



POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Creation of a Realistic Dataset for the evaluation of a Travel Offer Recommender System

TESI DI LAUREA MAGISTRALE IN
MATHEMATICAL ENGINEERING - INGEGNERIA MATEMATICA

Author: **Melis YILMAZ**

Student ID: 990334

Advisor: Rossi Matteo Giovanni

Academic Year: 2023-2024

Contents

Contents	iii
List of Figures	v
List of Tables	vii
Abstract	i
Abstract in lingua italiana	iii
Acknowledgements	v
Introduction	1
1 Background and Related Work	5
1.1 Related Work	5
1.2 Background	7
2 Dataset Generation Method	13
2.1 User profile Generation	13
2.2 Travel Offer Generation	16
3 Experimental Results	21
3.1 First Batch of Experiments	21
3.2 Second Batch of Experiments	24
3.3 Discussion	28
4 Conclusions and Future Work	31
Bibliography	33
A Appendix A	37

List of Figures

1.1	Main features used for data generation along with their type and category. [10]	9
2.1	Workflow of the generation dataset	19
3.1	The accuracies of each algorithm for all the users	23
3.2	The ratio of different algorithms is used as the best mode	24
3.3	The accuracies of each algorithm for all the users	27
3.4	The ratio of different algorithms is used as the best mode	28
A.1	Pseudo-code of Learner Algorithm [10]	37
A.2	Pseudo-code of Cluster Model Training Algorithm [10]	38
A.3	Pseudo-code of Ranker Algorithm [10]	38

List of Tables

2.1	Scenarios	14
2.2	Probabilities for each parameter according to scenarios	15
2.3	Parameter and Data Table	16
2.4	Numerical Values for Offer and User	18
3.1	User Profiles and Scenarios	22
3.2	Probabilities for each parameter according to scenarios	25
3.3	User Profiles and Scenarios	26

Abstract

The travel industry has been influenced by technological developments and has changed and grown day by day. Personalizing and improving user experiences has become important for this sector, which appeals to more users every day. This study aims to evaluate a system called "The Hybrid Offer Ranker" (THOR) in order to test its performance to understand user preferences in the travel industry and provide personalized travel recommendations based on these preferences. THOR is designed as a hybrid, personalized recommendation system to provide users with travel recommendations based on their past preferences. The main purpose of this study is to test the classification performance of the THOR system on users by creating a realistic data set. While creating this data set, users with different profiles were created. The route information created for each user was taken from the Google Maps API. A value was assigned to whether users received each offer or not, and the performance of the system in predicting this value was examined. As a result, this study is an important one in terms of testing THOR, which was created to increase the effectiveness of recommendation systems in the travel industry and improve user experience based on realistic data to measure whether the system correctly understands user preferences and is successful in providing personalized travel recommendations.

Keywords: Travel Industry, Recommendation Systems, Personalization, Data Analytics, User Preferences

Abstract in lingua italiana

Negli ultimi anni il settore dei viaggi ha visto un notevole sviluppo tecnologico e un crescente impatto delle applicazioni informatiche. Personalizzare e migliorare l'esperienza dell'utente è diventato importante per questo settore, che attira ogni giorno sempre più utenti. Questo studio mira a valutare un sistema chiamato "The Hybrid Offer Ranker" (THOR) al fine di testarne le prestazioni per comprendere le preferenze degli utenti nel settore dei viaggi e fornire consigli di viaggio personalizzati basati su queste preferenze. THOR è progettato come un sistema di consigli ibrido e personalizzato per fornire agli utenti consigli di viaggio basati sulle loro preferenze passate. Lo scopo principale di questo studio è testare le prestazioni di classificazione del sistema THOR sugli utenti creando un set di dati realistico. Durante la creazione di questo set di dati, sono stati creati utenti con profili diversi. Le informazioni sul percorso create per ciascun utente sono state prese dall'API di Google Maps. È stato assegnato un valore al fatto che gli utenti abbiano ricevuto o meno ciascuna offerta ed è stata esaminata la prestazione del sistema nel prevedere questo valore. Di conseguenza, questo studio è uno studio importante in termini di test THOR, che è stato creato per aumentare l'efficacia dei sistemi di raccomandazione nel settore dei viaggi e migliorare l'esperienza dell'utente, sulla base di dati realistici, per misurare se il sistema comprende correttamente le preferenze dell'utente e ha successo nel fornire consigli di viaggio personalizzati.

Parole chiave: Industria dei viaggi, Sistemi di raccomandazione, Personalizzazione, Analisi dei dati, Preferenze dell'utente

Acknowledgements

Thank you very much to everyone who helped me complete this project and the university. First of all, I would like to thank my advisor, Professor Matteo Rossi Giovanni, who guided and supported me at every stage of this project.

I would like to express my sincere gratitude to my family who stood by me with their understanding and encouragement during this difficult process. Their support is invaluable to me.

Finally, I would like to express my endless gratitude to Assistant Professor Dr. Erkan Işıklı, who helped me in my decision to come to Politecnico di Milano and supported me throughout the university.

Introduction

Technological developments and the development of the digital world have created changes in every aspect of our lives. With the rapid development and change of technology, the number of internet users is increasing day by day. Fields such as data science, machine learning, and artificial intelligence have also been affected by this increase and change. Data, when analyzed correctly, can provide valuable information and meaningful insights. Data science combines statistics, data analysis, and related methodologies, thus allowing us to understand and analyze real events. [22]. Data science includes the process of collecting, analyzing, cleaning, and transforming large amounts of data into meaningful results. The use of statistics, machine learning, and artificial intelligence techniques allows us to discover valuable information hidden in data. Machine learning is an important sub-branch of the field of artificial intelligence. This technology enables computer systems to perform certain tasks automatically by learning from experience and data. Businesses can use machine learning to extract valuable insights from large amounts of data. Machine learning is of great importance in matters such as predicting marketing trends, understanding customer behavior, and making demand forecasts. Machine learning provides personalized product and service recommendations by understanding customer preferences and behavior. This increases customer satisfaction and encourages loyalty. The increase in the number of users has also led to an increase in the amount of information on the internet. Therefore, a lot of information is available on the internet. This information may not always be of interest to users. This may cause a loss of time in research. Recommendation systems aim to learn users' preferences and predict them later. This can be helpful when users experience search or selection issues. Considering the rapid growth of information available on the Internet, users have many options.

A recommendation system adapts the results and ranks them according to each user's preferences, habits, and personal needs. Today, recommendation systems are used in e-commerce applications to support many consumers. For example, Amazon.com was the first company to use data mining and collaborative filtering to find users with similar preferences and predict what products they will be interested in. Therefore, customizing options is an important strategy to provide a better user experience. Recommender

systems offer value by presenting users with relevant options, even when their needs may not be explicitly articulated or known. [23]. Recommendation systems are used in many areas. For example, on e-commerce platforms, it can analyze the user's past shopping preferences and searches and suggest similar products that may be of interest to the user. On movie platforms, it can be used to recommend new content based on the user's viewing history.

Data science has affected the travel industry, like all sectors. Travelers turn to the internet to obtain the necessary information and decide on their routes before going on a trip. Recently, in the travel industry, the user's preferences can be determined by analyzing which solutions the user prefers, and thus suitable solutions can be offered by analyzing and classifying similar users. Today, transportation is no longer limited to a single mode. There are various means of travel, such as walking, cycling, automobiles, and public transportation. Users may give importance to different parameters, such as cost and time. They can search for different modes of transportation depending on the duration and purpose of their planned trip. They want to choose the most suitable mode of transportation. The concept of multimodal transportation combines transportation modes, allowing users to have different options in determining their routes. Users choose the one that suits them best from the different options. [17]. Travel recommendation systems typically evaluate factors such as the user's past travel experiences, budget, location, and time. Using this information, the system provides personalized travel recommendations and suggests solutions to meet the user's expectations. However, for individuals seeking to enhance their decision-making process, thorough exploration of options through methods like depth-first search, knowledge acquisition, and product selection becomes essential. In the tourism sector, two notable applications of recommender systems are TripMatcher by TripleHop, utilized on www.ski-europe.com, and MePrint by VacationCoach, featured on travelocity.com. These systems emulate human-like interactions to assist users in streamlining their search for holiday destinations. [20]. As a result, travel solutions based on user preferences play an important role in the travel industry by increasing customer satisfaction and making the experience more personal. Also, recommendation systems are widely recognized as among the most efficient methods for alleviating information overload. Take, for instance, a travel recommender system that could: (1) provide users with immersive experiences of destinations; (2) streamline the destination selection process by catering to individual needs and preferences; (3) enable convenient on-the-go purchasing of products and services; and (4) foster repeat visits to destinations through effective customer relationship management strategies. [23]. The introduction of the purchase recommendation system makes consumers feel perceived newness, and they encounter

various new product information that they did not expect. The uncertainty about the product becomes lower due to the cumulative effect of the information, and the possibility of purchasing increases. With the increase in information load in the travel industry, the Hybrid Offer Ranker (THOR) system was created using hybrid recommender systems to provide personalized travel recommendations to users. The THOR system aims to ease the difficulties of travelers in determining and choosing the offers they prefer by learning their travel preferences. Thor specifically assigns a preference model to each traveler because this model is important for ranking travel offers based on personal preferences. To test the performance of this system, the performance of the algorithms was measured by creating a suitable data set according to some rules. The results at the end of the study were positive. This study seeks solutions to two problems:

- Creating a realistic dataset to test the algorithms.
- Performing the dataset on classification algorithms.

The first of these is to create a realistic data set to test the algorithms. Since there was no data set suitable for this context, a realistic data set was created to test the study. The other problem that the study seeks to answer is how this data set, created in a different way, will perform on classification algorithms. The other sections of the article are organized as follows: Section 1.1 covers contains information about other studies on recommendation systems. Section 2 includes how THOR was designed and a description of the algorithms used. In Section 2.1, the steps followed when creating the new data set are detailed. Section 3 contains the results obtained by testing the new data set on the algorithm. Finally, Section 4 contains a summary of the study results and information about what studies can be done in the future.

1 | Background and Related Work

This section describes the article that inspired the relevant work and previous work on recommendation systems. In the first part, previous studies on recommendation systems and the methods used are discussed. In the second section, the procedure followed and the algorithms used in the article that inspired this study are mentioned.

1.1. Related Work

With the influence of globalization and technological advances, there is a competitive environment in many sectors. In a rapidly changing world, users also change. Their behavior may also change under similar conditions. It has become important for industries to learn users' current purchasing preferences to be able to recognize their changing preferences and take action accordingly. Because recommendation systems play an important role in increasing performance, Recommendation systems are created based on the user's past preferences. In addition, recommendation systems not only increase user satisfaction but also increase the user retention rate of practitioners. [5]. For example, YouTube uses the concept of "long click rates", where clicks on recommendations are counted only after the user has watched part of a given video, as reported in [5]. Similarly, Netflix uses "take rate", a metric that captures how many times a video or movie is actually played after being selected from a recommendation [7]. In an internet blog, it is stated that its users discover around 75% of shows through algorithmic recommendations [2]. According to their experiments, lifts are significant when recommendations based on a personalized strategy increase take rates compared to recommendations based on popularity. In a much smaller marketplace for electronic devices, Lerche mentioned that using alternative strategies could increase the referral rate to an external marketplace by more than 250% [14]. For improving recommendation systems, various studies have been carried out using different approaches. For example, Yajie Hu et al [9] talk about a new approach for song recommendation systems. In this approach, user preferences and the 'forgetting curve' are used. The forgetting curve shows that the ability to retain information depends on the time elapsed after learning. This method was not suitable for cold users. At

the same time, as the data increased, it became more complex and could not capture the changes in preferences. Bo Wang et al.'s [25] study developed a personalized recommendation algorithm. This algorithm takes into account the user's purchasing and comparison behavior. It is aimed at solving diversity and scalability problems by using similarity measures. With collaborative filtering recommendation systems, recommendations are provided based on data obtained from products evaluated by different users. This algorithm aims to give appropriate recommendations to the user by deciding how similar a user is to other users. There are three principles determined by Terveen et al: [24]: There is similarity between users, the system provides easy filtering, and algorithms can detect similar users. Content-based recommendation systems are based on the similarity between products. Similarity matrices are established between the products, and similar products are decided accordingly. For example, Kassak used cosine similarity between products to measure similarity [11]. Data mining can also be used in recommendation systems. In this context, data mining involves studies to extract meaningful and useful information from unstructured data [26]. For example, in the travel industry, analyzing information such as demographic information, spending data, and purchasing preferences of users can help create personalized recommendations and increase sales [16]. In the travel industry, many other parameters can be taken into consideration when making recommendations to users. For example, the user's age may affect the travel solution they choose. Younger users may prioritize cheaper solutions based on their economic situation. Additionally, depending on their time, they may not have problems choosing trips with more connections. For example, young adults (aged 25-40) have the highest likelihood of taking a flight at 35%, followed by individuals aged 40-55 at 27%, those aged 56-74 at 21%, and individuals in Generation Z (aged 8- 24) at 16% [8]. In addition, users who are older (for example, 50 and over) may pay more attention to comfort, reliability, and easy access than money due to their health problems, economic situation, and limited time. While young individuals care more about money than comfort, it can be said that this is reversed at later ages. According to statistics, users between the ages of 18 and 25 spend approximately \$1,900 per traveler on trips; This is the minimum amount spent in the adult range [8]. One of the factors that users are influenced by when choosing travel solutions is environmental friendliness. Virtuoso said that 75% of its users are willing to pay more for sustainable travel modes [12]. These users can be expected to choose the train instead of the plane. If solutions arise, they can also be expected to prefer bicycles instead of to public transportation in the city, if solutions arise. Travel purposes may also affect users' preferences. Users traveling for business purposes can be expected to value speed. For example, Chatterjee mentioned that business travelers pay less attention to money. In addition, users' expectations of quality and price may vary depending on

the purpose of the trip [4]. Another group may be users who seek luxury and want to travel in luxury. Users looking for luxury do not want to be uniform. James Brown, CEO of Brownstone Hotels & Resorts, said that these users do not want standardization and do not prefer experiences squeezed into a formulaic place [13]. For this reason, they can be expected to prefer business flying or traveling by car rather than using public transportation. Users traveling as a family may have different expectations. They may want to balance money and convenience. It was observed that 42% of these users attach importance to price comparison. Additionally, it was stated that 41% of groups traveling as families attach great importance to cleanliness and hygiene [1]. Not every user may plan their trip in advance. Sandal mentioned that business travelers, immigrants who need to travel urgently due to family situations, or those who need to travel for health reasons may be included in this group [21]. Since these users make last-minute plans, their priority may be speed rather than money

1.2. Background

The architecture of recommender systems and their applications to real-world problems is an active area of research. Recommender systems use data source analysis methods to develop proximity concepts that can be used to identify well-matched pairs between users and items. The term collaborative filtering was coined by Goldberg et al. in 1992 as “information filtering when people are involved in the filtering process.” It has been put forward with the idea that filtering may be more effective. In this method, similar items are recommended to users with similar preferences. Compared to content-based filtering, there is no need for error-prone element processing in the collaborative filtering method. Since the ratings are made by users, it takes real-quality evaluations into account. The collaborative filtering method is expected to provide random recommendations because the recommendations are based on user similarity, not item similarity. It requires user participation, but motivation to participate is often low. This problem is called the “cold start” (new users, new items, new communities or disciplines) problem. If a new user rates a few items or no items, the system cannot provide recommendations. If an item is new to the system and has not yet been rated by at least one user, it cannot be recommended. To overcome the cold start problem, implicit ratings can be inferred from interactions between users and items [3]. The content-based filtering method is based on the principle of analysing the user profile, in other words, considering the user’s past choices and recommending choices like these [3]. It is the process of user modelling in which users’ interests are inferred from the elements they interact with. Elements are generally textual. Engagement is often created through actions such as downloading content,

tagging it, or typing in content of interest. Elements are represented by the model containing attributes. Attributes are generally word-based. Some recommendation systems also include non-textual attributes such as writing style, layout information, or XML tags. The user model consists of attributes of user elements. To generate recommendations, the user model and recommendation candidates are compared using similarity methods [25]. Content-based methods do not require information about other users. Additionally, problems such as data sparsity are avoided, and confidentiality is ensured. This method is very sensitive to the completeness of the item and the user definition [3]. To increase the accuracy of recommendation results, two or more recommendation techniques are combined in some scientific studies [18]. Content-based and collaborative filtering methods have advantages but also disadvantages; Using both together can complement each other. Recommendation systems using mixed methods can generally provide more accurate and clear results than recommendation systems containing single algorithms. To combine the advantages of these approaches and eliminate their disadvantages, hybrid methods have been proposed using weighted, cascaded, or mixed combination techniques. The advantage of this method is that different recommendation methods and information from many sources can be used together [15].

While this study was being carried out, the hybrid recommendation system THOR used by Javadian was used [10]. As Javadian mentioned, it is assumed that users have access to an application for travel offers [10]. In this study, to offer passengers a personalized experience, they applied a recommendation core based on the contextual preference model. This study answers two questions. The first of these explores the question of how to design a ranked preference model using users' past preferences. The second question is whether a preference model can be created by clustering users according to their similarities without any initial information. Travel offers and user profiles are collected when users send mobility requests, where each request includes the so-called "search options"—i.e., situational preferences such as the desired means of transportation and the maximum number of connections, as shown in Figure 1.1.

Name	Type	Category	Name	Type	Category
Age	Cat.	Profile	Starting Point	Cat.	Offer
City	Cat.	Profile	Destination	Cat.	Offer
Country	Cat.	Profile	Via	List	Offer
Loyalty Cards	List	Profile	LegMode	List	Offer
Payment Cards	List	Profile	Class	List	Offer
PRM Type	List	Profile	Seats Type	List	Offer
Profile Type	Cat.	Profile	Arrival Time	Cat.	Offer
Quick	Float	Offer	Departure Time	Cat.	Offer
Reliable	Float	Offer	Preferred Transp. Types	List	Search
Cheap	Float	Offer	Preferred Carriers	List	Search
Comfortable	Float	Offer	Preferred Refund Type	Cat.	Search
Door-to-door	Float	Offer	Preferred Services	List	Search
Envir. Friendly	Float	Offer	Max. No. of Transfers	Int.	Search
Short	Float	Offer	Max. Transfers Duration	Cat.	Search
Multitasking	Float	Offer	Max. Walking Dist. to Stop	Cat.	Search
Social	Float	Offer	Walking Speed	Cat.	Search
Panoramic	Float	Offer	Cycling Distance to Stop	Cat.	Search
Healthy	Float	Offer	Cycling Speed	Cat.	Search
Legs Number	Int.	Offer	Driving Speed	Cat.	Search

Figure 1.1: Main features used for data generation along with their type and category. [10]

In this study, unlike a user-specified list such as filtering and sorting, THOR also takes into account the user’s past preferences. Thor aims to rank travel offers based on users’ preferences. User profiles and travel offers are collected upon request. An offer list is created in response to requests. Points are calculated for each offer according to the parameters. Offers ranked according to these scores are presented to the user. This new information, labeled as received or not received, is added to the user’s historical data. The amount of data belonging to a user may be less than a certain threshold value. When this happens, these users are called cold users. When this happens, users with similar profiles are clustered. New users can also be assigned to a cluster in this way. Some steps were taken to prepare the data. The first of these is hot encoding. It is applied to categorical data. Columns that do not contain information are deleted. For example, the information about which city the user is from can be deleted because it is not meaningful information when all columns contain the same information. The data is then normalized so that it is on the same scale. Solutions recommended for cold users are used in cluster training for the new user. K-means and density-based spatial clustering of applications with noise (DBSCAN) algorithms were used for this. In these algorithms, the elbow method and silhouette coefficients were used to calculate the optimum number of clusters. In order to obtain more precise clustering results with DBSCAN, we calculate the

silhouette coefficient. This coefficient takes its highest value when the distances between points in the same cluster are as small as possible and the distances between different clusters are as large as possible. In the comparison between K-means and DBSCAN, K-means is a prototype-based clustering algorithm and is more suitable for sparse and high-dimensional data. K-means divides data points into a certain number of clusters or categories. A center is determined for each cluster, and each data point is assigned to clusters based on the nearest center. On the other hand, DBSCAN is a clustering algorithm that divides data points into clusters based on density and distance criteria. Kernel points are classified as edge points and noise points and are used to identify dense regions and outliers. It is more resistant to noise but does not perform well on high-dimensional data. The best parameter is calculated for each algorithm, and thus the cluster set of users with similar profiles is calculated. Historical data on users in each cluster is combined, and these data are used in the training module. When a new user registers with the system, he is assigned to a cluster thanks to his profile information, and the recommendation model is made according to this cluster. The CLASSIFIER_TRAIN function retrieves a user's records. By deciding on the most appropriate parameters, a recommendation model is created with these parameters. First, data preprocessing steps are performed for the data set. The best algorithm and its best parameters are found and saved. Each of its offers is classified as having been purchased or not. Predictions are made by a classification algorithm that predicts which offers the user will buy. First, each user is associated with his / her own data with a unique contextual preference model. For this purpose, K-Nearest Neighbor (KNN), Support Vector Classifier (SVC), Decision Tree (DT), Random Forest (RF) classification algorithms were used. It then determines the purchase probability for each recommendation presented to the user and ranks the offers accordingly. KNN does not focus on building a global internal model; instead, it stores all examples corresponding to the training data in n-dimensional space. KNN mines data and classifies new data points using similarity measures (such as the Euclidean distance function). SVC attempts to provide the widest separation to classify data points by creating a hyperplane. SVC considers features to find the optimal hyperplane for a given classification problem and classifies data points using this hyperplane. DT is an algorithm used to perform classification or regression using a tree structure created by ranking the features in a data set according to a set of decision rules. RF is a classification and regression algorithm that is an ensemble learning method created by combining multiple decision trees. Each tree is trained with a random sample and a random subset and combines their results to make stronger and more stable predictions. All these algorithms were used in the CLASSIFIER_TRAIN function and were used to predict whether the user would receive the relevant travel offer on the data set. How THOR predicts the offers

users receive with this function is shown in Algorithm 1.1. The user's current records were first preprocessed. Then, the parameters for the search option were determined, and the relevant algorithms were applied. The algorithm with the best score was recorded.

Algorithm 1.1 Recommender Model Training

Input: user records, i.e., the most recent profile information (profile), purchased and not-purchased enriched travel offers (travel offers), as well as their corresponding search options (requests)

Output: recommender model.

```

1 Function CLASSIFIER_TRAIN(userrecords):
2   | train data ← data preprocessing(user records)
3   | search ranges ← parameters range define
4   | for algo in [KNN, SVC, DT, LR, RF] do
5   |   | temp best model, score ← BSCV(algo, train data, search ranges)
6   |   | best model ← compare scores of the models
7   | end
8   | recommender model ← save the model in file(best model)
9 end function

```

2 | Dataset Generation Method

In this section, the steps followed when creating the data set are explained. In the first part of the section, the steps followed when creating a user profile, how scenarios are created, and how parameters are assigned when creating users are explained. In the second part of the section, it is explained how the starting and destination locations are selected, how travel offers are created, and how the 'Bought Tag' column value is assigned.

2.1. User profile Generation

By briefly reviewing the literature, the criteria that users pay attention to when choosing travel solutions were first investigated. Different user profiles were created according to the criteria resulting from this research. Different criteria were mentioned in the reviewed articles. User profiles were created considering these criteria. In the continuation of the study, it was decided for each profile whether they would receive travel offers or not. The general idea of assigning these features should be explained. The scenarios were created taking into account the literature review and generated scenarios are listed in Table 2.1, following scenarios have been created. The parameters that users care about may vary depending on the scenario. Assignments were made at certain intervals to determine which parameters users were likely to purchase. This shows how much importance users attach to each parameter. A high range of numbers has been assigned to the parameter that the user is expected to attach importance to, depending on the scenario it belongs to. Since other parameters may vary depending on each user and THOR is expected to learn the preferences of different users, the selection range of the possibilities of these parameters has been kept wide. Furthermore, thanks to the advantage of changing numerical values in this way, changes can be made to the scenarios easily.

Scenarios
Eco-Friendly Traveler
Budget Traveler
Business Traveler
Luxury Traveler
Tech-Savvy Traveler
Family Traveler
Last Minute Traveler

Table 2.1: Scenarios

Users belonging to the Eco-friendly scenario are expected to make choices that care about the environment and health. Therefore, it was expected that they would be more likely to choose environment-friendly and healthy travel offers. The probability of these offers being selected. It was selected in the value range of 0.6 and 0.8, while other bids were randomly selected in the value range of 0.2 and 0.8. In addition, the transportation modes preferred by these users were determined to be transit. For example, one is expected to choose the train among the offers that offer a train solution and the other a 'flight' solution. The Budget Traveler scenario is for users who prioritize cheapness. These users can be expected to care more about cheapness than comfort or speed like young adults mentioned in [8]. Therefore, they are more likely to choose a 'cheap' offer. Other parameters are written to be selected within a wide range. The transportation modes preferred by these users are assigned as 'transit'. In the business travel scenario, users are expected to be more likely to choose 'Short', 'Door-to-door', 'Quick' 'Reliable' offers. These users have been assigned to prefer 'flight' and 'driving' modes. Users belonging to the luxury travel scenario were more likely to choose 'Quick', 'Comfortable', 'Reliable', 'Door-to-door', 'Healthy' offers, considering their interest in comfort and luxury. The transportation modes they are expected to choose are assigned as 'flight' and 'driving'. The Tech-Savvy Traveler Preferring In-Transit Connectivity scenario was created for technology-savvy users. In this scenario, users are people who want to try modes of transportation such as train, car, or 'flight', regardless of speed or cheapness. They want reliable, multitasking offers. Another scenario is family traveler balancing quickness and price. Parameters were created considering that these users, traveling as a family, make their travel choices by establishing a balance in terms of speed, comfort, and price. It was created as a scenario that could choose all modes of transportation. The parameters are chosen with a balance between quickness and cheapness. However, it is still expected to have a higher probability of comfort because it is mentioned on [1] that 41% of family travelers care about comfort.

The last scenario is where there are users who plan a trip at the last minute. In this scenario, there could be users who has a urgent need to make a trip to their countries like immigrants [21]. These users also attach more importance to speed and reliability than to cheapness. For this reason, their preferences are towards planes or cars. The parameters that the users care about according to the scenarios they belong to are randomly selected between the values of 0.6 and 0.8. Other parameters were chosen randomly between 0.2 and 0.8, as they may vary depending on the users. Table 2.2 shows how the parameters are selected according to the scenarios. These parameters show the probability that the user will select the offer with the relevant parameter. They will be used while assigning the 'Bought Tag' parameter, which is explained in Section 2.2.

	Eco-Friendly	Budget	Business	Luxury	Tech-Savvy	Family	Last-Minute
Preferred Transportation	Transit	Transit	Flight,Car	Flight,Car	Flight, Train, Car	Flight, Transit, Car	Flight,Car
Quick	(0.2,0.8)	(0.2,0.8)	(0.6,0.8)	(0.6,0.8)	(0.2,0.8)	(0.3,0.7)	(0.6,0.8)
Reliable	(0.2,0.8)	(0.2,0.8)	(0.6,0.8)	(0.6,0.8)	(0.6,0.8)	(0.6,0.8)	(0.6,0.8)
Cheap	(0.2,0.8)	(0.6,0.8)	(0.2,0.8)	(0.2,0.8)	(0.2,0.8)	(0.3,0.7)	(0.2,0.8)
Comfortable	(0.2,0.8)	(0.2,0.8)	(0.6,0.8)	(0.6,0.8)	(0.2,0.8)	(0.7,0.9)	(0.2,0.8)
Door-to-door	(0.2,0.8)	(0.2,0.8)	(0.6,0.8)	(0.6,0.8)	(0.2,0.8)	(0.2,0.8)	(0.2,0.8)
Environmentally friendly	(0.6,0.8)	(0.2,0.8)	(0.2,0.8)	(0.2,0.8)	(0.2,0.8)	(0.2,0.8)	(0.2,0.8)
Short	(0.2,0.8)	(0.2,0.8)	(0.6,0.8)	(0.2,0.8)	(0.2,0.8)	(0.2,0.8)	(0.2,0.8)
Multitasking	(0.2,0.8)	(0.2,0.8)	(0.2,0.8)	(0.6,0.8)	(0.6,0.8)	(0.6,0.8)	(0.2,0.8)
Social	(0.2,0.8)	(0.2,0.8)	(0.2,0.8)	(0.2,0.8)	(0.2,0.8)	(0.2,0.8)	(0.2,0.8)
Panoramic	(0.2,0.8)	(0.2,0.8)	(0.2,0.8)	(0.2,0.8)	(0.2,0.8)	(0.2,0.8)	(0.2,0.8)
Healthy	(0.6,0.8)	(0.2,0.8)	(0.2,0.8)	(0.6,0.8)	(0.2,0.8)	(0.2,0.8)	(0.2,0.8)

Table 2.2: Probabilities for each parameter according to scenarios

Other parameters belonging to the user are randomly selected from the data of each parameter and assigned to the user. These parameters are categorical variables belonging to user profiles. 'Loyalty', which indicates which program the customers belong to, 'Payment', which shows their preferred payment method; 'Class', which shows which class they want to travel in; 'Seat', which shows their seat preferences; 'Refund', which shows their preferences for the refund method; 'Refund', which shows the customer profile. 'Profile', 'Services' showing the preferred services, 'Transfer' and 'Transfer Duration' showing the preferred maximum number of transfers and preferred minimum transfer time, and 'Walking Distance' and 'Walking Speed' parameters showing the preferred maximum walking distance and walking speeds. It was randomly selected from the data shown in Table 2.3 and assigned to each user.

Parameter	Data
Loyalty	FlyingBlue, Golden Card, Cartafreccia
Payment	Mastercard, Paypal, Visa
Class	First Class, Economy, Business
Seat	Large, Window, Normal
Refund	Automatic refund
Profile	Business, Family, Leisure, Basic
Services	Air, Water transport, Highspeed Train, Miscellaneous
Transfer	Max 1, Max 2, Max 3, Unlimited
Transfer Duration	At least 35 min, At least 40 min, At least 45 min, At least 30 min, At least 25 min, At least 20 min, At least 15 min, At least 10 min
Walking Distance	500m, 1500m, 1000m, 4000m, 2000m, 3500m, 3000m, 4500m
Walking Speed	Slow, Medium, Fast

Table 2.3: Parameter and Data Table

2.2. Travel Offer Generation

The Travel Destinations data set was used when selecting the route for users. This dataset is taken from the travel-bot-project github repository. This data set includes 62 locations from different parts of Europe. For each offer, two positions were randomly selected, one of which was starting and the other was destination. If the starting and destination locations are the same, two random locations are selected again since a route cannot be created between them.

After these two locations were selected, various routes were created between them. Google Maps API was used for the realism of these routes. When Google Maps API generates a response, in addition to location information, the parameter of which mode will be used to create the route should also be given. There upon, the API generates a response containing information such as Leg Mode, leg carrier, distance, duration, vehicle type for each route, after the start and end information of the route. However, for the realism of the data set and for our model to understand user preferences, it is necessary to obtain results containing different travel models. For this reason, travel mode transit was selected first when creating API responses. While this mode includes transportation vehicles such as trains and buses, it also provides vehicle solutions such as metro and tram for urban public transportation. In order to limit the obtained API answers, a maximum of 4 of these answers were randomly selected. The distance between starting and destination point is extracted from the API response. If this distance was 300 km or more, 'flight' mode was added to one of these 4 answers randomly. Additionally, 'driving' was randomly added to one of the 4 answers. Thus, 4 offers were created with an API query. For example,

when the API response is generated for Duomo, Milan and Turin, there will be no flight mode because the distance between them is less than 300 km, but 1 proposal will include a driving mode, while the other 3 proposals will be created with transit modes. If an API response is generated between Duomo di Milano and Taksim Square Istanbul, this time there will be an offer including flight mode in addition to driving and transit mode. For each offer, Quick, Reliable, Cheap, Comfortable, Door-to-door, environmentally friendly, Short, Multitasking, Social, Panoramic, Healthy values are created as numerical values. These values were chosen randomly, between 0.2 and 0.8. Thus, each offer is kept with leg mode, leg carrier, vehicle type and numerical values. For the last stage of the data set, the 'Bought Tag' parameter should be assigned to inform whether users' offers will be accepted or not. This value is assigned by comparing the values the user cares about in the offer and the values the offer has, and adding to this comparison the similarity of the transportation modes the user prefers and the transportation modes suggested by the offer. We take all possibilities into account by multiplying the probability of the user selecting the offer with parameter x (p_x) by the probability of the offer having parameter x (q_x). This multiplication also includes the similarity ratio of the transportation modes preferred by the user and the transportation modes of the offer as shown in Equation 2.1

$$\prod_x p_x * q_x \quad (2.1)$$

For example, what percentage of the transportation modes in the offer fall into the user's preferred modes. This was taken as the transportation mode similarity value. The 'quick' value preferred by the user is multiplied by the 'quick' value of the offer. By doing this, for each numerical value, these probabilities are multiplied. The sigmoid function was used to scale the values between 0 and 1.

$$h_{\theta}(x) = \frac{1}{1 + e^{-\theta^T x}}$$

With this final probability, a random selection was made between 0 and 1 values. If the 'Bought Tag' value is 1, this means that the offer has been purchased; if it is 0, this means that the offer has not been purchased.

For example, let us take the user from the Business Traveler scenario who is looking for a route from Duomo Milano to Istanbul Taksim Square. Different purchasing probabilities were calculated for 4 offers from the API response. For example, the probability of receiving an offer that includes 'flight' mode is 89%, while the probability of purchasing another offer that uses transit modes is 52%. Numerical values of the offer containing

Flight mode and the user are shown in Table 2.4 The similarity of the transportation modes preferred by the user and the transportation modes in the offer. It is 15%. When these values are multiplied and put into the sigmoid function, 89% is obtained. It means that the user will buy the offer with probability 89%. In other words, the 'Bought Tag' will be chosen '1' with probability 89%

Parameter	Offer Values	User Values
Quick	0.4746	0.4506
Reliable	0.6996	0.6069
Cheap	0.7088	0.2887
Comfortable	0.6274	0.6294
Door-to-door	0.3143	0.7088
Environmentally friendly	0.6246	0.4675
Short	0.9824	0.738
Multitasking	0.49	0.2478
Social	0.3032	0.2114
Panoramic	0.9714	0.3199
Healthy	0.6273	0.6013

Table 2.4: Numerical Values for Offer and User

These steps are visualized and shown as a flowchart in Figure 2.1. As it is seen, after choosing the scenario, the starting and destination point is also chosen. According to that, travel offers are generated by an API. If the distance between points is higher than 300 km, a 'flight' mode is added as one of the offers. After that, regardless of the distance, between a 'driving' mode is added. Since the offers are ready for users, the probability of purchase is calculated, and according to the that probability, 'Bought Tag' is assigned by randomly choosing with calculated probability.

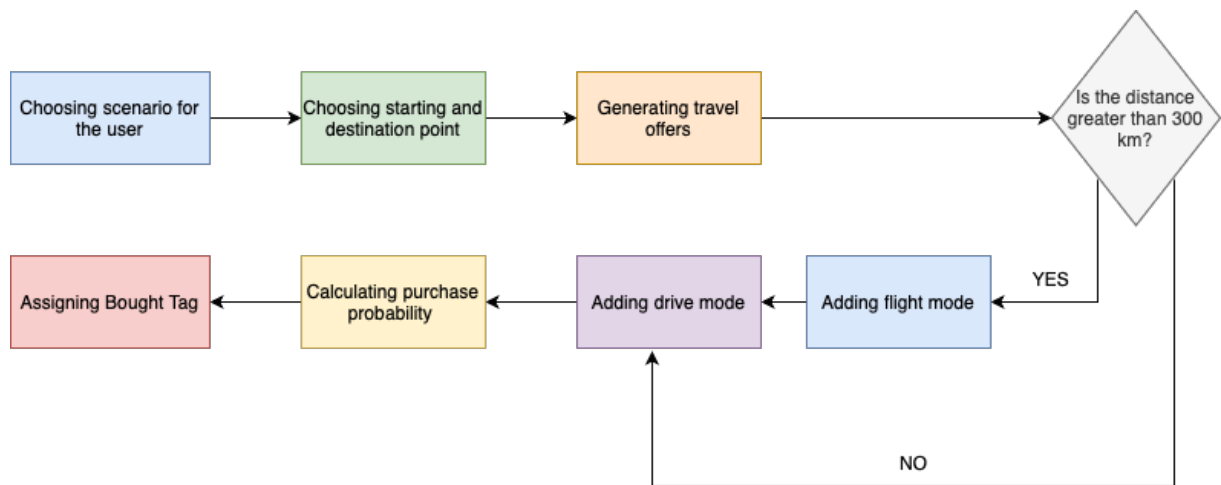


Figure 2.1: Workflow of the generation dataset

3 | Experimental Results

In this section, the results obtained after creating the data set are interpreted. Two experiments were carried out in this section. Thus, the performance of THOR was investigated on different data sets. First, the created data set was interpreted, and then the data set was revised to improve the results. In the first part of the section, the results of the first data set are interpreted. In the second part, a second data set was created by changing the numerical parameters. It is also explained how two second data sets were created, and the results of this new data set are interpreted. In the last section, the results are generally evaluated.

3.1. First Batch of Experiments

A new user set was created to test the performance of the THOR system's classification algorithms. With the method explained in Section 3, 30 users and 25 API queries for each user were sent. The parameters used according to the scenario when creating each user are given in Table 2.2. Thus, 100 offers were created for each of the 30 users. While creating these 30 users, each scenario was randomly selected. Table 3.1 shows which scenario each user belongs to. In the first data set, there are five Family Travelers, six Luxury Travelers, three Eco-Friendly Travelers, five Budget Travelers, three Tech Savvy Travelers and four Last Minute Travelers.

There is good distribution in terms of providing a variety of users for each scenario. Thus, the results of different users in the same scenario can be compared. Thus, it can be seen whether THOR can learn the preferences of users who may have different preferences in some parameters of the same scenario.

User Profile	Scenario	User Profile	Scenario
Profile 1	Family	Profile 16	Tech-Savvy
Profile 2	Family	Profile 17	Business
Profile 3	Luxury	Profile 18	Eco-Friendly
Profile 4	Luxury	Profile 19	Luxury
Profile 5	Luxury	Profile 20	Luxury
Profile 6	Eco-Friendly	Profile 21	Budget
Profile 7	Eco-Friendly	Profile 22	Last Minute
Profile 8	Budget	Profile 23	Budget
Profile 9	Tech-Savvy	Profile 24	Budget
Profile 10	Family	Profile 25	Tech-Savvy
Profile 11	Last Minute	Profile 26	Luxury
Profile 12	Family	Profile 27	Tech-Savvy
Profile 13	Budget	Profile 28	Tech-Savvy
Profile 14	Luxury	Profile 29	Last Minute
Profile 15	Last Minute	Profile 30	Business

Table 3.1: User Profiles and Scenarios

80% of the 100 offers of information that users had were used for training, while the rest were used for testing. In this way, it was measured whether the THOR system could accurately predict whether the user would purchase the offer or not. In Figure 3.1, the accuracy rate for each algorithm is shown as a box plot. Although the highest median belongs to the decision tree algorithm, it has also been observed that the decision tree algorithm gives an accuracy rate of 0 for some users. In addition, although the median value of the logistic regression algorithm is 48%, it can be seen that it has high values for some users. However, the accuracy values obtained in general are not promising. Again, it was investigated whether the model worked better for some scenarios. However, it has been seen that there is no pattern, depending on the scenario. For example, Profile 14 is a user profile belonging to the Luxury scenario, and the decision tree has an accuracy score of 87% for this profile, while Profile 19 for the same scenario has an accuracy score of 15% for this Luxury Traveler profile when predicting offers. Therefore, it was investigated whether users with very high and very low accuracy rates had a special pattern. However, no significant pattern is seen for the users who give these accuracy rates. For example, the values of Profile 16, which gives an accuracy rate of 80%, were examined. No significant pattern was found. At the end, it was decided that the learning rate of the model did not have a significant relationship with the scenario or the pattern of users.

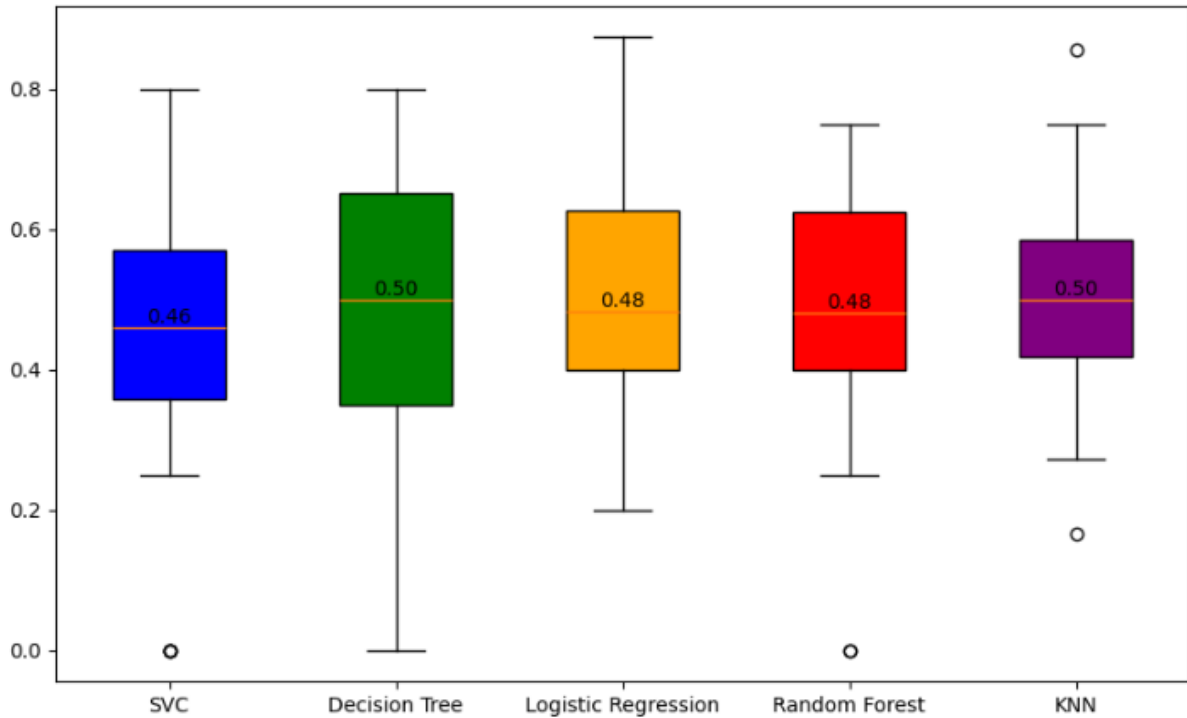


Figure 3.1: The accuracies of each algorithm for all the users

Additionally, in Figure 3.2, the rates of the algorithms giving the highest accuracy score for each user profile are shown as a pie chart. As can be seen from this graph, the graph that gives the best score for each user profile may be different, so we cannot argue that one algorithm will be better than the other. Again, it was investigated whether some models worked better for certain scenarios. For example, it was seen that KNN gave the best result for one of the users in the Budget Traveler scenario, SVC gave the best result for another, and the DT algorithm gave the best result for another. It was concluded that one model is not better than the other in the scenario.

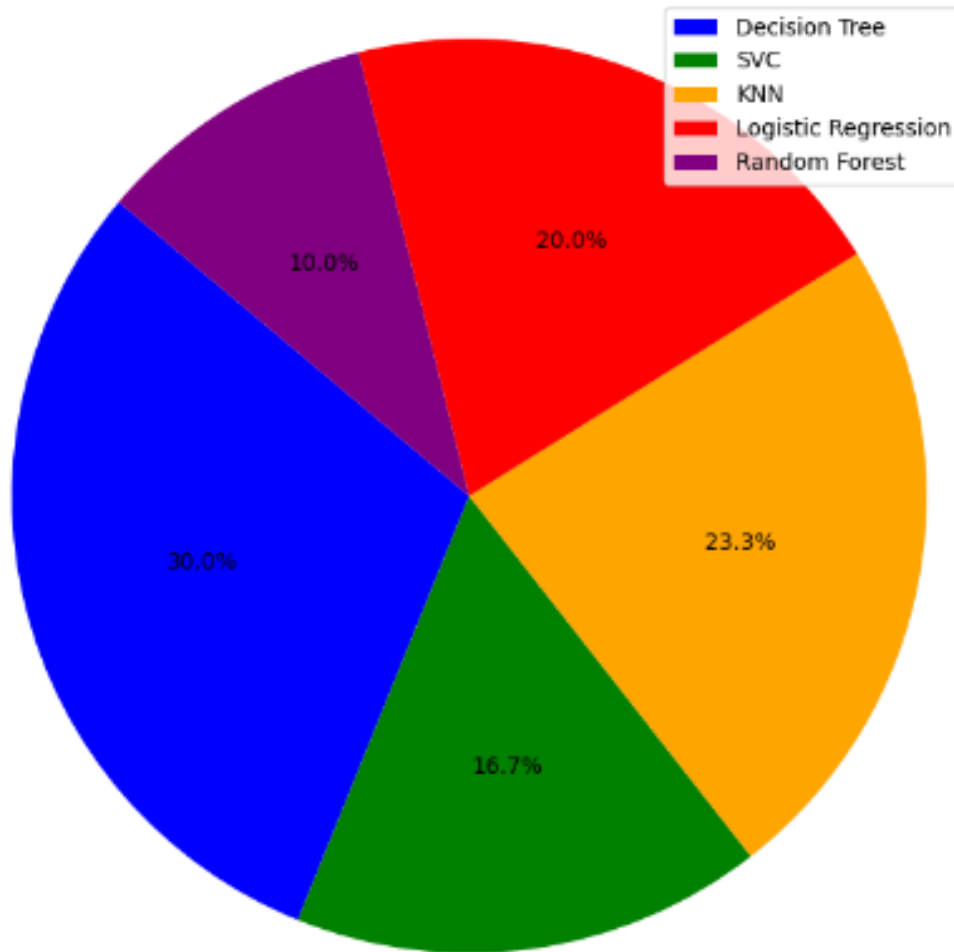


Figure 3.2: The ratio of different algorithms is used as the best mode

3.2. Second Batch of Experiments

Since the results were not at the expected level, a new trial was made by changing the user parameters. Changes have been made to the users' Quick, Reliable, Cheap, Comfortable, Door-to-door, Environmentally Friendly, Short, Multitasking, Social, Panoramic and Healthy values. Values other than those specifically set to be high were randomly selected between 0.2 and 0.4, and were set to be particularly low. Thus, the parameters that the user cares about are made clear. This was done to make the data set skewed. So that it can be seen that what users will particularly choose. The ranges in which the parameters were created while creating the new data set were selected in Table 3.2. 30 users were created again with these parameters.

	Eco-Friendly	Budget	Business	Luxury	Tech-Savvy	Family	Last-Minute
Preferred Transportation	Transit	Transit	Flight,Car	Flight,Car	Flight, Train, Car	Flight, Transit, Car	Flight,Car
Quick	(0.2,0.4)	(0.2,0.4)	(0.6,0.8)	(0.6,0.8)	(0.2,0.4)	(0.3,0.7)	(0.6,0.8)
Reliable	(0.2,0.4)	(0.2,0.4)	(0.6,0.8)	(0.6,0.8)	(0.6,0.8)	(0.6,0.8)	(0.6,0.8)
Cheap	(0.2,0.4)	(0.6,0.8)	(0.2,0.4)	(0.2,0.4)	(0.2,0.4)	(0.3,0.7)	(0.2,0.4)
Comfortable	(0.2,0.4)	(0.2,0.4)	(0.6,0.8)	(0.6,0.8)	(0.2,0.4)	(0.7,0.9)	(0.2,0.4)
Door-to-door	(0.2,0.4)	(0.2,0.4)	(0.6,0.8)	(0.6,0.8)	(0.2,0.4)	(0.2,0.4)	(0.2,0.4)
Environmentally friendly	(0.6,0.8)	(0.2,0.4)	(0.2,0.4)	(0.2,0.4)	(0.2,0.4)	(0.2,0.4)	(0.2,0.4)
Short	(0.2,0.4)	(0.2,0.4)	(0.6,0.8)	(0.2,0.4)	(0.2,0.4)	(0.2,0.4)	(0.2,0.4)
Multitasking	(0.2,0.4)	(0.2,0.4)	(0.2,0.4)	(0.6,0.8)	(0.6,0.8)	(0.6,0.8)	(0.2,0.4)
Social	(0.2,0.4)	(0.2,0.4)	(0.2,0.4)	(0.2,0.4)	(0.2,0.4)	(0.2,0.4)	(0.2,0.4)
Panoramic	(0.2,0.4)	(0.2,0.4)	(0.2,0.4)	(0.2,0.4)	(0.2,0.4)	(0.2,0.4)	(0.2,0.4)
Healthy	(0.6,0.8)	(0.2,0.4)	(0.2,0.4)	(0.6,0.8)	(0.2,0.4)	(0.2,0.4)	(0.2,0.4)

Table 3.2: Probabilities for each parameter according to scenarios

The scenarios selected for the new users created are shown in Table 3.3. Again, these randomly selected scenarios are important for the model to see how the same scenario yields results for different users. In the second data set, there are five Family Travelers, two Luxury Travelers, six Eco-Friendly Travelers, one Budget Traveler, eight Tech Savvy Travelers and five Last Minute Travelers. Although the highest median belongs to the Logistic Regression algorithm, it has also been observed that the decision tree algorithm gives an accuracy rate of 1.0 for some users. However, the accuracy values obtained in general are not promising. Again, it was investigated whether the model works better for some scenario-specific scenarios. However, there is no pattern, depending on the scenario. For example, Profile 4 is a user profile belonging to the Tech Savvy scenario, and the decision tree has an accuracy score of 83% for this profile, while Profile 17 for the same scenario has an accuracy score of 33%. So, it was decided that the learning rate of the model did not have a significant relationship with the scenario. Therefore, it was investigated whether users with very high and very low accuracy rates had a special pattern. However, no significant pattern is seen for the users who give these accuracy rates. For example, the values of Profile 6, which gives an accuracy rate of 100%, were examined. No significant pattern was found. At the end, it was decided that the learning rate of the model did not have a significant relationship with the scenario or the pattern of users.

User Profile	Scenario	User Profile	Scenario
Profile 1	Eco-Friendly	Profile 16	Budget
Profile 2	Tech-Savy	Profile 17	Tech-Savy
Profile 3	Family	Profile 18	Eco-Friendly
Profile 4	Tech-Savy	Profile 19	Business
Profile 5	Eco-Friendly	Profile 20	Luxury
Profile 6	Tech-Savy	Profile 21	Tech-Savy
Profile 7	Business	Profile 22	Luxury
Profile 8	Last Minute	Profile 23	Eco-Friendly
Profile 9	Tech-Savy	Profile 24	Family
Profile 10	Eco-Friendly	Profile 25	Tech-Savy
Profile 11	Last Minute	Profile 26	Last Minute
Profile 12	Family	Profile 27	Business
Profile 13	Tech-Savy	Profile 28	Family
Profile 14	Last Minute	Profile 29	Last Minute
Profile 15	Family	Profile 30	Eco-Friendly

Table 3.3: User Profiles and Scenarios

First of all, the general results were looked at for accuracy scores. These results did not give promising results, as seen in Figure 3.3 as a box plot. There was a slight improvement in logistic regression, SVC, and random forest algorithms, but these results are still not at the desired level. There were users for whom the SVC algorithm gave a 0% accuracy rate.

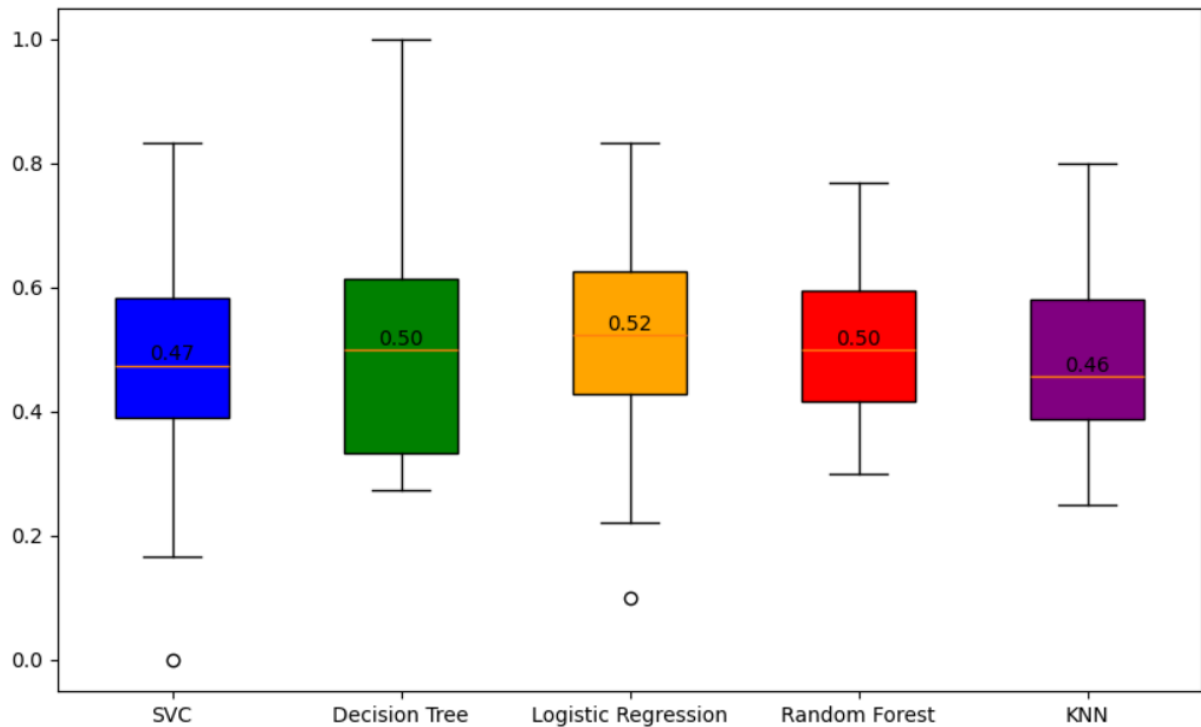


Figure 3.3: The accuracies of each algorithm for all the users

For this data set, it was investigated whether the model worked better in some scenarios. However, again, no significant pattern was found. For example, while Eco Friendly achieved an 83% accuracy rate for users, a 15% accuracy rate was also observed for other users in the same scenario. Additionally, it has been seen that different algorithms give the best scores for the same scenarios. The rates of the algorithms giving the highest accuracy score for this data set are shown in Figure 3.4 as a pie chart. As can be seen from this graph, there is no algorithm that specifically gives better results. Again, it was investigated whether some models worked better for certain scenarios. For example, it was seen that KNN gave the best result for one of the users in the Tech Savvy Traveler scenario, LR gave the best result for another, and the SVC algorithm gave the best result for another. It was concluded that one model is not better than the other, specifically for the scenario.

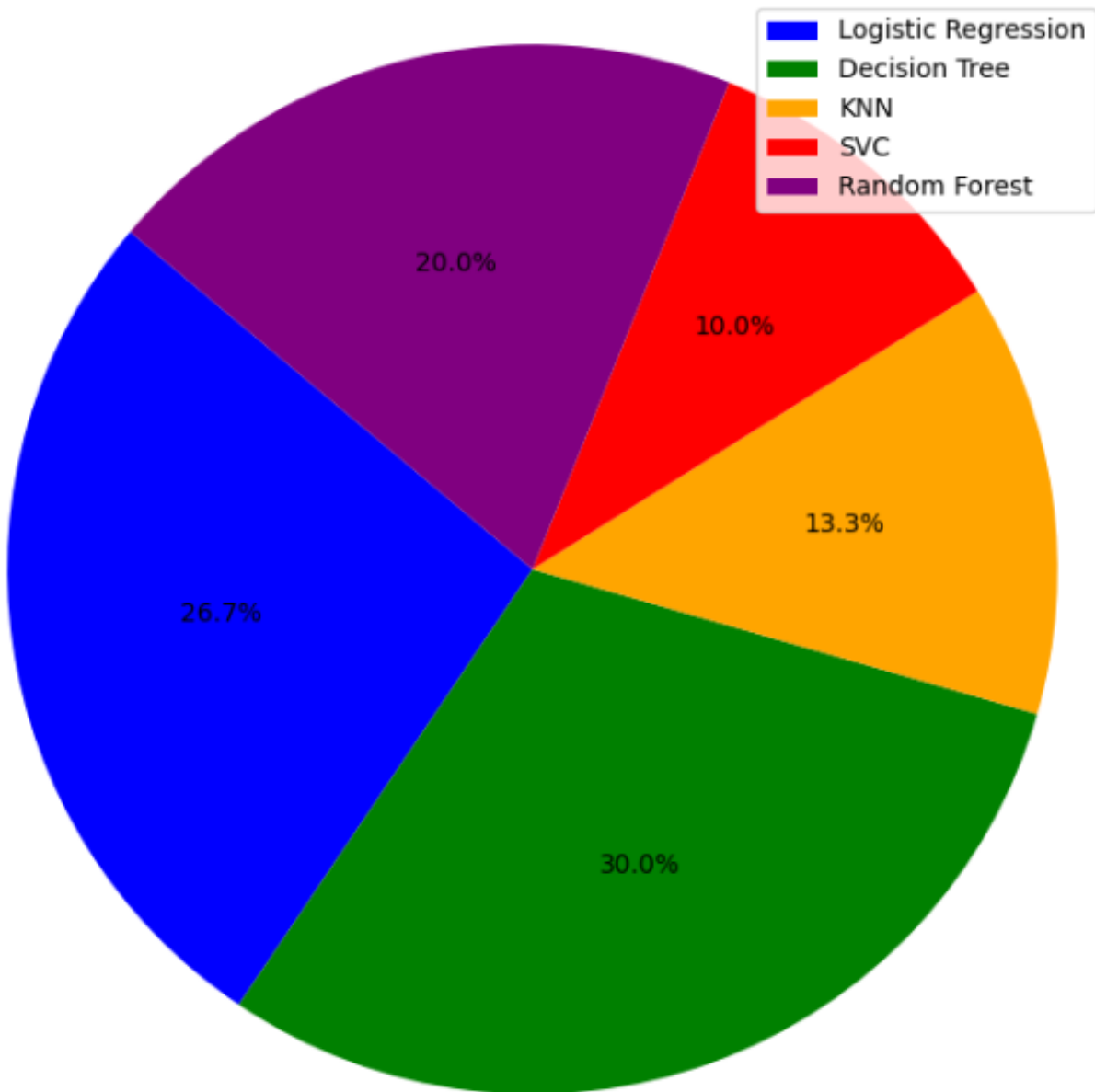


Figure 3.4: The ratio of different algorithms is used as the best mode

3.3. Discussion

Considering the general results of the study, promising accuracy rates were not achieved. Therefore, the method of creating the data set should be questioned. It is thought that the method of assigning the "Bought Tag" value and the way the numerical values of the offers are assigned increase the complexity and make the model difficult to learn. Although some predictions with high accuracy values were made, no pattern was found between them. This caused questions about the method of creating the data set.

Although most of the values are limited by giving ranges, the selection of numerical parameters for the offers is still done randomly. For example, the 'Environmentally Friendly' parameter of an offer that includes 'flight' mode may be assigned high because it is randomly assigned. This may be one of the reasons why no significant pattern was found when looking at the results. Following a pattern when assigning these numerical values to offers can improve the model's ability to learn user preferences and make the correct classification. In conclusion, it may be useful to create a new data set by making this method more meaningful.

In addition, since the THOR system is expected to learn the personal preferences of users, work can be done on expanding the classification algorithms to learn different situations. because users of the same scenario may have different preferences. Thor is expected to find out. Therefore, it may be useful to improve the model to understand more specific users.

4 | Conclusions and Future Work

This study was conducted to test the performance of the THOR system, which can predict users' preferred travel offers by learning their preferences. Since there is no real data set to test the THOR system, the primary aim was for the data set created in this study to be realistic. The second goal is to measure the performance of THOR's classification algorithms for this data set. Before creating the data set, a literature review was conducted, and certain scenarios were created based on this literature review. These scenarios represent certain profiles of traveling users. The scenarios created in this study are: Eco-Friendly Traveler, Budget Traveler, Business Traveler, Luxury Traveler, Tech-Savvy Traveler, Family Traveler and Last Minute Traveler. In each scenario Quick, Reliable, Cheap, Comfortable, Door-to-door, Environmentally friendly, Short, Multitasking, Social, Panoramic, Healthy values between 0.2 and 0.8 were assigned to the parameters to indicate the probability of being chosen by the user about the relevant parameter. A dataset containing Europe travel points was used to determine the users' travel routes. Travel solutions between these travel points have been created with the Google Maps API. Solutions for transit mode were obtained with the API, and 'driving' mode was added to these modes manually. 'Flight' mode has also been included according to the distance between travel points. Travel offers were obtained from an API query, resulting in 4 offers. A total of 30 user profiles were created, and the scenario of each user's profile was randomly selected. 25 API queries were created for each user, i.e., 100 offers in total. These created user datasets were tested in THOR's classification algorithms. 80% of the datasets were used for training and the rest for testing, and it was tried to predict whether the user would buy the offer or not. THOR uses DT, KNN, LR, RF, and SVC classification algorithms by learning the user's preferences. The result of the algorithms gave an accuracy value of around 50%, which is not a promising value. Therefore, the parameters were changed to make the data set skewed. However, this did not significantly increase the results of the algorithms. Therefore, it may be useful to retest the THOR system in the future by changing the method of creating the dataset and generating data again. There is no specific algorithm that gives the best results for a particular scenario, nor is there a specific algorithm that gives the best results for a particular scenario. In

general, no significant results were found within or between scenarios. While creating this data set, random values were given to each offer, and with these values, the variable of whether the user would receive the offer or not was assigned a multiplying of probabilities. Another method may be tried in future studies. For example, since this variable may depend on many variables, one might consider using Bayesian probability. An attempt can be made to recreate the data set with the approach produced by Gogoshin to create conditional distributions in his 2021 study [6]. Additionally, the approach suggested by Mendez-Vazquez in his study, [19] can be used to obtain an output as a result of the effectiveness of meaningful patterns. In this approach, the Markov chain was used to create the model, and the acceptance-rejection system was used together. It is thought that more realistic and meaningful results will emerge as a result of the harmonious production of data. Additionally, in future studies, it may be considered to expand THOR's classification algorithms to adapt them to different users. Thus, the accuracy level of classification algorithms can be increased by creating a model that is more sensitive to special situations or users.

Bibliography

- [1] 64;CondorFerries. 60+ family travel statistics trends 2023. URL <https://www.condorferries.co.uk/family-travel-statistics>.
- [2] Xavier Amatriain. Netflix recommendations: Beyond the 5 stars (part 2), 2012. URL <https://www.kdnuggets.com/2012/06/netflix-recommendations-beyond-5-stars.html>.
- [3] Joeran Beel, Bela Gipp, Stefan Langer, and Corinna Breitingner. Research paper recommender systems: A literature survey. *International Journal on Digital Libraries*, pages 1–34, 2015. ISSN 1432-5012. doi: 10.1007/s00799-015-0156-0.
- [4] Sheshadri Chatterjee, Nripendra P Rana, Sangeeta Khorana, Patrick Mikalef, and Anuj Sharma. Assessing organizational users’ intentions and behavior to AI integrated CRM systems: A meta-UTAUT approach. *Inf. Syst. Front.*, 25(4):1299–1313, August 2023.
- [5] James Davidson, Benjamin Liebald, Junning Liu, Palash Nandy, Taylor Van Vleet, Ullas Gargi, Sujoy Gupta, Yu He, Mike Lambert, Blake Livingston, and Dasarathi Sampath. The YouTube video recommendation system. In *Proceedings of the fourth ACM conference on Recommender systems*, New York, NY, USA, September 2010. ACM.
- [6] Grigoriy Gogoshin, Sergio Branciamore, and Andrei S. Rodin. Synthetic data generation with probabilistic bayesian networks. *Mathematical Biosciences and Engineering*, 18(6):8603–8621, 2021. ISSN 1551-0018. doi: 10.3934/mbe.2021426. URL <http://dx.doi.org/10.3934/mbe.2021426>.
- [7] Carlos A Gomez-Uribe and Neil Hunt. The netflix recommender system. *ACM Trans. Manag. Inf. Syst.*, 6(4):1–19, January 2016.
- [8] Andrew Graft. Travel and tourism statistics: The ultimate collection, Jun 2023. URL <https://blog.accessdevelopment.com/tourism-and-travel-statistics-the-ultimate-collection>.

- [9] Yajie Hu and Mitsunori Ogihara. Nexttone player: A music recommendation system based on user behavior. pages 103–108, 01 2011.
- [10] Alireza Javadian Sabet, Mahsa Shekari, Chaofeng Guan, Matteo Rossi, Fabio Schreiber, and Letizia Tanca. THOR: A hybrid recommender system for the personalized travel experience. *Big Data Cogn. Comput.*, 6(4):131, November 2022.
- [11] Ondrej Kaššák, Michal Kompan, and Mária Bieliková. Personalized hybrid recommendation for group of users: Top-N multimedia recommender. *Inf. Process. Manag.*, 52(3):459–477, May 2016.
- [12] Debbi Kickham. Virtuoso reveals top travel trends for 2023, Nov 2022. URL <https://www.forbes.com/sites/debbikickham/2022/11/25/virtuoso-reveals-top-travel-trends-for-2023/>.
- [13] Brandon K Kochkodin. Treat them like royalty: Customer experience lessons from luxury brands, Apr 2022. URL <https://www.forbes.com/sites/chipbell/2022/04/14/treat-them-like-royalty-customer-experience-lessons-from-luxury-brands/>.
- [14] Lukas Lerche. Using implicit feedback for recommender systems: characteristics, applications, and challenges, 2016.
- [15] Jie Lu, Dianshuang Wu, Mingsong Mao, Wei Wang, and Guangquan Zhang. Recommender system application developments: A survey, 04 2015.
- [16] Vincent Magnini, Earl Honeycutt, and Sharon Hodge. Data mining for hotel firms: Use and limitations, 04 2003.
- [17] Reem Fawzy Mahrous. Multimodal transportation systems : modelling challenges, 2012. URL <http://essay.utwente.nl/84855/>.
- [18] Bernadetta Maleszka. A framework for research publication recommendation system, 08 2019.
- [19] Andres Mendez-Vazquez, Sumi Helal, and Diane Joyce Cook. Simulating events to generate synthetic data for pervasive spaces. 2008. URL <https://api.semanticscholar.org/CorpusID:6488866>.
- [20] Francesco Ricci. Travel recommender systems. 2002. URL <https://api.semanticscholar.org/CorpusID:16238338>.
- [21] Candace Sandal. By the way . . . the last-minute international traveler. *Workplace*

- Health amp; Safety*, 67(2):53–55, November 2018. ISSN 2165-0969. doi: 10.1177/2165079918806535. URL <http://dx.doi.org/10.1177/2165079918806535>.
- [22] Iqbal H Sarker. Data science and analytics: An overview from data-driven smart computing, decision-making and applications perspective. *SN Comput. Sci.*, 2(5): 377, July 2021.
- [23] Chak Keung Simon Wong and Fung Ching Gladys Liu. A study of pre-trip use of travel guidebooks by leisure travelers. *Tour. Manag.*, 32(3):616–628, June 2011.
- [24] Loren Terveen and Will Hill. Beyond recommender systems: Helping people help each other. 02 2001.
- [25] Bo Wang, Feiyue Ye, and Jialu Xu. A personalized recommendation algorithm based on the user’s implicit feedback in e-commerce. *Future Internet*, 10(12):117, November 2018.
- [26] AYSAN ŞENTÜRK. *Veri Madenciliği : Kavram ve Teknikler*. Ekin Yayınevi, 2006.

A | Appendix A

If you need to include an appendix to support the research in your thesis, you can place it at the end of the manuscript. An appendix contains supplementary material (figures, tables, data, codes, mathematical proofs, surveys, . . .) which supplement the main results contained in the previous chapters.

Algorithm Learner

Input: username; reClusterTag; new profile (optional).

Output: user-specific recommender model or cluster-wide recommender model.

```

1: function MODELTRAIN(username, reClusterTag, new profile)
2:   user's historical records  $\leftarrow$  fetch user records from database(username)
3:   user type  $\leftarrow$  check the number of user's historical records
4:
5:   if user new profile is given then
6:     update the database with new profile
7:   end if
8:
9:   if user type is old user then
10:    user recommender model  $\leftarrow$  CLASSIFIER_TRAIN(user's historical records)
11:  else
12:    if cluster model does not exist OR reClusterTag is True then
13:      search ranges  $\leftarrow$  parameters range define
14:      cluster model, cluster-wide recommender models  $\leftarrow$  ClusterModelTrain(search
15: ranges)
16:    end if
17:    user cluster  $\leftarrow$  get user cluster(cluster model, username)
18:    cluster-wide recommender model  $\leftarrow$  get cluster-wide recommender model(cluster-wide
19: recommender models, user cluster)
20:    user recommender model  $\leftarrow$  copy model(cluster-wide recommender model)
21:  end if
22:  user-specific recommender model  $\leftarrow$  save model(user recommender model)
23: end function

```

Figure A.1: Pseudo-code of Learner Algorithm [10]

Algorithm Cluster Model Training

Input: search ranges for each algorithm.

Output: cluster model; cluster-wide recommender models.

```

1: function CLUSTERMODELTRAIN(search ranges)
2:   user profiles ← fetch all user profiles from the database
3:   best parameters ← compute best parameters for algorithms(search ranges)
4:   cluster model ← CLUSTER_TRAINING(best parameters, user profiles)
5:
6:   for cluster ∈ cluster model do
7:     cluster historical records ← merge all the users' records in the cluster
8:     cluster-wide recommender model ← CLASSIFIER_TRAIN(cluster historical records)
9:   end for
10: end function

```

Figure A.2: Pseudo-code of Cluster Model Training Algorithm [10]

Algorithm RANKER

Input: user's recommender model (model), most recent profile information (profile), search options (request), list of enriched travel offers (travel offers)

Output: ranked list of travel offers (ranked offers)

```

1: function RANKER(model, profile, request, travel offers)
2:   scored_travel_offers = []
3:
4:   for offer ∈ travel offers do
5:     (offer_id, prediction's score) ← CLASSIFIER_RESPONSE(model, profile, request, offer)
6:     scored_travel_offers.append((offer_id, prediction's score))
7:   end for
8:   ranked offers ← sort(scored_travel_offers)
9: end function

```

Figure A.3: Pseudo-code of Ranker Algorithm [10]