al. [56]	yes								yes				
Liang et al. [25]													
Zhang et al. [53]	yes												
Supanich et al. [42]													
Wang [50]													
Bai et al. [5]													
Zhong et al. [55]													
Huang et al. [19]	yes												
Bin et al. [9]	yes				yes					yes	yes	yes	
Xu et al. [51]	yes												
Mahardika et al. [28]	yes					yes							
Maejima et al. [27]	yes												
Ananda et al. [2]	yes					yes							
Li et al. [24]	yes					yes						yes	
Paulavičius et al. [30]	yes	yes					yes					yes	
Cena et al. [11, 13]	yes		yes	yes	yes								
Rusu et al. [38]	yes	yes		yes									
Tarantino et al. [44]	yes							yes					yes
Gavalas et al. [17]	yes		yes										
Tlili et al. [45]	yes												
Proposed	yes		yes		yes		yes						

Chapter 3

Methodological approach

This chapter outlines the methodological approach adopted in this study. It begins by explaining the research question and associated hypotheses, then narrows the scope of personalization to a finite set of dimensions rooted in the literature. Performance metrics are defined to evaluate the system and a formal problem definition is given. Finally, an overview of the implementation pipeline is given and the strategy used to validate the solution is described.

3.1 Research question

As established in Chapter 2.2.1, the goal of this study was synthesized into the Research Question (RQ) below. The intended meanings of the highlighted keywords in the RQ are explained in sequence.

RQ: How can a system recommend **complete** and **high-quality** touristic **itineraries** for users that conform to **user preferences** and **travel time restrictions**?

- **Complete:** Effectively uses the time dedicated to the trip.
- **High-quality:** Reflects what tourists actually want to visit.
- **Itinerary:** An ordered list of POIs to visit for each day of the trip.
- **User preferences:** The individual touristic tastes of each user, in other words, the aspects that should be fulfilled for that user to be satisfied in their trip.
- **Travel time restrictions:** Travel times between POIs and time spent at each POI.

3.2 Hypothesis

Using the RQ above as a basis, a Primary Hypothesis and associated Null Hypothesis were formulated. The system created as a result of this research was thus constructed to test the validity of

the Primary Hypothesis.

H1: A multi-day travel plan recommender system that takes into account user preferences and travel time restrictions is able to provide itineraries that are of higher-quality without compromising their completeness, when compared with baselines that do not consider these dimensions of personalization.

$H1_0$: (null) There is no difference between the proposed planner and baselines on quality or completeness.

3.3 Dimensions of personalization

As is evident from Tables 2.2 and 2.3, there exists a multitude of dimensions of personalization that could be taken into account for the generation of touristic recommendations. Considering the purposes of this dissertation and the existing literature, the scope of characteristics considered for the proposed solution was narrowed to the following:

- Users' demographic information, namely, age and gender [29] [1] [4].
- The POI features most valued by each user [54]. The features considered were:
  - History, or how historical the POI is [22, 37, 48].
  - Nature, or how much natural beauty is present in the POI [20, 48].
  - Art, or how artistically significant the POI is.
  - Culture, or how important the POI is for local culture [20, 22, 32, 37, 48].
  - Leisure, or how leisurely is the activity of visiting the POI [32].
- Users' familiarity with the city [30].
- Users' pace of visitation [9, 11, 13] in terms of:
  - Time to spend at each POI.
  - Speed to commute between POIs.
  - Maximum acceptable commute duration between POIs.

3.4 Performance metrics

To compare the itineraries created by the proposed solution to those of the baseline and competing systems, two quantitative dimensions were defined: *Completeness* and *Quality*. Each dimension is associated with a numerical metric, described in the following subsections.

3.4.1 Completeness

Completeness measures how efficiently an itinerary uses the daily time window available for visits. It is defined as the average percentage of usable trip time devoted to visits (as opposed to commutes) across the trip.

Let $\mathscr{P}$ denote the set of all POIs in the trip, and let D be the number of days. For each POI $i \in \mathscr{P}$ and day d, let $z_{d,i} \in \{0,1\}$ indicate whether POI i is scheduled on day d (1 if scheduled, 0 otherwise), and let $v_{d,i}$ be the visit duration of POI i on day d. The total visiting time on day d is then

$$V_d \;=\; \sum_{i \in \mathscr{P}} z_{d,i}\, v_{d,i}. \tag{3.1}$$

Let t_{start} and t_{end} denote the daily bounds of usable trip time defined by the user. Completeness is then given by

$$\text{Completeness} \;=\; \frac{1}{D} \sum_{d=1}^{D} \frac{V_d}{t_{\text{end}} - t_{\text{start}}}. \tag{3.2}$$

Equation (3.1) defines the total visiting time per day, while Equation (3.2) aggregates these values across the trip to produce the Completeness metric.

3.4.2 Quality

Quality measures how well the itinerary aligns with user tastes. It is defined as the average *fitness* of the POIs included in the trip. The fitness of a POI combines (i) the score given by the target user to that POI and (ii) the POI's popularity. The combination is controlled by a customizable popularity weight, which allows favoring major highlights for first-time visitors (through higher popularity weight) or hidden gems for familiar users (through lower popularity weight).

Let $\mathscr{P}$ be the set of all POIs, D the number of days, and $z_{d,i} \in \{0,1\}$ indicate whether POI $i \in \mathscr{P}$ is scheduled on day d. Let $r_{u,i}$ denote the rating of POI i for user u, and let p_i denote the popularity of POI i (both scaled to the same range). Define the popularity weight $w_{\text{pop}} \in [0,1]$, and let $w_{\text{usr}} := 1 - w_{\text{pop}}$. The fitness of POI i for user u is given by

$$f_i^{(u)} \;=\; w_{\text{usr}}\, r_{u,i} \;+\; w_{\text{pop}}\, p_i, \qquad w_{\text{usr}} + w_{\text{pop}} = 1. \tag{3.3}$$

Let $N = \sum_{d=1}^{D} \sum_{i \in \mathscr{P}} z_{d,i}$ be the total number of scheduled POIs in the trip. The Quality metric is the average POI fitness over all scheduled POIs:

$$\text{Quality} \;=\; \frac{\displaystyle\sum_{d=1}^{D} \sum_{i \in \mathscr{P}} z_{d,i}\, f_i^{(u)}}{\displaystyle\sum_{d=1}^{D} \sum_{i \in \mathscr{P}} z_{d,i}} \;=\; \frac{1}{N} \sum_{d=1}^{D} \sum_{i \in \mathscr{P}} z_{d,i}\, f_i^{(u)}. \tag{3.4}$$

Equation (3.3) defines POI fitness as a convex combination of user score and popularity, while Equation (3.4) aggregates these fitness values across the trip to produce the Quality metric.

3.5 Problem formalization

Given a user u, a set of candidate POIs $\mathscr{P}$ in a city and a set of trip days $\mathscr{D}$, the goal is to construct a multi-day itinerary that *maximizes* (i) *Completeness* (Equation 3.2) and (ii) *Quality* (Equation 3.4), while satisfying opening-hour and travel-time constraints for each day.

Sets and indices.

- $\mathscr{D} = \{1, \ldots, D\}$: set of trip days.
- $\mathscr{P}$: set of POIs.
- For each POI $i \in \mathscr{P}$, $\mathscr{K}_i$ indexes its open intervals $(a_{i,k}, b_{i,k})$, $k \in \mathscr{K}_i$.

Parameters.

- h: user-defined start location for daily itineraries.
- $t_{\text{start}}, t_{\text{end}}$: user-defined daily bounds of usable time.
- s_i: base visit duration for POI i (minutes).
- m_u: user-defined time modifier; personalized visit duration becomes $v_{u,i} = m_u s_i$.
- $\delta_{i,j}$: geodesic distance between POIs i and j (km).
- $\delta_{h,i}$: geodesic distance between the start location h and POIs (km).
- p_u: user-defined walking pace (minutes per km).
- $T_{\max}$: maximum allowable commute time between subsequent POIs (minutes).
- $f_i^{(u)}$: fitness of POI i for user u (Equation 3.3).

Decision variables.

- $z_{d,i} \in \{0,1\}$: 1 if POI i is visited on day d, else 0.
- $x_{d,i,j} \in \{0,1\}$: 1 if on day d POI j is visited immediately after POI i.
- $x_{d,h,i} \in \{0,1\}$: 1 if on day d the tour starts by going from the start location h to POI i.
- $\tau_{d,i} \in [0,\infty)$: start time of the visit to POI i on day d (minutes after midnight).

Derived quantities.

$$t_{i,j} = p_u \cdot \delta_{i,j} \qquad \text{(commute time, minutes).} \tag{3.5}$$

$$t_{h,i} = p_u \cdot \delta_{h,i} \qquad \text{(commute time, minutes).} \tag{3.6}$$

$$v_{u,i} = m_u s_i \quad \text{(visit time).} \tag{3.7}$$

Objective.

$$\max\ (\mathit{Completeness},\ \mathit{Quality}) \quad \text{subject to Constraints (3.9)–(3.17).} \tag{3.8}$$

Constraints. For all $d \in \mathscr{D}$ and $i, j \in \mathscr{P}$, $i \neq j$:

(C1) Day bounds.

$$\tau_{d,i} \geq t_{\text{start}} \quad \text{whenever } z_{d,i} = 1. \tag{3.9}$$

$$\tau_{d,i} + v_{u,i} \leq t_{\text{end}} \quad \text{whenever } z_{d,i} = 1. \tag{3.10}$$

(C2) Time windows.

$$\exists k \in \mathscr{K}_i \text{ s.t. } a_{i,k} \leq \tau_{d,i} \text{ and } \tau_{d,i} + v_{u,i} \leq b_{i,k} \quad \text{if } z_{d,i} = 1. \tag{3.11}$$

(C3) Visit at most once in the trip.

$$\sum_{d \in \mathscr{D}} z_{d,i} \leq 1. \tag{3.12}$$

(C4) Predecessor consistency.

$$\sum_{j \in \mathscr{P} \setminus \{i\}} x_{d,j,i} + x_{d,h,i} = z_{d,i}. \tag{3.13}$$

(C5) Successor consistency.

$$\sum_{j \in \mathscr{P} \setminus \{i\}} x_{d,i,j} \leq z_{d,i}. \tag{3.14}$$

(C6) Time propagation.

$$x_{d,h,i} = 1 \Rightarrow \tau_{d,i} \geq t_{\text{start}} + t_{h,i}. \tag{3.15}$$

$$x_{d,i,j} = 1 \Rightarrow \tau_{d,j} \geq \tau_{d,i} + v_{u,i} + t_{i,j}. \tag{3.16}$$

(C7) Commute bound.

$$t_{i,j} \leq T_{\max} \quad \text{whenever } x_{d,i,j} = 1. \tag{3.17}$$

Discussion. The objective (3.8) combines *Completeness* (Equation 3.2) and *Quality* (Equation 3.4) aiming for multi-attribute optimization in the itineraries created. Constraints (3.9)–(3.17) ensure that each day starts no earlier than t_{start}, finishes by t_{end}, every visit lies entirely within a valid opening interval, temporal feasibility is respected between consecutive POIs, and no individual commute exceeds $T_{\max}$.

3.6 Implementation pipeline

The two-layered architecture proposed by Paulavičius et al. [30] and Cena et al. [11] [13] was chosen as the most suitable basis for implementing a comprehensive solution. As such, the **Core module** of the proposed system consists of two layers:

1. The *Personalized Rating Prediction Layer*, which uses a specially trained AI model to predict users' ratings of POIs considering their preferences, demographics and the POI's features.

2. The *Itinerary Creation Layer*, which uses the fitness model provided by the first layer to generate a visitation schedule in accordance with the restrictions outlined in Section 3.5.

An additional module, called the **Importation Module** was created to support the core with the task of importing workable POIs. It is divided into two layers:

1. The *POI Importation Layer*, which uses city-specific prompts to import POI data and standardizes the data into a usable format.

2. The *POI Feature Estimation Layer*, which employs AI to create a model of POI characteristics. This is done by estimating the value of each POI in 6 feature dimensions: History, Nature, Art, Culure, Leisure and Popularity. The first five correspond to and are intended for comparison with user category preferences to predict personalized POI ratings. The final dimension, Popularity, is used to personalize recommendations to each user's level of city familiarity, in accordance with Equation (3.3). This layer is also responsible for estimating the average visit duration of each POI, which is used as a base for calculating personalized visit duration as outlined in Equation (3.7).

The near-infinite scope of cities from which touristic data could be imported was narrowed down to three: Porto and Lisbon in Portugal, and Paris in France. The first two were chosen for their prominent role in the national tourism scene of Portugal; and the latter for its variety and sheer volume of available POIs, together with its unmatched place in the global imagination as one of the most tourist hubs in the world. POI data for these cities was imported from OpenStreetMap (OSM), due to its status as the world's largest free and open database of geographic data.

The proposed system architecture therefore follows the model showcased in Figure 3.1. Details on the implementation of each module are available in Chapter 4.

3.7 Validation

To ensure the quality of all aspects of the system, the outputs of each of its 4 layers were validated separately. The first phase of validation (3.7.1) confirmed the quality of the POI data imported by the POI Importation Layer. The second (3.7.2) verified the validity of feature estimations provided

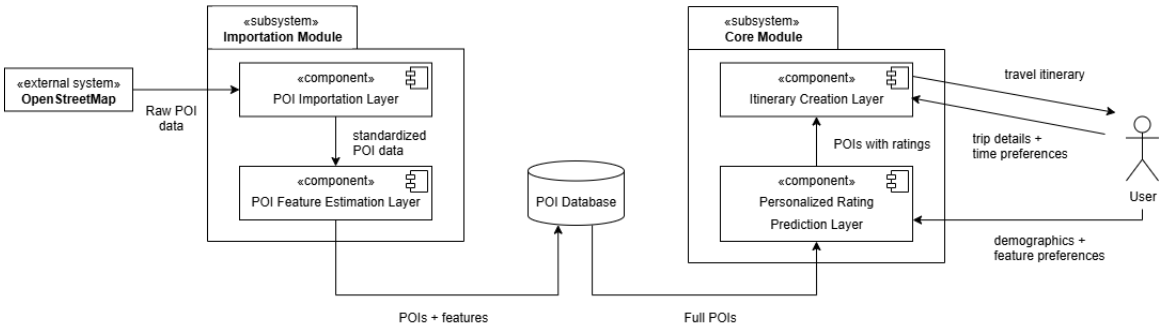

Figure 3.1: Proposed system architecture.

by the POI Feature Estimation Layer. The third (3.7.3) validated the ability of the Personalized Rating Prediction Layer to provide accurate POI rating predictions for users. And the fourth phase (3.7.4) compared the itineraries produced by the Itinerary Creation Layer to those of the baseline proving the validity of the hypothesis. This section outlines the method used to validate each of these layers, with the results achieved by applying these methods being explored in Chapter 5.

3.7.1 POI importation validation

As discussed in Section 3.6, the POI Importation Layer imports POIs by submitting city-specific prompts to the OSM API. To validate the quality of the prompt format chosen, the list of POIs it imported was compared to the Attractions tab of each city's Tripadvisor[1] page. Tripadvisor is the largest community-driven database of touristic data in the world, making it a useful approximation of what real tourists are interested in.

The Tripadvisor Attractions tab is approximately ordered by POI popularity and contains everything from must-sees to largely obscure and irrelevant attractions. As such, local experts from each of the three cities were asked to indicate an approximate cutoff point between relevant and irrelevant POIs in the list. Based on the values provided, the top 40, 60, and 90 POIs in the Tripadvisor Attractions tab for Porto, Lisbon, and Paris respectively, were considered for comparison with the POIs imported from OSM. The prompt used was deemed acceptable if it included a minumum of 75% of the Tripadvisor comparison lists for each of the cities in its importations.

3.7.2 POI feature estimation validation

As a benchmark against which to compare the estimations provided by the POI Feature Estimation Layer, a local expert was asked to estimate the values of the six feature dimensions (History, Nature, Art, Culture, Leisure, and Popularity) for a diverse, human-selected set of ten attractions in Porto, Portugal. Multiple AI models were tested, and their estimates were compared to the

[1] https://www.tripadvisor.com

human-estimated values using various quality metrics. The initial prompt used was also adjusted to achieve the best possible metrics. The model that achieved the best metrics in a reasonable estimation time was selected for the solution.

3.7.3 POI personalized rating prediction validation

To collect training data for the AI model and validate the quality of its predictions, a web survey was conducted over the course of two months. Respondents were requested to provide demographic information (Age and Gender) as well as their degree of preference for the five POI feature dimensions of History, Nature, Art, Culture and Leisure (Popularity is excluded as it is not used in rating prediction). Respondents were also asked whether they'd traveled to each of the three cities and if so were prompted to rate all POIs they remembered visiting. Their rating was used as the target variable for training and their preferences and demographic data, together with POI features, served as dependent variables.

The predictive powers of many different models were tested and compared using the standard quality metrics for regression models. Parameters were tuned to achieve the best metrics for each model tested and the leading overall model was selected for the solution.

3.7.4 Itinerary validation

Finally, to evaluate the impact of personalization on the Quality and Completeness of itineraries produced, a greedy baseline algorithm was developed that did not take into account either personal preferences or travel time restrictions when creating itineraries. Instead, it used average rating as a measure of POI fitness and default values for the travel time restriction variables of pace (15 min/km), time modifier (1), and maximum allowable commute time (30 min).

Many possible solutions, employing a diverse range of algorithms, were tested against the baseline using a comprehensive test suite including all three cities, as well as real preference data from five survey respondents, for a total of 15 tests for each approach. An algorithm was considered statistically superior to the baseline if it achieved, on average, a minimum 5% improvement in the Quality metric with any loss in Completeness being no greater than 50% of the Quality gain. Additionally, a human review of the itineraries generated by each algorithm was performed to assess their realism and conformance to subjective human standards of itinerary logic.

Chapter 4

Implementation

This chapter presents the implementation details of the proposed solution, primarily developed in Python, with the exception of the survey frontend, which was implemented in JavaScript. It is organized into six sections: Section 4.1 describes the importation of POI data from OpenStreetMap (OSM), while Section 4.2 outlines the subsequent data cleaning process, both carried out by the POI Importation Layer. Section 4.3 details the operation of the POI Feature Estimation Layer, and Section 4.4 explains the development of the survey used to collect user preference data. Finally, Section 4.5 presents the Personalized Rating Prediction Layer, and Section 4.6 discusses the generation of itineraries by the Itinerary Creation Layer.

4.1 OSM data importation

OSM data is structured using a simple model with three data types: *Node*, *Way* and *Relation*. Irrespective of type, every OSM element also contains a collection of key-value pairs called tags which describe its characteristics. Each OSM data type represents a different type of real-world object and is therefore structured differently:

1. *Nodes* represent points on earth and feature a latitude and longitude.
2. *Ways* symbolize a building or geographical area in the world and are structured as an ordered set of *Nodes*, i.e. a polygon.
3. *Relations* are a grouping of *Ways* and *Nodes* that form a coherent whole. These are often used to represent groups of areas or areas with holes.

Figure 4.1 shows an example of each of the three OSM data types. The *Node* example represents a location in the city of Porto, Portugal, the *Way* shows an area defined by the *Nodes* n1, n2 and n3, in that order, and the *Relation* combines multiple elements: a *Way* (w1) representing an outer area, another *Way* (w2) defining a hole inside w1, a separate second area (w3) and a *Node* (n1).

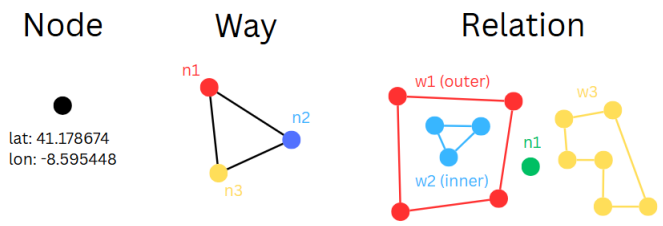

Figure 4.1: Data types in OSM.

OSM also provides a powerful read-only API called Overpass, which can be leveraged to query the database using its own query language, called OverpassQL (OQL). OQL allows for the definition of geographic search boundaries from established political divisions such as municipalities or states. It also permits filtering data points based on the presence or absence of certain tag keys or by matching tag values with target values. Using the mechanisms described, queries were created in OQL to discover POIs of touristic importance for each of the three cities considered (Porto, Lisbon and Paris). The structure of a generic query pseudocode is provided in Listing 4.1 and its implementation into OQL for the example city of Porto is shown in Appendix A.1. The query is structured as follows:

- Line 1 defines the output format as a JSON object.
- Line 2 sets a timeout of 90 seconds to ensure termination in case of a server crash.
- Lines 4-9 define the variable GREATER_AREA, which ideally encompasses the full extent of what tourists might consider visiting on a trip to the target city. This boundary often extends beyond the official city limits, as illustrated in the example city of Porto, where popular attractions such as Matosinhos Beach (Matosinhos municipality) [33] and Jardim do Morro [34] (Vila Nova de Gaia municipality) fall outside of the city proper.
- SURROUNDING_REGION, in line 11, establishes the maximum distance tourists are assumed willing to travel for attractions of outstanding natural beauty, recognizing that these often lie far beyond the urban perimeter.
- Line 13 specifies that the query may return objects belonging to any of the three OSM data types.
- Line 14 retrieves elements within GREATER_AREA that contain the tag "tourism" and simultaneously excludes those associated with accommodations (e.g., hotels, lodgings) or other non-attraction POIs.
- Line 15 expands the scope of search to important city parks, which are often missing the "tourism" tag in the OSM database despite their clear touristic appeal. To filter out small or

irrelevant parks, only those with both Wikidata and Wikipedia entries are retained, as these usually correlate with higher touristic significance.

- Lines 16-18 further broaden the scope to include forests, nature reserves, and natural parks within SURROUNDING_REGION. The additional filtration layer present for parks is also used in line 16 to exclude less relevant items from the search.
- Finally, line 20 requests that the geometries of the selected elements be included in the output.

Listing 4.1: Pseudocode for a generic OSM search query.

```
1  OUTPUT_FORMAT = JSON
2  TIMEOUT = 90 seconds
3
4  AREA_1 = AREA_FILTER(name="Area1")
5  AREA_2 = AREA_FILTER(name="Area2")
6  ...
7  AREA_N = AREA_FILTER(name="AreaN")
8
9  GREATER_AREA = UNION(AREA_1, AREA_2, ..., AREA_N)
10
11 SURROUNDING_REGION = AREA_FILTER(name="Surrounding_Area")
12
13 RESULTS = SELECT NODES, WAYS AND RELATIONS WHERE (
14 (IN_AREA(GREATER_AREA) AND HAS_TAG("tourism") AND VALUE("tourism") NOT IN ["
       guest_house", "hotel", ...]))
15 OR (IN_AREA(GREATER_AREA) AND HAS_TAG("leisure"="park") AND HAS_TAG("wikidata")
       AND HAS_TAG("wikipedia"))
16 OR (IN_AREA(SURROUNDING_REGION) AND HAS_TAG("landuse"="forest") AND HAS_TAG("
       wikidata") AND HAS_TAG("wikipedia"))
17 OR (IN_AREA(SURROUNDING_REGION) AND HAS_TAG("leisure"="nature_reserve"))
18 OR (IN_AREA(SURROUNDING_REGION) AND HAS_TAG("boundary"="national_park")))
19
20 OUTPUT(RESULTS, geometry_format="full")
```

A local database was created to store POI data, with dedicated tables for the individual *nodes* of OSM elements of type *Way* or *Relation*. City-specific OverpassQL queries are submitted to the Overpass API by a Python program, which also parses the results into a format compatible with the database. This program additionally derives a category from each POI's tags, specifying its touristic function (e.g., museum, church) and serving as a key feature in Section 4.3. The POIs obtained at this stage are stored in the temporary table `poi_temp`.

4.2 OSM data cleaning

To ensure quality in the imported data, two further post-processing functions are applied to the local database in order to merge or delete POIs deemed irrelevant. The procedure to judge whether two POIs should be merged follows the pseudocode in Listing 4.2 and is available in full in Appendix A.2. Likewise, the logic responsible for deleting unimportant POIs is described in Listing 4.3 and shown in Python in Appendix A.3. These procedures can be summarized as follows:

1. **POI merger** (Listing 4.2).
    - Line 1 filters out POIs that are geographically too far apart.
    - Line 2 merges POIs with excessively similar names.
    - Lines 4-5 address cases where historic buildings are listed separately from the exhibitions they host. As an example, in Paris, the entries "Louvre Museum" and the historic "Louvre Palace" in which it is housed are selected for merger here.
    - Line 6 handles a different edge case where a POI name contains another nearby POI name by merging them into the larger entity for clarity. A guardrail is also put in place to preserve POIs belonging to certain crucial categories, ensuring that irrelevant entries such as "Fountain of the square of ..." are merged with "Square of ...", while meaningful ones such as "Museum of the church of ..." remain separate from "Church of ...".
2. **POI deletion** (Listing 4.3).
    - Line 1 iterates over all POIs, excluding the target, that are of type *Way* or *Relation*, i.e., those with an associated geographical area.
    - Line 2 checks if the target POI is contained within the area of another POI. This process differs if the target POI is a simple point in space (*Node*) or is also linked to an area (*Way* or *Relation*).
    - Lines 3-4 provide a guardrail that prevents the deletion of POIs with an associated Wikidata entry, as this frequently indicates higher touristic importance.

Listing 4.2: Pseudocode of POI merger code.

```
1  IF DISTANCE_BETWEEN(POI1, POI2) < DISTANCE_THRESHOLD{
2    IF SIMILARITY(POI1.NAME, POI2.NAME) >= SIMILARITY_THRESHOLD {SHOULD_MERGE =
         TRUE}
3    ELSE IF SIMILARITY(POI1.NAME, POI2.ADDRESS_NAME) >= SIMILARITY_THRESHOLD {
         SHOULD MERGE = TRUE}
4    ELSE IF SIMILARITY(POI1.ADDRESS_NAME, POI2.NAME) >= SIMILARITY_THRESHOLD {
         SHOULD MERGE = TRUE}
```

```
    ELSE IF POI1.NAME CONTAINS POI2.NAME AND POI1.CATEGORY IS NOT CRUCIAL {
        SHOULD_MERGE = TRUE}
    ELSE IF POI2.NAME CONTAINS POI1.NAME AND POI2.CATEGORY IS NOT CRUCIAL {
        SHOULD_MERGE = TRUE}
    ELSE {SHOULD_MERGE = FALSE}
  }
  ELSE {SHOULD_MERGE = FALSE}
```

Listing 4.3: Pseudocode of POI deletion code.

```
  FOR POI2 IN POIS WHERE POI2 != POI1  AND POI2.DATA_TYPE != NODE{
    IF POI1 INSIDE_OF POI2 {
      IF POI1.WIKIDATA IS NONE {SHOULD_DELETE_POI = TRUE}}
      ELSE {SHOULD_DELETE_POI = FALSE}
    }
    ELSE {SHOULD_DELETE_POI = FALSE}
  }
```

In addition to the automated data cleaning process, a few manual adjustments were applied. Firstly, as described in Section 3.7.1, the list of imported POIs was compared to the top elements in the Tripadvisor Attractions tab for each city, with POIs present in the latter but missing from the former being added manually to ensure completeness. A human review of the database was also conducted to remove the few elements of no touristic value which were able to survive the data cleaning process.

4.3 POI feature estimation

To estimate the values of the six feature dimensions (History, Nature, Art, Culture, Leisure, and Popularity) for the POIs, a Large Language Model (LLM) was provided with POI names and respective categories, and instructed to estimate an integer score from 1 to 10 for each dimension, as well as the average visit duration (in minutes). POIs were submitted to the LLM in batches of ten, which was experimentally determined to be a near-optimal balance between computational speed, result accuracy, and prompt size limitations. The prompt used was developed following established prompt-engineering principles [8] and is included in full in Appendix A.4. Its structure is outlined below:

1. Request grading of a given set of POIs in a specific city across the six feature dimensions.
2. Clarify the meaning of each dimension by describing characteristics associated with low and high scores for each.
3. Request estimation of average visit duration.

4. Specify the desired output format and explicitly forbid common formatting deviations observed during testing.

5. Warn against assigning excessively high popularity scores, an issue identified during preliminary tests.

6. Instruct the LLM to treat POIs with the same name but different categories as distinct entities and rate them separately.

7. Provide the list of POIs to be rated, along with their categories.

After receiving the model's output, the Python program parses the estimated feature values and visit durations and stores the enriched POIs in a new table named `poi` in the local database.

4.4 Survey

The survey used to collect training data for the personalized grade prediction model was implemented using a Frontend-Backend-Database architecture, shown in Figure 4.2. The frontend was developed in React and hosted on Vercel [1], the backend was written in Python and hosted on Render [2], and the database was created in SQLite and hosted on SQLite Cloud [3]. The three layers communicate with each other exclusively via HTTP requests, no direct communication is allowed between the frontend and database as all requests are routed through the backend.

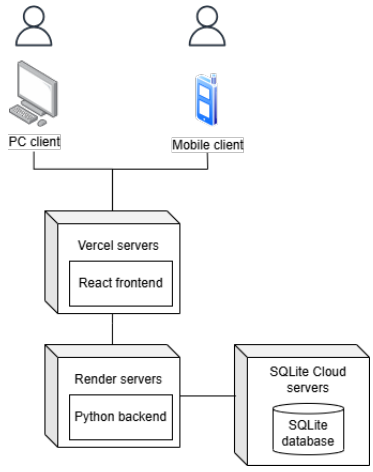

Figure 4.2: Survey deployment architecture.

The page-by-page navigation flow of the survey is described below:

[1] https://vercel.com
[2] https://render.com
[3] https://sqlitecloud.io

1. **Page 1 - Demographic Information:** Upon accessing the survey website, participants are asked to provide their demographic information, namely age and gender. All three fields are mandatory and a field requesting the participant's name was omitted for privacy.

2. **Page 2 - User Preferences for POI Features:** Participants are presented with five sliders, one for each of the POI features considered in POI rating prediction (History, Nature, Art, Culture, Leisure). Each slider uses a five-point Likert-like scale with the following options: "I have no real interest in _", "I have little interest in _", "I have some interest in _", "I'm very interested in _", and "My priority when traveling is _", where _ is replaced by the feature name. Sliders default to the middle value and the displayed text updates dynamically as they are moved. Values are stored in the database as the integers 1 to 5.

3. **Page 3 - City Selection:** Participants are asked to select all visited cities out of a list containing the cities considered (Porto, Lisbon and Paris).

4. **Page 4 - POI Selection:** All POIs from the first city selected in the previous page are displayed in a grid with their names, categories, and photos (if available). Participants select all POIs they recall visiting.

5. **Page 5 - POI Scoring** The POIs selected in the previous page are shown in a grid and participants are asked to assign them each a rating on a scale from 1 to 5 (in 0.5 increments). Upon completion, in the case that multiple cities were selected on Page 3, the survey returns to Page 4 and repeats this process for the next remaining city until none remain.

6. **Page 6 - Thanks** After all POIs for all selected cities have been rated, a thank-you message from the author is displayed.

Regarding language support, although the survey was primarily intended for a Portuguese-speaking audience, it was initially presented in English, with the option to switch to Portuguese and back to English at any time. This design choice was made due to the higher prevalence of POI names available in English, as compared to Portuguese.

4.5 Personalized rating prediction

The data collected from the survey, comprising user demographic information, user preferences for the five POI features, and user ratings of visited POIs, were combined with the POI feature values estimated by the POI Feature Estimation layer. This combined dataset was then cleaned using the procedure described in Appendix A.5 to produce a training dataset for the Personalized Rating Prediction layer. After cleaning, the dataset was left with the target variable `target_rating`, representing the rating given by a user to a POI, and the following groups of dependent variables:

- **POI features:** `poi_history`, `poi_nature`, `poi_art`, `poi_culture`, `poi_leisure`, `poi_popularity`, `poi_score`

- **User preferences:** `user_history`, `user_nature`, `user_art`, `user_culture`, `user_leisure`
- **User demographics:** `user_age`, `user_gender_male`, `user_gender_female`, `user_gender_other`

The cleaned data was then used to train a regression model to predict `target_rating` from the dependent variables using a hybrid filtering approach that concatenates collaborative filtering and content-based features. Latent user and POI vectors were first extracted via matrix factorization using Singular Value Decomposition (SVD) to generate baseline rating estimates using CF. These were then combined with the dependent variables listed above and used to train an ElasticNet-based regression model, whose hyperparameters were tuned to maximize predictive accuracy.

4.6 Itinerary generation

The itinerary generation process relies on a composite fitness score for each POI, defined in Equation (3.3) as a weighted combination of its predicted user rating and popularity. The user can adjust the relative weight of these two components, enabling the prioritization of highly popular POIs for first-time visitors or hidden gems for locals or frequent visitors.

To generate personalized itineraries, the proposed solution employs a novel algorithm that combines an initial POI clustering step and solves each day's route as an Orienteering Problem with Time Windows (OPTW) using a Branch and Bound (BB) approach. In the first step, a Constrained K-means clustering algorithm with an additional refinement step partitions the candidate POIs into as many clusters as there are days in the trip, producing groups that are both geographically coherent and roughly balanced in size and average POI fitness. In the second step, each cluster is treated as an independent OPTW instance and solved with BB.

As discussed in Section 3.5, the BB implementation respects multiple temporal and spatial constraints to ensure that the generated itineraries are feasible and, most importantly for this study, personalized to each user's touristic profile. These constraints include:

- **`time_mod`:** Adjusts the base estimated visit duration for each POI. Values above 1 favor longer visits, resulting in fewer POIs being visited per day, while values below 1 favor shorter visits with more POIs per day.
- **`start_h`:** The start time of each day's itinerary.
- **`end_h`:** The latest allowable end time of each day's itinerary.
- **`start_loc`:** The starting location of each day's itinerary.
- **`pace`:** The walking pace (in minutes per kilometer) to use in travel-time calculations.

- **`max_walkable_mins`:** The maximum allowable travel time between two consecutive POIs.

Because BB is an optimal algorithm with potentially exponential time complexity, several efficiency measures were implemented to reduce runtime without sacrificing optimality:

- Precomputing a POI travel-time matrix and discarding edges that exceed `max_walkable_mins`.
- Caching opening-hour intervals for each POI.
- Pruning subtrees whose lower bound already exceeds the day's time limit.
- Memoizing visited states and applying Pareto dominance to prune states that are strictly worse in travel time.
- Ordering candidate POIs by earliest feasible start time and limiting branching via a beam width.
- Skipping unreachable successors outright.
- Using depth-first expansion to quickly find good incumbents and strengthen subsequent pruning.
- Encoding visited sets as bitmasks and using integer representations for times to speed up state checks and comparisons.

The main entry point for itinerary generation is the `create_plan` function. In addition to the constraints listed above, it accepts the following arguments:

1. **`user_id`:** Id of a user with preferences and ratings in the database.
2. **`city_id`:** Id of one of the three cities in the database.
3. **`pop_weight`:** A number between 0 and 1 specifying the percentage of a POI's fitness to be derived from its popularity, with the remainder coming from the predicted rating.
4. **`days`:** A list of the days in which the trip will take place.
5. **`file_name`:** A name for the json file to which the itineraries generated will be saved.
6. **`visualize`:** An optional boolean flag that, if True, produces map visualizations of each day's itinerary.

The high-level logic of the itinerary generation pipeline is presented in Listings 4.4 and 4.5, with the full Python code being available in Appendix A.6. The flow operates as follows:

1. **Listing 4.4.**

- Line 4 invokes the Personalized Rating Prediction algorithm which returns a list of all POIs in the selected city with ratings estimated for the specified user.
- Lines 5-7 compute each POI's fitness according to Equation (3.3) and adjusts their visit duration using the user-defined `time_mod`.
- Line 9 applies constrained K-means clustering to group the POIs into N clusters of approximately equal size, where N is the number of days in the trip.
- Line 10 calls `decide_day_per_cluster` which uses the opening days of each cluster's POIs to assign clusters to their most suitable trip days, returning a list of (`day`, `cluster_id`) pairs. The code for this function is available in Appendix A.7.
- Line 11 invokes `refine_clusters`, which swaps POIs between clusters to balance their average fitness.
- Lines 13-14 call `iterative_bb_planner` to solve each cluster as a OPTW using BB and combine the resulting daily plans into a full multi-day itinerary. The flow of this is shown in Listing 4.5 and explained later in this section.
- Line 16 exports the resulting itinerary as a JSON file.
- Lines 18-20 call `create_html_visuals` which uses the OpenRouteService Python API to generate map-based visualizations of each day's itinerary as HTML files.

2. **Listing 4.5.**

- Line 6 selects the top-N highest-fitness POIs in the cluster (starting with $N = 2$), called the prefix.
- Line 7 applies the BB algorithm to attempt to find the most suitable route including all POIs in the prefix, in other words, solve the prefix as a TSP. If no feasible route exists, the returned plan is marked with `feasible` = False.
- Lines 9-11 check the plan's feasibility. If feasible, it becomes the new `best_plan` and N is incremented so that the next iteration tries a larger prefix.
- If no feasible plan is found for the current prefix, lines 12-13 remove from the pool of available POIs, the POI that caused the infeasibility (present in the current prefix but absent from the last feasible prefix).
- Finally, line 5 determines the repetition of the above flow until all remaining available POIs are included in the current `best_plan`, which will happen when N, the size of the current `best_plan`, exceeds the size of the remaining available POI pool.

Listing 4.4: Pseudocode of trip plan creation code.

```
1
```

```
2  create\_plan(user_id, city_id, pop_weight, time_mod, days, start_h, end_h,
       start_loc, pace, max_walkable_mins, save = False, file_name = None, visualize
        = False):
3
4    POIS = PREDICT_USER_RATINGS_FOR_CITY_POIS(user_id, city_id)
5    FOR POI IN POIS
6      POI.FITNESS = POI.RATING * (1 - pop_weight) + POI.POPULARITY * pop_weight
7      POI.VISIT_DURATION = POI.AVERAGE_VISIT_DURATION * time_mod
8
9    POI_CLUSTERS = CKM_CLUSTERER(POIS, days)
10   DAY_PER_CLUSTER = DECIDE_DAY_PER_CLUSTER(POI_CLUSTERS, days)
11   POI_CLUSTERS = REFINE_CLUSTERS(POI_CLUSTERS, DAY_PER_CLUSTER)
12
13   FOR DAY, CLUSTER_ID IN DAY_PER_CLUSTER:
14     DAILY_PLANS[DAY] = iterative_bb_planner(POI_CLUSTERS[CLUSTER_ID], DAY,
           start_h, end_h, start_loc, pace, max_walkable_mins)
15
16   CREATE_JSON_FILE(NAME = file_name + ".json", CONTENT = DAILY_PLANS)
17
18   IF visualize:
19     FOR DAILY_PLAN IN DAILY_PLANS:
20       CREATE_HTML_VISUALS_FILE(NAME = file_name + DAILY_PLAN[DAY] + ".html",
             CONTENT = DAILY_PLANS[DAY])
```

Listing 4.5: Pseudocode of daily plan creation code.

```
1
2  iterative_bb_planner(cluster_of_pois, day, start_h, end_h, start_loc, pace,
       max_walkable_mins):
3
4  N = 2
5  WHILE N <= SIZE(cluster_of_pois):
6    PREFIX = FIRST N ROWS OF cluster_of_pois
7    PLAN = BRANCH_AND_BOUND(PREFIX, day, start_h, end_h, start_loc, pace,
         max_walkable_mins)
8
9    IF PLAN.FEASIBLE:
10     BEST_PLAN = PLAN
11     N = N + 1
12   ELSE:
13     REMOVE ROW AT POSITION (N-1) FROM cluster_of_pois
14
15 RETURN BEST_PLAN
```

Chapter 5

Results and discussion

This chapter critically examines the results obtained at each stage of the proposed solution and evaluates them against the validation goals defined in Section 3.7. It is organized into four sections, each dedicated to one of the system's layers. The first section evaluates the effectiveness of the POI importation process by comparing its output from OSM against Tripadvisor's "Attractions" tab. The second assesses the performance of the different LLMs tested for POI feature estimation and provides a detailed analysis of the chosen model's estimates. The third examines the results of the survey and compares the predictive performance of the regression models tested for personalized POI rating prediction. Finally, the fourth analyzes the *Quality* and *Completeness* of itineraries generated by the various TTDP solver algorithms and compares them to the baseline.

5.1 POI importation

The POI importation prompt was deemed acceptable by successfully retrieving at least 75% of the top N elements in the Tripadvisor "Attractions" tab for each of the three cities considered. N was set at 40 for Porto, 60 for Lisbon, and 90 for Paris to reflect distinct city sizes. Using this criterion, the final prompt, provided in full for the example city of Porto in Appendix A.1, achieved the following results:

Table 5.1: Percentage of the top N items in the Tripadvisor "Attractions" tab automatically imported from OSM using the proposed prompt, by city.

City	Country	n	Success rate	Success percentage
Porto	Portugal	40	30/40	75%
Lisbon	Portugal	60	46/60	76.67%
Paris	France	90	70/90	77.78%

The results show an approximately constant success percentage across cities, suggesting that the chosen values for N were well calibrated: increasing or decreasing any one of them was observed to result in significant imbalances between cities' success rates. Notably, the types of

POIs missed by the algorithm were remarkably consistent across cities and fall mostly into one of three identifiable categories:

1. **Neighbourhood:** Well known touristic neighbourhoods or areas, which are not considered by the prompt at all.

   Examples: Cais da Ribeira (Porto), Alfama (Lisbon), Montmartre (Paris).

2. **Streets:** Touristic streets, which are inconsistently marked with the label "tourism" in OSM, causing many relevant streets to be left out.

   Examples: Rua das Flores (Porto), Rua Augusta (Lisbon), Rue Cler (Paris).

3. **Novelty:** Novelty experiences that may be enjoyed by tourists but are not typically classified as touristic landmarks in OSM.

   Examples: USAxe Club, an axe-throwing venue (Porto), LXFactory, an open-air retail space (Lisbon), Shakespeare and Company, an English-language book shop (Paris).

Several attempts were made to alter the prompt to increase the success rate of automatic POI importation by capturing these missing categories. However, all such adjustments led to the inclusion of an unacceptably large number of irrelevant POIs alongside the desired ones. Consequently, the compromise was made to add the missing POIs manually, as discussed in Section 4.2.

5.2 POI feature estimation

5.2.1 Model testing

To select the model to use for POI feature estimation, several LLMs were tested for their ability to approximate human estimates of the characteristics. Each model was asked to assign scores for the six feature dimensions considered (History, Nature, Art, Culture, Leisure, and Popularity) to a pre-selected set of ten POIs with distinct touristic profiles in the city of Porto. The resulting estimates were compared to those produced by a local expert, and model accuracy was evaluated using Mean Absolute Error (MAE) and Mean Squared Error (MSE).

Table 5.2 presents the results of this comparison. Among the tested models, Google's Gemma-2-27b-it achieved the lowest MSE overall and the lowest MAE of any locally run model and was therefore selected as the basis for POI feature estimation in the proposed system.

5.2.2 Data analysis of estimates

A detailed analysis was performed on the feature estimates produced by the selected model to assess their internal consistency and better understand the patterns in the data.

The three selected cities share broadly similar touristic profiles: dense, highly urban Western European cultural hubs with rich histories. This is reflected in Table 5.3, where all three exhibit comparable feature distributions, with high mean Art scores (≈ 5.9) and low mean Nature scores

Table 5.2: Comparison of LLM and human ratings of POI features.

Model name	**Hosted**	**MAE**	**MSE**
GPT-4o	Online	2.1	6.3
DeepSeek-V3	Online	1.68	4.88
deepseek-r1:32b	Local	2.67	14.2
Meta-Llama-3-70B	Local	1.85	5.58
Gemma-2-2b-it	Local	1.92	6.52
Gemma-2-9b-it	Local	1.88	5.98
Gemma-2-27b-it	Local	1.82	4.62

Table 5.3: Comparison of POI feature distributions by city.

Metric	Porto	Lisbon	Paris	Total
Number of POIs	459	747	1828	3034
Popularity (mean)	4.46	3.96	3.65	3.85
Popularity (SD)	2.01	1.95	1.91	1.95
Popularity (median)	4	3	3	3
History (mean)	4.61	4.45	4.66	4.60
History (SD)	2.32	2.40	2.26	2.3
History (median)	4	4	4	4
Nature (mean)	3.06	3.10	3.09	3.09
Nature (SD)	2.56	2.43	2.58	2.54
Nature (median)	2	2	2	2
Art (mean)	5.86	5.87	6.19	6.06
Art (SD)	2.08	1.93	2.08	2.05
Art (median)	6	6	7	7
Culture (mean)	4.98	4.79	4.91	4.89
Culture (SD)	1.91	1.83	1.79	1.82
Culture (median)	5	5	5	5
Leisure (mean)	5.01	4.84	5.13	5.04
Leisure (SD)	1.72	1.61	1.55	1.60
Leisure (median)	5	5	5	5

(≈ 3.1), while the other features lie between these extremes. Had this analysis been applied to cities with markedly different profiles, such as the nature-heavy Edinburgh [40], leisure-centric Las Vegas [35], or highly historic Rome [41], substantially different distributions should be expected. Nevertheless, some statistically significant differences between the cities were noted and critically examined:

- **History:** Lisbon's mean History score is significantly lower than that of the other cities ($p = 0.01483$), likely reflecting the scarcity of Renaissance-era and medieval sites, due to the near-total destruction of the city in the 1755 earthquake [3].
- **Art:** Paris has a significantly higher mean Art score than the other cities ($p = 0.00001$), aligning with its long-standing reputation as a global center for art [36].
- **Leisure:** Paris also shows a significantly higher mean Leisure score ($p = 0.00009$), which may be explained by its perception as an entertainment capital, with an abundance of galleries, shopping centers, cafés, theatres, and nightlife venues of touristic relevance.
- **Popularity:** Porto has a significantly higher mean Popularity than Lisbon (Welch's t-test, $p = 0.000009$), which in turn is significantly higher than in Paris ($p = 0.00017$). This may indicate a negative correlation between the total number of POIs in a city and mean Popularity, though further testing is needed to confirm this effect beyond these specific cities.

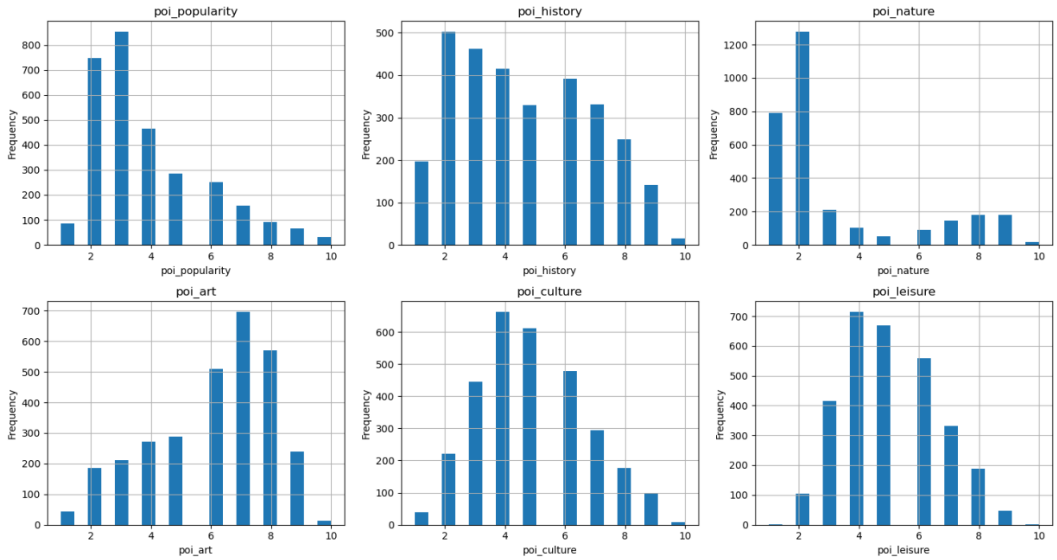

Figure 5.1: Distribution of feature estimates.

Examining the histograms of feature scores across all cities in Figure 5.1 reveals additional patterns:

- The scarcity of values at the extremes (1 and 10) in most features suggests the model is reluctant to assign extreme scores.

- Nature exhibits the most skewed distribution, with a strong concentration of low scores and a small secondary peak at very high values. This reflects the urban character of the dataset: most POIs contain little nature, while the few that do (primarily parks, squares, and reserves) are strongly nature-focused and thus receive very high Nature scores [23].

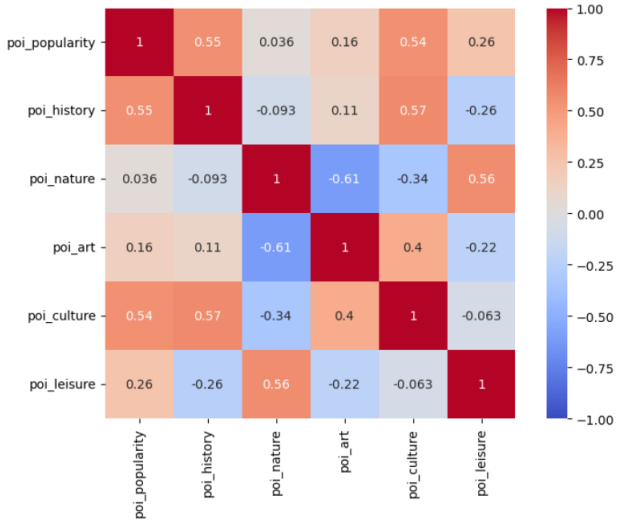

Figure 5.2: Heatmap of correlation between POI features.

Finally, a correlation analysis of the feature scores in Figure 5.2 also revealed several noteworthy relationships:

- History correlates with Culture (+0.57), as historical heritage is often tightly intertwined with local culture [49].
- History also shows a slight negative correlation with Leisure (-0.26), since highly leisurely POIs such as shopping malls or theme parks tend to be less historical [18, 21].
- History and Culture both correlate with Popularity (+0.55 and +0.54 respectively), showing how older and more culturally significant POIs attract more visitors [6].
- Nature correlates with Leisure (+0.56), as high-Nature POIs like parks are often associated with relaxation, rather than knowledge acquisition [7].
- Furthermore, Nature shows its highest negative correlation with Art (-0.61), since, apart from a few open-air museums, most high-Art POIs are indoors and contain little nature [26].
- For the same reason, Nature also has a slight negative correlation with Culture (-0.34), as many cultural venues are indoors and thus low in Nature [31].
- Art correlates with Culture (+0.40), as artistic traditions are often an important component of cultural heritage [52].

- Additionally, Art has a slight negative correlation with Leisure (-0.22), since many high-Art POIs are museums, which are intensive rather than leisurely activities [14].

- Finally, regarding Popularity, Nature stands out as the only feature showing no correlation. This can be explained by the fact that POIs with the highest Nature scores are often forests or reserves located far from city centers, being therefore less popular with tourists [43]. This interpretation is supported by the data in Figures 5.3 and 5.4, which show the relationship between each feature and distance from the city centre. These figures illustrate that Nature is the only feature consistently positively correlated with distance from the centre, which, in turn, is negatively correlated with Popularity across all cities.

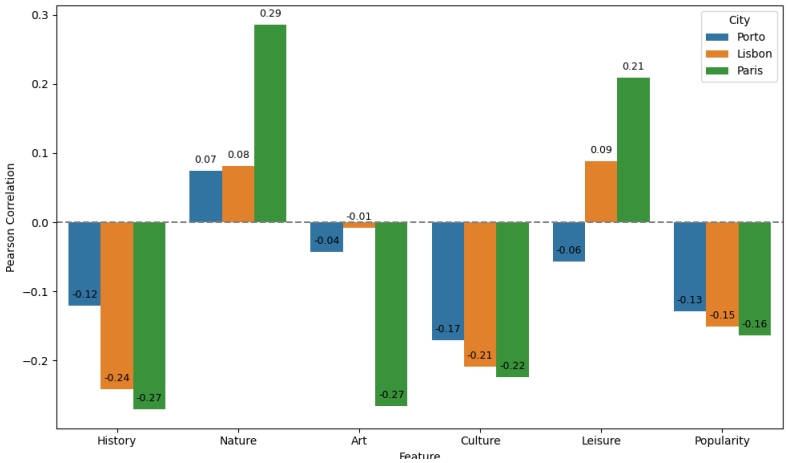

Figure 5.3: Correlation between POI features and distance from the centre, computed using fixed city centres: Porto (41.1413, -8.6149), Lisbon (38.7075, -9.1365), and Paris (48.8707, 2.3323).

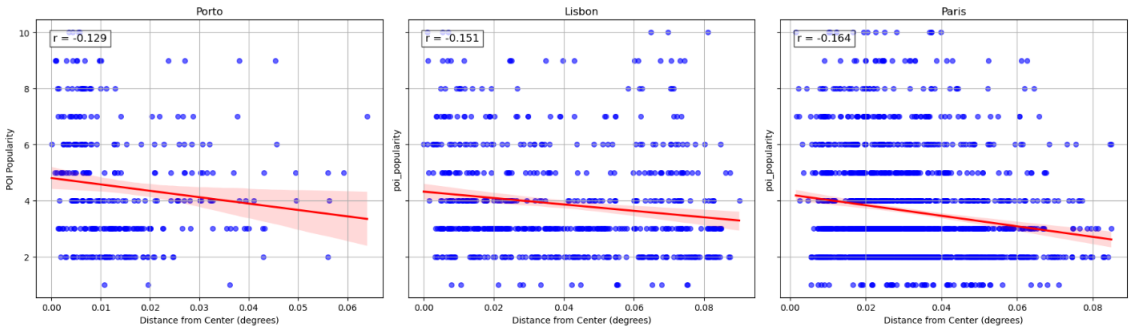

Figure 5.4: Correlation between POI Popularity and distance from the centre per city, computed using fixed city centres: Porto (41.1413, -8.6149), Lisbon (38.7075, -9.1365), and Paris (48.8707, 2.3323).

5.3 Personalized POI rating prediction

5.3.1 Data analysis of survey results

To construct the training dataset for the personalized POI rating prediction model, a survey was conducted between 22/05/2025 and 23/07/2025, resulting in responses from 184 distinct participants. An analysis of the respondents' demographic profiles, shown in Figure 5.5, reveals a predominantly young sample, with an average age of 24, and a slight gender imbalance, with approximately 57% identifying as women. This profile closely reflects the Portuguese university environment in which the survey was primarily disseminated [16].

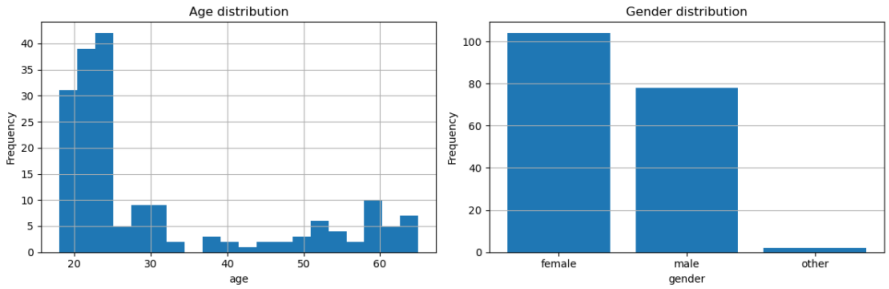

Figure 5.5: Age and gender distributions of survey respondents.

In total, the 184 participants submitted 8,667 ratings covering 482 unique POIs, averaging just over 50 ratings (50.28) per respondent. As shown in Figure 5.6, these ratings are heavily skewed towards the higher end of the scale, with an overall mean of 8.38. This pattern is consistent with the well-documented phenomenon of acquiescence bias, in which positive responses are selected more frequently than negative ones in survey-based settings, a tendency also observed in online ratings of tourist attractions [39].

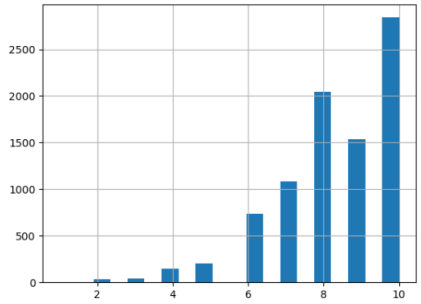

Figure 5.6: Histogram of POI ratings by survey respondents.

Figure 5.7 presents the distributions of respondents' self-declared preferences across the five POI feature dimensions. All five exhibit unimodal distributions skewed towards high values, featuring: a mode of 4, few 1 scores, and broadly similar shapes. Among them, Culture stands out as the most universally valued feature, with the highest mean score (4.12) and lowest standard

deviation (0.71), followed closely by History in both metrics (mean = 3.92, std = 0.92). This suggests stronger consensus among participants on the importance of these two dimensions [37]. In contrast, Art emerges as the most polarizing feature, being the only one with a standard deviation above 1 (1.06) and a median of 3 (compared to 4 for the others), indicating a more divided perception, with both strong supporters and detractors [10].

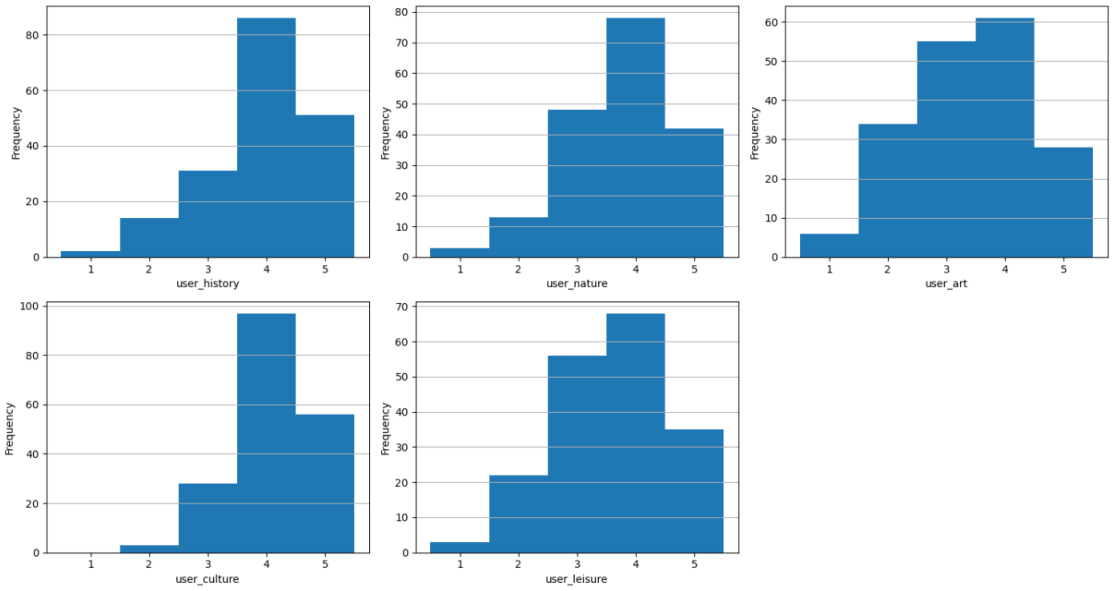

Figure 5.7: Histograms of preferences for the five feature dimensions by survey respondents.

5.3.2 Personalized POI rating prediction algorithm comparison

The data collected from the survey was combined with POI feature estimates into the dataset used both to train and evaluate the performance of POI rating prediction models. To determine the most suitable model for personalized rating prediction, a broad range of Content-Based Filtering (CBF), Collaborative Filtering (CF), and Hybrid Filtering (HF) regression algorithms were tested. Each algorithm underwent a random search over an extensive set of hyperparameters to ensure that its best-performing configuration was considered in the comparison. Table 5.4 lists the algorithms evaluated, along with the best Root Mean Square Error (RMSE) achieved for each during the random search process, and their classification as CBF, CF, or Hybrid.

The results clearly indicate that HF approaches, which combine CF-derived latent factors with explicit content-based features through feature concatenation, achieve substantially higher predictive accuracy than either pure CF or pure CBF models. Among all tested configurations, the SVD + Elastic Net Regressor achieved the lowest RMSE and was therefore selected as the final predictive model used in the proposed system for personalized POI rating prediction.

Table 5.4: Comparison of algorithms for personalized POI rating prediction.

Algorithm	Best RMSE	Classification
Elastic Net Regressor	1.4653	CBF
Random Forest Regressor	1.4698	CBF
Gradient Boosting Regressor	1.5299	CBF
KNN Regressor	1.4912	CBF
XGB Regressor	1.4787	CBF
SVR Regressor	1.4679	CBF
Decision Tree Regressor	1.5342	CBF
LightGBM Regressor	1.4767	CBF
AdaBoost Regressor	1.4687	CBF
Bagging Regressor	1.4686	CBF
MLP Regressor	1.6756	CBF
Huber Regressor	1.4729	CBF
SVD	1.2988	CF
SVD with implicit-aware MF	1.3296	CF
Item-based KNN	1.4278	CF
SVD + Elastic Net Regressor	0.6446	Hybrid
SVD + XGB Regressor	0.6519	Hybrid

5.4 Route generation

As with the POI rating prediction module, a wide range of algorithmic strategies were explored to generate multi-day itineraries from a set of graded POIs. These approaches are summarized below:

1. **Naive Greedy (NG):** A simple greedy algorithm that iteratively selects the highest-rated POI until each day is full. Each day's selected POIs are then ordered to form the most efficient route.

2. **Dynamic Greedy (DG):** Similar to NG, but at each step evaluates POIs based on a combined score of their weighted fitness minus their weighted distance from the previously visited POI. This helps reduce total travel times, though often at the cost of slightly lower average POI scores.

3. **Simulated Annealing (SA):** Uses a DG-generated plan as the initial solution and applies iterative random perturbations to attempt to find better candidate plans.

4. **Clustering + Daily Solver:** A novel two-stage approach that first clusters POIs into one cluster per day, and then solves each day's cluster as an independent routing problem using a daily solver. Two clustering strategies were tested:

(a) **Constrained K-Means + Refinement (CKMR):** Produces approximately equal-sized clusters using Constrained K-Means, followed by a refinement step that swaps border POIs between clusters to homogenize average fitness.

(b) **Integer Linear Programming (ILP):** Uses linear programming to produce clusters of approximately equal size and balanced average fitness.

As for daily solvers, four approaches were tested:

(a) **Naive Greedy (NG)** and **Dynamic Greedy (DG):** As described above, applied within each cluster.

(b) **Iterated Local Search (ILS):** A metaheuristic that starts with a base route and applies local search iteratively to improve it.

(c) **Iterative Branch-and-Bound (IBB):** Solves TSPs with an increasing number of POIs using Branch-and-Bound, iterating until no feasible solution can be found.

To evaluate these approaches, each algorithm was tasked with planning a 2-day itinerary for each of the three selected cities (Porto, Lisbon and Paris) for five real survey respondents with distinct demographic and preference profiles. Because the survey did not collect time-related preferences, these were manually assigned, and cover a diverse yet realistic range. Table 5.5 summarizes the user profiles selected and Table 5.6 lists the starting locations used for each city, representing a popular, centrally-located hotel.

Table 5.5: User profiles used to test itinerary generation algorithms.

User	1	2	3	4	5
Gender	Male	Male	Female	Female	Male
Age group	60-69	50-59	40-49	30-39	20-29
Number of ratings	141	141	106	85	96
History	3	5	4	5	4
Nature	5	4	5	3	3
Art	3	5	2	4	2
Culture	4	4	5	4	5
Leisure	4	3	4	4	1
pop_weight	0.1	0.2	0.3	0.4	0.4
pace	15	20	12	18	13
time_mod	1.3	1.2	1.3	0.7	0.8
max_walkable_mins	45	30	20	45	25
start_h	9	8	7	9	10
end_h	19	20	20	18	19

The baseline algorithm implements a variation of dynamic greedy with a fixed 0.9 weight on fitness and 0.1 on distance, effectively functioning as a naive greedy approach with a safeguard to avoid excessively distant POIs. The baseline did not use user-specific preferences, instead relying

Table 5.6: Start locations by city.

	Porto	Lisbon	Paris
Latitude	41.146840	38.712338	48.860724
Longitude	-8.603897	-9.139225	2.346429
Hotel	Hotel Ibis Porto Centro	Rossio Plaza Hotel	Novotel Paris Les Halles

on global average POI ratings as fitness, and ignored temporal constraints, applying fixed standard values shown in Table 5.7.

Table 5.7: Baseline standard parameter values.

	pop_weight	pace	time_mod	max_walkable_mins	start_h	end_h
value	0.7	15	1	30	9	19

Each of the 11 algorithmic approaches (NG, DG, SA, CKM+NG, CKM+DG, CKM+ILS, CKM+IBB, ILP+NG, ILP+DG, ILP+ILS, ILP+BB) and the baseline were used to generate 15 itineraries each (one per city per user) for a total of 180 itineraries. The algorithms' resulting average Quality and Completeness metrics, averaged from their outputted itineraries, are compared in Table 5.8.

Table 5.8: Comparison of itinerary generation algorithms

Method	**Avg. POIs/day**	**Avg. Quality (%)**	**Avg. Completeness (%)**
NG	15.33	91.10	80.27
DG	39.80	78.88	92.76
SA	39.80	78.88	92.76
CKM + NG	16.40	90.57	79.61
CKM + DG	22.00	87.11	84.69
CKM + ILS	6.60	86.30	53.59
CKM + IBB	16.47	89.36	80.77
ILP + NG	15.40	91.09	80.50
ILP + DG	21.00	87.14	84.05
ILP + ILS	5.73	86.35	51.17
ILP + IBB	16.80	89.00	79.95
Baseline	17.33	82.46	68.58

Several noteworthy patterns emerge from these results:

- The baseline achieved surprisingly high Quality, although its Completeness lagged behind. This likely reflects both the relatively low number of ratings in the dataset, causing average ratings (especially for less popular POIs) to approximate individual user ratings fairly well,

and the low distance weight, which allowed distant high-rated POIs to be chosen at the expense of Completeness.

- NG, despite its simplicity, performed remarkably well on both metrics, likely because many of the most desirable POIs are geographically central in all three cities and the max_walkable _mins guardrail prevents the further away ones from being selected, assuring high Completeness.
- DG selected an abnormally large number of POIs per plan. This may be due to its distance penalty favoring denser routing patterns, allowing the schedule to fit many short visits and thus raise Completeness at the expense of average visit duration.
- SA failed to improve upon DG, indicating that DG already approximates local optimal solutions quite well. Running SA on base solutions other than DG produced similar results.
- Introducing a clustering step improved Quality but slightly reduced Completeness across all solutions, likely because the geographic partitioning limits flexibility, by preventing the inclusion of nearby POIs located across cluster boundaries, which can hinder fully filling each day.
- CKM and ILP produced similar results, with CKM tending to yield slightly more POIs per day on average. This may be due to CKM focusing on minimizing intra-cluster distances, which creates more compact clusters, allowing for a higher amount of POI visits in each day.
- ILS performed unexpectedly poorly, producing very short itineraries with few POIs and low Completeness, likely due to inadequate seed itineraries.
- The clustering step had little effect when combined with NG, but a clearly positive effect when combined with DG, which is more sensitive to geographic layout and therefore more likely to "wander" toward good but far POIs, resulting in lower Completeness, a behavior that is restricted in a clustered setting.
- Overall, NG (with or without clustering), clustering+DG, and clustering+IBB produced the best itineraries. All algorithms except DG (without clustering), SA, and clustering+ILS met the predefined criterion of improving Quality by at least 5% without significantly reducing Completeness (in fact managing to improve this metric as well).

A final human-led qualitative analysis was conducted on the itineraries produced by the highest-performing algorithms (NG, CKM+NG, CKM+DG, CKM+IBB, ILP+NG, ILP+DG, ILP+IBB), as well as the baseline approach, in order to evaluate their realism, that is, the extent to which the resulting plans align with what a human tourist might plausibly plan and find enjoyable.

This analysis revealed several noteworthy patterns. First, although NG-based approaches (both with and without a preceding clustering step) achieved very high quantitative results in terms of

Quality and Completeness, they often selected POIs located in remote or untouristic areas solely due to their high ratings. While these selections respected the max_walkable_mins constraint, they resulted in itineraries that were geographically incoherent and unlikely to reflect a realistic tourist experience.

This issue is exemplified in the plan shown in Figure 5.8, which depicts day one of the Porto itinerary generated by ILP+NG for user 2. The route sends the user on a lengthy detour through peripheral urban areas with little touristic appeal. A similar issue was observed in the baseline approach because despite its basing on DG, the low weight applied on distance causes it to behave similarly to NG.

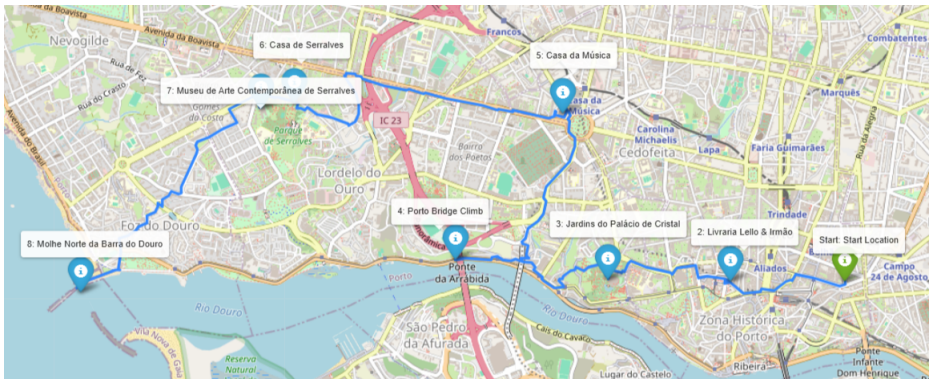

Figure 5.8: Day one of the Porto plan for user 2 made by ILP+NG.

In contrast, DG- and BB-based approaches performed markedly better in terms of geographic coherence and selection of POIs. Both consistently identified highly appropriate sets of attractions that align well with user preferences and touristic appeal. However, their distinct approaches to visitation ordering resulted in vastly different routes. DG, due to its greedy nature lacking high-level planning, frequently produced chaotic sequences that sent tourists back and forth across the city, retracing previously covered paths. By contrast, the Iterative Branch-and-Bound (BB) method, by treating each day as a Traveling Salesman Problem, was able to create optimal visiting orders. This yielded routes that were substantially more linear, efficient, and realistic.

This difference is clearly visible when comparing Figures 5.9 and 5.10, which show the day one plans for user 5 in Porto produced by CKM+IBB and CKM+DG respectively. In the CKM+IBB itinerary, Mercado do Bolhão, a POI close to the start location, is visited early in the day as part of a smooth linear route. Conversely, in the CKM+DG itinerary, the route begins by heading south, only looping back to Mercado do Bolhão at the end of the day, resulting in an inefficient and unintuitive path.

Taken together, the findings from both the quantitative evaluation and qualitative analysis support the hypothesis H1 introduced in Section 3.2: By incorporating user preferences and temporal constraints, it is possible to design an algorithm that improves itinerary Quality without compromising Completeness, when compared to a greedy baseline. Among all tested methods, the most

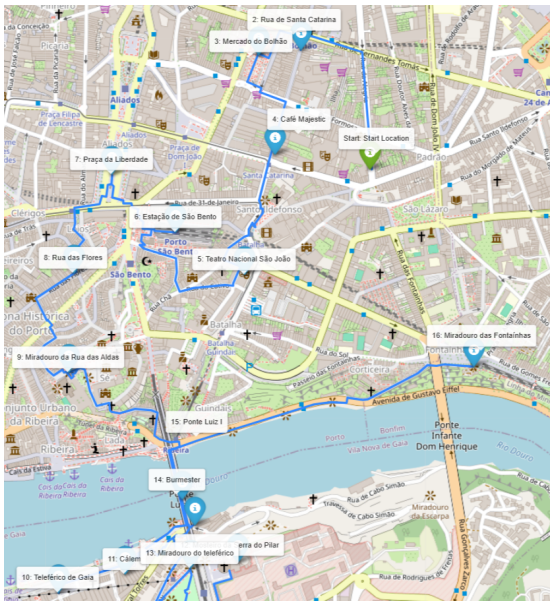

Figure 5.9: Day one of the Porto plan for user 5 made by CKM+IBB.

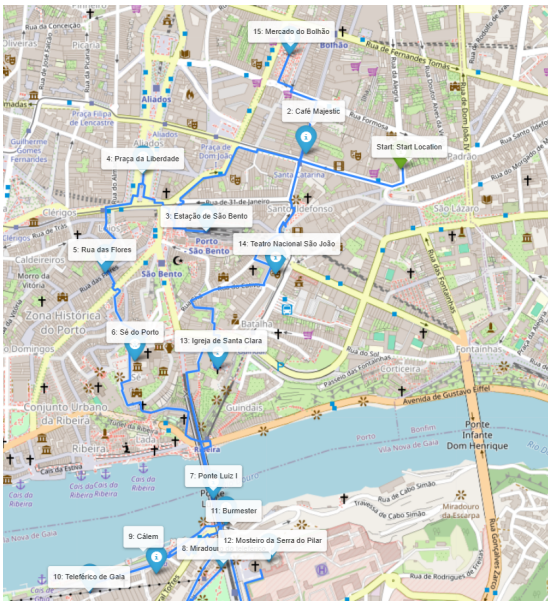

Figure 5.10: Day one of the Porto plan for user 5 made by CKM+DG.

effective overall, considering both the quantitative metrics and the human realism assessment, was the novel combination of Constrained K-Means clustering of POIs followed by an Iterative Branch-and-Bound daily planner (CKM+IBB).

Chapter 6

Conclusion

This research addressed the challenge of developing a comprehensive Tourism Recommender System (TRS) capable of generating personalized multi-day itineraries that balance user preferences with practical travel constraints. The work proposed and evaluated an implementation of a two-layered architecture that separates personalized rating prediction from itinerary optimization, enabling more effective handling of both personalization requirements and spatio-temporal constraints. It also employed a two-layered itinerary generation approach with POI clustering and a novel Iterative Branch-and-Bound algorithm.

The system was implemented and validated using real-world data from three major European cities (Porto, Lisbon, and Paris), incorporating automated POI importation from OpenStreetMap, Large Language Model-based feature estimation, and empirical user preference data collected through custom surveys. The evaluation demonstrated that the proposed approach consistently outperforms baseline methods in both itinerary quality and completeness, while maintaining geographic realism and practical feasibility.

The primary hypothesis H1, which states that a multi-day travel plan recommender system considering user preferences and travel time restrictions would provide higher-quality itineraries without compromising completeness, was conclusively validated through comprehensive quantitative and qualitative evaluation across multiple cities and user profiles.

6.1 Main contributions

This dissertation makes several contributions to the field of Tourism Recommender Systems which are summarized and categorized into Applicational, Scientific, Technological and Knowledge contributions below:

- **Applicational contributions:**
  - Prototype of a complete TRS incorporating a state-of-the art two-layered recommendation architecture and proven viable using real-world data.

- **Scientific contributions:**
  - A mathematic framework for judging touristic itinerary recommendations using the novel metrics of Quality and Completeness.
  - A rigorous mathematical definition of a specific version of the Tourist Trip Design Problem (TTDP).
- **Technological contributions:**
  - A novel high-performing algorithm called Constrained K-Means with Iterative Branch-and-Bound (CKM+IBB) for generating tourist itinerary recommendations, tested on real-world data.
  - A complete POI importation and LLM-based characteristics estimation pipeline using OpenStreetMap data.
- **Knowledge contributions:**
  - An extensive review of current literature on TRS design with a focus on systems integrating high degrees of personalization and rich constraints.
  - A comprehensive human-curated dataset of POIs in Porto, Lisbon, and Paris.
  - A detailed analysis of real-world tourist preference data, revealing new insights into their behavioural patterns and the relationship between preference dimensions.
  - A systematic comparison of the performance of different well-known optimization algorithms in generating tourist itineraries using real-world data.

Together, these insights advance the field of TRS design and may prove useful for future work on the field.

6.2 Future work

Several research directions emerge to build upon the findings and address the limitations of this work. Firstly, while the proposed solution aspires to universal applicability, the training dataset primarily reflects a young European population and European touristic landscapes, which may bias the results. Further research on more diverse tourist demographics and non-European destinations is needed to confirm or challenge its generalizability.

Beyond diversifying data, a broader spectrum of preferences and constraints could also be considered. These can be grouped into two categories. The first involves aspects that enhance personalization, such as accounting for user accessibility requirements, budget limitations, and social group dynamics in multi-person travel planning, particularly when traveling with children. The second involves aspects that enhance realism, including support for lunch breaks, multimodal

transport, and the ability to recommend accommodation and restaurant options alongside POIs.

Improvements to the system's engineering could also be pursued, vying a more commercial application. In this context, real-time route recalculation, a feature already explored in the literature, would be particularly useful in a real-world scenario. Likewise, incorporating long-term learning and improvement mechanisms, especially for POI feature estimation and personalized rating prediction would further increase system robustness and adaptability.

Finally, the methodological insights gained from combining hybrid filtering with constraint-aware optimization may hold value in related domains, such as event planning, educational curriculum design, or logistics optimization. Exploring these domains could help validate the broader applicability of the proposed approaches.

References

[1] Elisa Alén, Nieves Losada, and Trinidad Domínguez Vila. "The Impact of Ageing on the Tourism Industry: An Approach to the Senior Tourist Profile". In: *Social Indicators Research* 127 (Apr. 2015). DOI: 10.1007/s11205-015-0966-x.

[2] Saskia Putri Ananda, ZKA Baizal, and Gia Septiana Wulandari. "Improved Whale Optimization Algorithm with Variable Neighbourhood Search Strategy (WOA-VNS) in Solving Vehicle Routing Problem (VRP) for Recommending Multi-days Tourist Routes in Yogyakarta." In: *International Journal of Intelligent Engineering & Systems* 17.5 (2024). DOI: 10.22266/ijies2024.1031.53.

[3] John R. Mullin and. "The reconstruction of Lisbon following the earthquake of 1755: A study in despotic planning". In: *Planning Perspectives* 7.2 (1992), pp. 157–179. DOI: 10.1080/02665439208725745. eprint: https://doi.org/10.1080/02665439208725745. URL: https://doi.org/10.1080/02665439208725745.

[4] Rasuole Andruliene, Aida Macerinskiene, and Sigitas Urbonavicius. "Relations of tourist push and pull motivations with their activities: The case of Lithuania". In: *International journal of sustainable development and planning* 13.6 (2018), pp. 893–904. DOI: 10.2495/SDP-V13-N6-893-904.

[5] Maddala Lakshmi Bai, Rajendra Pamula, and Praphul Kumar Jain. "Tourist Recommender System using Hybrid Filtering". In: *2019 4th International Conference on Information Systems and Computer Networks (ISCON)*. 2019, pp. 746–749. DOI: 10.1109/ISCON47742.2019.9036308.

[6] Octavian Barna and Cristian Serea. "Historical Monuments as Tourist's Attractions in Europe". In: *Journal of Tourism – Studies and Research in Tourism* 23 (2023).

[7] Mahsa Bazrafshan et al. "Greater place attachment to urban parks enhances relaxation: Examining affective and cognitive responses of locals and bi-cultural migrants to virtual park visits". In: *Landscape and Urban Planning* 232 (2023), p. 104650. ISSN: 0169-2046. DOI: https://doi.org/10.1016/j.landurbplan.2022.104650. URL: https://www.sciencedirect.com/science/article/pii/S0169204622002997.

[8] Albert Berryman John and Ziegler. *Prompt Engineering for LLMs*. O'Reilly Media, 2024.

[9] Chenzhong Bin et al. "A personalized POI route recommendation system based on heterogeneous tourism data and sequential pattern mining". In: *Multimedia Tools and Applications* 78 (2019), pp. 35135–35156. DOI: 10.1007/s11042-019-08096-w.

[10] Juan Gabriel Brida, Marta Disegna, and Raffaele Scuderi. "Visitors of two types of museums: A segmentation study". In: *Expert Systems with Applications* 40.6 (2013), pp. 2224–2232. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2012.10.039. URL: https://www.sciencedirect.com/science/article/pii/S0957417412011633.

[11] Federica Cena et al. “Combining Genetic Algorithms and Temporal Constraint Satisfaction for Recommending Personalized Tourist Itineraries”. In: *International Conference of the Italian Association for Artificial Intelligence*. Springer. 2023, pp. 441–452. DOI: 10.1007/978-3-031-47546-7_30.

[12] Federica Cena et al. “How Personality Traits can be Used to Shape Itinerary Factors in Recommender Systems for Young Travellers”. In: *IEEE Access* 11 (2023), pp. 61968–61985. DOI: 10.1109/ACCESS.2023.3285258.

[13] Federica Cena et al. “Including the Temporal Dimension in the Generation of Personalized Itinerary Recommendations”. In: *IEEE Access* 12 (2024), pp. 112794–112809. DOI: 10.1109/ACCESS.2024.3441710.

[14] Gareth Davey. “What is museum fatigue?” In: *Visitor Studies Today* 8 (Jan. 2005), pp. 17–21.

[15] Maciej Debski. “Tourism habits and preferences – comparative analysis in selected European countries”. In: *Journal of Intercultural Management* 6 (Dec. 2014). DOI: 10.2478/joim-2014-0034.

[16] Maria Ferrão and Leandro Almeida. “Student’s access and performance in the Portuguese Higher Education: Issues of gender, age, socio-cultural background, expectations, and program choice”. In: *Avaliação: Revista da Avaliação da Educação Superior (Campinas)* 24 (Oct. 2019), pp. 434–450. DOI: 10.1590/s1414-40772019000200006.

[17] Damianos Gavalas et al. “An Efficient Heuristic for the Vacation Planning Problem”. In: *2019 IEEE 10th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*. 2019, pp. 0803–0808. DOI: 10.1109/UEMCON47517.2019.8993068.

[18] Keith Hollinshead. “Theme parks and the representation of culture and nature: The consumer aesthetics of presentation and performance”. In: Jan. 2009, pp. 269–289. DOI: 10.4135/9780857021076.n16.

[19] Feiran Huang, Jie Xu, and Jian Weng. “Multi-Task Travel Route Planning With a Flexible Deep Learning Framework”. In: *IEEE Transactions on Intelligent Transportation Systems* 22.7 (2021), pp. 3907–3918. DOI: 10.1109/TITS.2020.2987645.

[20] G. N. Iseh et al. “Cultural and Environmental Factors Influence on Tourists’ Choice of Destination in Delta State, Nigeria”. In: *European Journal of Hospitality and Tourism Research* 9.2 (2021), pp. 8–15. URL: https://ssrn.com/abstract=3828569.

[21] Priya Kumar. “The Kids Are (M)all Right: Youth Culture and Shopping Centers in Late Twentieth Century America”. Master’s thesis. Montreal, Quebec, Canada: Concordia University, June 2021.

[22] Didem Kutlu et al. “The Influence of World Heritage Sites on Tourism Dynamics in the EU 27 Nations”. In: *Sustainability* 16.20 (2024). ISSN: 2071-1050. DOI: 10.3390/su16209090. URL: https://www.mdpi.com/2071-1050/16/20/9090.

[23] Edyta Łaszkiewicz et al. “Greenery in urban morphology: a comparative analysis of differences in urban green space accessibility for various urban structures across European cities”. In: *Ecology and Society* 27.3 (2022). DOI: 10.5751/ES-13453-270322. URL: https://www.ecologyandsociety.org/vol27/iss3/art22/.

[24] Haowei Li et al. “A Contextual Multi-armed Bandit Approach to Personalized Trip Itinerary Planning”. In: *2024 IEEE International Conference on Smart Mobility (SM)*. 2024, pp. 55–60. DOI: 10.1109/SM63044.2024.10733530.

[25] Kaibo Liang et al. “Enhancing scenic recommendation and tour route personalization in tourism using ugc text mining”. In: *Applied Intelligence* 54.1 (2024), pp. 1063–1098. DOI: 10.1007/s10489-023-05244-6.

[26] Jesús Lorente. *Public Art and Museums in Cultural Districts*. Sept. 2018. ISBN: 9781351120302. DOI: 10.4324/9781351120302.

[27] Yuto Maejima, Hayato Horanai, and Liya Ding. “An Intelligent System for Safe and Satisfactory Individual Travel Tours in Tokyo”. In: *Machine Learning and Artificial Intelligence*. IOS Press, 2022, pp. 99–104. DOI: 10.3233/FAIA220430.

[28] Made Dwija Mahardika and Z. K. A. Baizal. “Recommender System for Tourist Routes in Yogyakarta Using Simulated Annealing Algorithm”. In: *2023 IEEE 8th International Conference for Convergence in Technology (I2CT)*. 2023, pp. 1–6. DOI: 10.1109/I2CT57861.2023.10126218.

[29] Fang Meng and Muzaffer Uysal. “Effects of Gender Differences on Perceptions of Destination Attributes, Motivations, and Travel Values: An Examination of a Nature-Based Resort Destination”. In: *Journal of Sustainable Tourism* 16.4 (2008), pp. 445–466. DOI: 10.1080/09669580802154231.

[30] Remigijus Paulavičius et al. “A novel greedy genetic algorithm-based personalized travel recommendation system”. In: *Expert Systems with Applications* 230 (2023), p. 120580. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2023.120580.

[31] Myrtille Picaud. “Framing performance and fusion: how music venues’ materiality and intermediaries shape music scenes”. In: *American Journal of Cultural Sociology* 10.2 (2022), pp. 285–315. DOI: 10.1057/s41290-022-00151-8.

[32] Yaniv Poria, Richard Butler, and David Airey. “Links between Tourists, Heritage, and Reasons for Visiting Heritage Sites”. In: *Journal of Travel Research* 43 (Aug. 2004), pp. 19–28. DOI: 10.1177/0047287504265508.

[33] Turismo do Porto. *Visit Porto - Districts - Atlantic*. 2025. URL: https://visitporto.travel/en-GB/districts/atlantic (visited on 06/10/2025).

[34] Turismo do Porto. *Visit Porto - Districts - Historic*. 2025. URL: https://visitporto.travel/en-GB/districts/historic (visited on 06/10/2025).

[35] Scott M. Pruett. “Formula for Success: How Las Vegas Became the Entertainment Capital of the World”. Master of Arts (MA) thesis. University of Nevada, Las Vegas, 2008. DOI: 10.25669/dvey-ipx2. URL: http://dx.doi.org/10.25669/dvey-ipx2.

[36] Alain Quemin. “Art and the City: Contemporary Art Galleries Districts in Paris from the End of the 19th Century until Today”. In: *Arts* 11.1 (2022). ISSN: 2076-0752. DOI: 10.3390/arts11010020. URL: https://www.mdpi.com/2076-0752/11/1/20.

[37] Greg Richards. “Tourism and the World of Culture and Heritage”. In: *Tourism Recreation Research* 25.1 (2000), pp. 9–17. DOI: 10.1080/02508281.2000.11014896. eprint: https://doi.org/10.1080/02508281.2000.11014896. URL: https://doi.org/10.1080/02508281.2000.11014896.

[38] Iuliana-Elena Rusu and Adrian Alexandrescu. “Travel Planning Optimization System Employing Genetic Algorithms with Multiple Parameters”. In: *2024 28th International Conference on System Theory, Control and Computing (ICSTCC)*. 2024, pp. 264–269. DOI: 10.1109/ICSTCC62912.2024.10744666.

[39] Apostolos Skotis, Christina Morfaki, and Christos Livas. “Identifying drivers of evaluation bias in online reviews of city destinations”. In: *International Journal of Information Management Data Insights* 3.2 (2023), p. 100184. ISSN: 2667-0968. DOI: https://doi.org/10.1016/j.jjimei.2023.100184. URL: https://www.sciencedirect.com/science/article/pii/S2667096823000319.

[40] Staff Reporter. “How Edinburgh’s Green Infrastructure Is Inviting Nature Enthusiasts Globally?” In: *The Edinburgh Reporter* (Aug. 2024). URL: https://theedinburghreporter.co.uk/2024/08/how-edinburghs-green-infrastructure-is-inviting-nature-enthusiasts-globally/.

[41] Barbara Staniscia. “Tourism and sustainability in the historic city of Rome: challenge or threat?” In: Nov. 2020, pp. 146–156. ISBN: 978-0-367-46797-5. DOI: 10.4324/9781003031192-16.

[42] Weeriya Supanich and Suwanee Kulkarineetham. “Personalized Tourist Attraction Recommendation System Using Collaborative Filtering on Tourist Preferences”. In: *2022 19th International Joint Conference on Computer Science and Software Engineering (JCSSE)*. 2022, pp. 1–6. DOI: 10.1109/JCSSE54890.2022.9836255.

[43] Pei Tan and Hairul Ismail. “A Review of Distance Decay Research Trends in Tourism from 2000-2020”. In: *Environment-Behaviour Proceedings Journal* 5 (July 2020), pp. 137–145. DOI: 10.21834/ebpj.v5i14.2275.

[44] Ernesto Tarantino, Ivanoe De Falco, and Umberto Scafuri. “A mobile personalized tourist guide and its user evaluation”. In: *Information Technology & Tourism* 21.3 (2019), pp. 413–455. DOI: 10.1007/s40558-019-00150-5.

[45] Takwa Tlili and Saoussen Krichen. “A simulated annealing-based recommender system for solving the tourist trip design problem”. In: *Expert Systems with Applications* 186 (2021), p. 115723. DOI: 10.1016/j.eswa.2021.115723.

[46] Chieh-Yuan Tsai, Lin-Yi Tai, and Chih-Chung Lo. “An LSTM based Personalized Travel Route Recommendation System”. In: *2023 Congress in Computer Science, Computer Engineering, & Applied Computing (CSCE)*. 2023, pp. 824–829. DOI: 10.1109/CSCE60160.2023.00140.

[47] *UNWTO World Tourism Barometer and Statistical Annex, January 2020*. 2020. DOI: 10.18111/wtobarometereng.2020.18.1.1. URL: http://dx.doi.org/10.18111/wtobarometereng.2020.18.1.1.

[48] Sebastian Vengesayi, Felix Mavondo, and Yvette Reisinger. “Tourism Destination Attractiveness: Attractions, Facilities, and People as Predictors”. In: *Tourism Analysis* 14 (Dec. 2009), pp. 621–636. DOI: 10.3727/108354209X12597959359211.

[49] Cristiana Vîlcea, Popescu Liliana, and Amalia Nita. “Historical Buildings and Monuments as Cultural Heritage In Situ—Perspectives from a Medium-Sized City”. In: *Heritage* 6 (May 2023), pp. 4514–4526. DOI: 10.3390/heritage6060239.

[50] Zhonghua Wang. “Intelligent recommendation model of tourist places based on collaborative filtering and user preferences”. In: *Applied Artificial Intelligence* 37 (Apr. 2023). DOI: 10.1080/08839514.2023.2203574.

[51] Bo Xu, Xiangqian Li, and Zhi-Ping Fan. “Selection and visiting sequence of daily attractions: Multi-day travel itinerary recommendation based on multi-source online data”. In: *Expert Systems with Applications* 250 (2024), p. 123895. ISSN: 0957-4174. DOI: 10.1016/j.eswa.2024.123895.

[52] Zahra Al-Zadjali. “The Significance of Art in Revealing a Culture’s Identity and Multiculturalism”. In: *Open Journal of Social Sciences* 12 (Jan. 2024), pp. 232–250. DOI: 10.4236/jss.2024.121015.

[53] Jiale Zhang et al. “Encoder-Decoder Based Route Generation Model for Flexible Travel Recommendation”. In: *IEEE Transactions on Services Computing* 17.3 (2024), pp. 905–920. DOI: 10.1109/TSC.2024.3376231.

[54] Zhiwei Zhang et al. “Solo traveler typology based on personal value: Incorporating cluster analysis into means-end chain”. In: *Tourism Management Perspectives* 51 (2024), p. 101247. DOI: 10.1016/j.tmp.2024.101247.

[55] Xiaolei Zhong et al. “A Personalized Recommendation Algorithm for Tourist Attractions Using User Behavior Data”. In: *International Journal of Computer Science and Information Technology* 3 (July 2024), pp. 242–254. DOI: 10.62051/ijcsit.v3n2.28.

[56] Xiao Zhou et al. “Intelligent Tourism Recommendation Algorithm based on Text Mining and MP Nerve Cell Model of Multivariate Transportation Modes”. In: *IEEE Access* 9 (2021), pp. 8121–8157. DOI: 10.1109/ACCESS.2020.3047264.

Appendix A

Appendix

A.1 Query for POI selection from OSM database

Below, in Listing A.1, is the OverpassQL query to import POIs of touristic relevance from the OSM database for the example city of Porto, Portugal.

Listing A.1: OverpassQL search query for Porto Portugal.

```
[out:json][timeout:90];

area["name"="Porto"]["boundary"="administrative"]["admin_level"=7]->.porto;
area["name"="Maia"]["boundary"="administrative"]["admin_level"=7]->.maia;
area["name"="Matosinhos"]["boundary"="administrative"]["admin_level"=7]->.
    matosinhos;
area["name"="Vila do Conde"]["boundary"="administrative"]["admin_level"=7]->.vc;
area["name"="Santa Marinha e Sao Pedro da Afurada"]["boundary"="administrative"][
    "admin_level"=8]->.vng;

(.porto;.maia;.matosinhos;.vc;.vng;)->.greaterPorto;

area["name"="Area Metropolitana do Porto"]["boundary"="statistical"]->.portoSR;

(
nwr(area.greaterPorto)["tourism"]["tourism"!~"guest_house|hotel|apartment|hostel|
    motel|camp_site|information|camp_pitch|caravan_site|chalet|wilderness_hut|
    tours|no"];
nwr(area.greaterPorto)["leisure"="park"]["wikidata"]["wikipedia"];
nwr(area.portoSR)["landuse"="forest"]["wikidata"]["wikipedia"];
nwr(area.portoSR)["leisure"="nature_reserve"];
nwr(area.portoSR)["boundary"="national_park"];
);

out geom;
```

A.2 Python code for evaluating if two POIs should be merged

Listing A.2 presents the Python code responsible for evaluating whether two POIs should be merged. Since the language used in the `name` attribute of an OSM element may be either English or the local language of the country in which it is located, the `stopwords_set` argument of the function in line 14 should include the stopwords of all relevant languages. The function `haversine`, defined in line 24, applies the Haversine formula to compute the distance in metres between POIs, using either their coordinates (for *Nodes*) or the midpoint of their geometry (for *Ways* and *Relations*).

Listing A.2: Python code for evaluating if two POIs should be merged.

```
1  from fuzzywuzzy import fuzz
2
3  name_similarity_threshold = 95
4  distance_threshold = 100
5  crucial_categories = ["museum", "theme_park", "gallery"]
6
7  def remove_stopwords(string, stopwords_set):
8    word_list = []
9    for word in simple_preprocess(string):
10     if word not in stopwords_set:
11       word_list.append(word)
12   return " ".join(word_list)
13
14 def compare_attribute_similarity(poi1_attribute, poi2_attribute, method,
       stopwords_set):
15   poi1_processed = remove_stopwords(str(poi1_attribute).lower(), stopwords_set)
16   poi2_processed = remove_stopwords(str(poi2_attribute).lower(), stopwords_set)
17   if method == "token_sort_ratio":
18     ratio = fuzz.token_sort_ratio(poi1_processed, poi2_processed, force_ascii=
         True, full_process=True)
19     return ratio
20   if method == "partial_ratio":
21     ratio = fuzz.partial_ratio(poi1_processed, poi2_processed)
22     return ratio
23
24 def should_merge(poi1, poi2, stopwords_set):
25   distance = haversine(poi1, poi2)
26   if distance <= distance_threshold:
27     ratio_names = compare_attribute_similarity(poi1.name, poi2.name,  "
         token_sort_ratio", stopwords_set)
28     if ratio_names >= name_similarity_threshold:
29       return True
30
31     if poi1.addr_name is not None:
```

```
32        if compare_attribute_similarity(poi1.addr_name, poi2.name,  "
              token_sort_ratio", stopwords_set) >= name_similarity_threshold:
33          return True
34
35      if poi2.addr_name is not None:
36        if compare_attribute_similarity(poi2.addr_name, poi1.name,  "
              token_sort_ratio", stopwords_set) >= name_similarity_threshold:
37          return True
38
39      if compare_attribute_similarity(poi1.name, poi2.name, "partial_ratio",
            stopwords_set) >= name_similarity_threshold:
40        specific = poi1 if len(poi1.name) > len(poi2.name) else poi2
41        if specific.tourism in crucial_categories:
42          return False
43        else:
44          return True
45    return False
```

A.3 Code for evaluating if a POI should be deleted

Listing A.3 presents the Python code responsible for evaluating whether a POI should be deleted for being contained within another POI. When both POIs are of type *Way* or *Relation*, the auxiliary function `multipolygon_contains_multipolygon` ensures that the target POI is deleted only if all elements of its associated area are fully contained within the other POI.

Listing A.3: Python code for evaluating if a POI should be deleted

```
1   from shapely.geometry import Point, Polygon
2   from services.poi_temp_service import get_polygons_from_poi
3
4   def multipolygon_contains_multipolygon(greater_multipolygon, lesser_multipolygon)
        :
5     for lesser_polygon_area in lesser_multipolygon:
6       lesser_polygon = Polygon(lesser_polygon_area["nodes"])
7       lesser_polygon_is_contained_in_greater_multipolygon = False
8       for greater_polygon_area in greater_multipolygon:
9         greater_polygon = Polygon(greater_polygon_area["nodes"])
10        if greater_polygon.contains(lesser_polygon):
11          lesser_polygon_is_contained_in_greater_multipolygon = True
12          break
13      if not lesser_polygon_is_contained_in_greater_multipolygon:
14        return False
15    return True
16
17  def should_delete(pois, target_poi):
```

```
    target_id = target_poi.get("poi_temp_id")
    for area_poi in pois:
      area_poi_temp_id = area_poi.get("poi_temp_id")
      if target_id != area_poi_temp_id:
        if target_poi.get("wikidata") is None:
          area_polygons = get_polygons_from_poi(area_poi_temp_id)
          target_type = target_poi.get("type")
          if target_type == "node":
            target_point = Point(target_poi.get("lat"), target_poi.get("lon"))
            for area_polygon in area_polygons:
              polygon = Polygon(area_polygon["nodes"])
              if polygon.contains(target_point):
                return True
          else:
            target_polygons =  get_polygons_from_poi(target_id)
            if target_polygons is not None:
              if multipolygon_contains_multipolygon(area_polygons,  target_polygons
                  ):
                return True
    return 0
```

A.4 POI feature estimation full prompt

Below is the full prompt submitted to the LLM for estimating the values of the six POI feature dimensions for the POIs of any city. The placeholder `city_name` is replaced by the name of the target city before submission. The list of POIs is then appended to the prompt in the format: poi_name1 (category1), poi_name2 (category2), . . . , poi_nameN (categoryN).

> "Rate the following POIs in the city of city_name from 1 to 10 on 6 different dimensions based on their characteristics: History, Nature, Art, Culture, Leisure, Popularity. Before grading, consider the following meanings of each dimension: 1. History: POIs with lower history scores are more recent and have little content in them or in their structure that is of historical relevance to the city. POIs with higher history scores are often quite old and are of historical significance to the city or showcase the history of the city. 2. Nature: POIs with lower nature scores are indoors and have little to no greenery to be seen. POIs with higher nature scores are outdoors and allow visitors to see or interact with nature and landscapes that include natural elements. 3. Art: POIs with lower art scores have little about them of artistic significance, neither the POI itself or anything inside it would qualify to be showcased in an art book or artistic exposition. POIs with higher art scores are artistically or architecturally significant in themselves or, showcase pieces or interior details that are artistically rich. 4. Culture, or more accurately Local Culture:

```
POIs with higher culture scores are POIs that one would go to if their
objective is to learn about the culture of the city or engage in relevant
local cultural activities.  POIs with lower culture are more universal
in nature and aren't that particular to the city in question.  5.  Leisure:
POIs with lower leisure scores require more intellectual effort to
understand and are less of what would conventionally be considered
fun.  POIs with higher leisure scores are more relaxing, entertaining
and leisurely and generally require less effort to enjoy.  6.  Popularity:
POIs with lower popularity scores are usually more niche and known
by locals but wouldn't show up on internet lists of the top things
to do in the city.  POIs with higher popularity scores are the widely
known highlights of a city, the must-see sights that every tourist
will visit on their first time in the city and that always show up
on internet lists of top things to do in the city.  In addition, estimate
the Time attribute, which represents the average duration of a visit
to the POI in minutes.  Return a response exactly in the following
json format:  { "POI 1 name" :  {"dimension 1" :  value, "dimension
2" :  value ...}, "POI 2 name" :  {"dimension 1" :  value, "dimension
2" :  value ...} Include nothing else in your response except for the
names of the POIs, the names of the dimensions and the POI's grades
for each of the 6 dimensions in the format requested.  Do not include
explanations, do not include a preamble saying "Here's a rating of
the given POIs".  Do not include the POI category in your response.
The only information included in the response are the POI names, the
names of the dimensions and the POI's score in each dimension as well
as the Time attribute.  If you cannot find information on a POI do
not give it a popularity rating higher than 4.  If you encounter POIs
with the same name, rate them separately, do not join them into a single
POI. If you see a repeated POI in the list of POIs, do not eliminate
the repetition, instead rate both instances of the POI as if they were
individual attractions.  In other words, if you receive 10 POIs as
input, return 10 POIs in the output, no more, no less.  The POIs and
respective categories (in parentheses) are:  "
```

A.5 Code for cleaning survey data

Below, in Listing A.4, is the Python code to clean the raw data collected from survey participants to make it usable for prediction model training.

Listing A.4: Python code for cleaning survey data

```
1  users = pd.DataFrame(get_all_users())
2  user_preferences = pd.DataFrame(get_all_user_preferences())
3  pois = pd.DataFrame(get_pois_by_city([1, 2, 12]))
```

```
poi_average_ratings = pd.DataFrame(get_all_poi_average_ratings())
user_ratings = pd.DataFrame(get_all_survey_visited_ratings())

user_with_preferences = users.merge(user_preferences, on="user_id")
poi_average_ratings.drop(["poi_id"], axis=1, inplace=True)

poi_with_average_rating = pois.merge(poi_average_ratings, on=["osm_type", "
    osm_ref"])
poi_with_average_rating["average_rating"] = (
poi_with_average_rating["rating_total"] / poi_with_average_rating["rating_amount"
    ]
)

df = user_ratings.merge(poi_with_average_rating, on=["poi_id"]).merge(
    user_with_preferences, on=["user_id"])

df.drop(["survey_visited_ratings_id", "poi_id", "poi_temp_id", "lat", "lon"],
    axis=1, inplace=True)
df.drop(["city_id", "user_accessibility", "walking", "pace", "start_day_time", "
    lunch_time"], axis=1, inplace=True)
df.drop(["lunch_duration", "dinner_time", "dinner_duration", "end_day_time", "
    current"], axis=1, inplace=True)
df.drop(["name_en", "name_pt", "wikipedia", "source", "photo_url", "
    origin_country_id"], axis=1, inplace=True)
df.drop(["student", "residence_country_id", "user_preferences_id", "user_cost", "
    user_popularity"], axis=1,
inplace=True)
df.drop(["rating_total", "rating_amount", "name_y", "manual_add", "
    poi_average_rating_id"], axis=1, inplace=True)
df.drop(
["missing_photo_url", "photo_blob", "check", "poi_opening_hours", "poi_grade", "
    poi_accessibility", "poi_cost"],
axis=1, inplace=True)
df = df.rename(columns={"name_x": "poi_name"})
df = df.rename(columns={"rating": "target_rating"})
df = df.rename(columns={"category": "poi_category"})
df = df.rename(columns={"average_rating": "poi_score"})
df = df.rename(columns={"age": "user_age"})
df = df.rename(columns={"gender": "user_gender"})
df['osm_id'] = df['osm_type'].astype(str) + "_" + df['osm_ref'].astype(str)
```

A.6 Code for generating travel plans

Below, in Listing A.5, is the Python code responsible for generating travel plans using the CKM+IBB algorithm.

Listing A.5: Python code for generating multi-day travel plans

```
def create_ranked_list(user_id, city_id, popularity_weight, time_modifier):
  predicted_ratings = predict_user_poi_scores(user_id, city_id)

  predicted_rating_data = apply_metric(predicted_ratings, popularity_weight)
  predicted_rating_data = apply_time_modifier(predicted_rating_data,
      time_modifier)

  return predicted_rating_data


def create_plan(user_id, city_id, popularity_weight, time_modifier, days,
    start_time, end_time, start_loc, pace, max_walkable_minutes, save = False,
    file_name = None, visualize = False):
  ranked_pois_df = create_ranked_list(user_id, city_id, popularity_weight,
      time_modifier)
  clusters = ckm_clusterer(ranked_pois_df, len(days))
  day_per_cluster = decide_day_per_cluster(clusters, days)
  clusters = refine_clusters(clusters, day_per_cluster)

  plans = []
  for day, cluster in day_per_cluster:
    single_cluster = clusters[clusters["cluster"] == cluster].copy()
    single_day_plan = iterative_bb_planner(cluster, day, start_time, end_time,
        start_loc, pace, max_walkable_minutes)
    plans.append(single_day_plan["plan"])

  print_plan(plans)

  if save:
    out_path = Path(PLAN_OUTPUT_PATH + file_name + ".json")
    with out_path.open("w", encoding="utf-8") as f:
      json.dump(plan, f, indent=2, ensure_ascii=False)
  if visualize:
    out_path = Path(VISUALS_OUTPUT_PATH + file_name)
    create_visuals(plan, out_path)


def iterative_bb_planner(cluster, day, start_time, end_time, start_loc, pace,
    max_walkable_minutes):
  best_plan = None
  n_pois = 2

  while n_pois <= len(cluster):
    answer = bb(cluster.head(n_pois), day, start_time, end_time, start_loc, pace,
        max_walkable_minutes)
    feasible = answer["feasible"]
    if feasible:
```

```
41        best_plan = answer
42        n_pois += 1
43      else:
44        cluster = cluster.drop(cluster.index[n_pois - 1])
45
46    return best_plan
```

A.7 Code to assign travel days to clusters

Below, in Listing A.6, is the full Python code used to decide which cluster of POIs to assign to each day of a multi-day trip. The code assumes an equal amount of trip days and clusters. It also employs the function `is_open_day` which checks if a given POI is open on a given day. The entrypoint is the function `decide_day_per_cluster`.

Listing A.6: Python code for assigning clusters of POIs to trip days

```
1  def check_lone_single_candidate_day(candidate_days_per_cluster, days):
2    candidate_clusters_per_day = {}
3    for day in days:
4      candidate_clusters_per_day[day] = []
5
6    i = 0
7    while i < len(candidate_days_per_cluster):
8      for candidate_day in candidate_days_per_cluster[i]:
9        candidate_clusters_for_day = candidate_clusters_per_day.get(candidate_day)
10       candidate_clusters_for_day.append(i)
11       candidate_clusters_per_day[candidate_day] = candidate_clusters_for_day
12     i+=1
13
14   changed = False
15   for day in candidate_clusters_per_day:
16     if len(candidate_clusters_per_day.get(day)) == 1:
17       cluster = candidate_clusters_per_day.get(day)[0]
18       if len(candidate_days_per_cluster[cluster]) != 1:
19         candidate_days_per_cluster[cluster] = [day]
20         changed = True
21
22   return [candidate_days_per_cluster, changed]
23
24
25 def redo_candidates_for_clusters(candidate_days_per_cluster, days):
26   used_days = []
27   while True:
28     new_used = None
29     for candidate_days in candidate_days_per_cluster:
```

```
30        if len(candidate_days) == 1 and not (candidate_days[0] in used_days):
31          new_used = candidate_days[0]
32          used_days.append(candidate_days[0])
33
34      if new_used is None:
35        candidate_days_per_cluster, changed = check_lone_single_candidate_day(
              candidate_days_per_cluster, days)
36        return [candidate_days_per_cluster, len(used_days) == len(
              candidate_days_per_cluster)]
37
38      i=0
39      while i < len(candidate_days_per_cluster):
40        candidate_days = candidate_days_per_cluster[i]
41        if len(candidate_days) != 1:
42          candidate_days = [item for item in candidate_days if item != new_used]
43          candidate_days_per_cluster[i] = candidate_days
44        i+=1
45
46
47  def restrict_candidates(df, days):
48    done = False
49    candidate_days_per_cluster = []
50    for _ in days:
51      candidate_days_per_cluster.append(days)
52
53    for poi in df.itertuples(index=True, name="Row"):
54      poi_full = poi._asdict()
55      if not poi_full:
56        print("Couldn't get full POI")
57        continue
58
59      poi_cluster = poi_full["cluster"]
60      candidate_days_for_cluster = candidate_days_per_cluster[poi_cluster]
61
62      new_candidate_days_for_cluster = candidate_days_for_cluster.copy()
63      for day in candidate_days_for_cluster:
64        if not is_open_day(poi_full, day):
65          new_candidate_days_for_cluster = [item for item in
                new_candidate_days_for_cluster if item != day]
66
67      if len(new_candidate_days_for_cluster) == 0:
68        continue
69
70      candidate_days_per_cluster[poi_cluster] = new_candidate_days_for_cluster
71      candidate_days_per_cluster, done = redo_candidates_for_clusters(
            candidate_days_per_cluster, days)
72      if done:
73        break
74    return candidate_days_per_cluster, done
```

```
75
76
77  def clusters_sorted_by_size(df):
78    counts = df['cluster'].value_counts(ascending=False)
79    return counts.index.tolist()
80
81
82  def decide_day_per_cluster(df, days, baseline=False):
83    if baseline:
84      df = df.sort_values(by="average_rating", ascending=False)
85    else:
86      df = df.sort_values(by="metric", ascending=False)
87
88    candidate_days_per_cluster, done = restrict_candidates(df, days)
89    if done:
90      pairs = [(candidate_days[0], cluster) for cluster, candidate_days in
          enumerate(candidate_days_per_cluster)]
91      return sorted(pairs, key=lambda x: x[0])
92    clusters_by_size = clusters_sorted_by_size(df)
93
94    used_days = []
95    for cluster in clusters_by_size:
96      candidate_days = candidate_days_per_cluster[cluster]
97      if len(candidate_days) == 1:
98        used_days.append(candidate_days[0])
99      else:
100       chosen_day = [day for day in sorted(candidate_days) if day not in used_days
              ][0]
101       used_days.append(chosen_day)
102       candidate_days_per_cluster[cluster] = [chosen_day]
103
104   pairs = [(candidate_days[0], cluster) for cluster, candidate_days in enumerate(
          candidate_days_per_cluster)]
105   return sorted(pairs, key=lambda x: x[0])
```