



# International Journal of Research Publication and Reviews

Journal homepage: [www.ijrpr.com](http://www.ijrpr.com) ISSN 2582-7421

## Smart Recommendations for Travel Destinations

**Rakesh Kothapalli**

Post Graduate Student, M.C.A., Department of IT, University College of Engineering, Science & Technology, JNTUH, Kukatpally, Hyderabad, Telangana, India – 500085

[kothapallirakesh3@gmail.com](mailto:kothapallirakesh3@gmail.com)

### ABSTRACT

Tourism has become one of the largest industries in the world, and selecting a suitable destination is often a complex task for travelers due to the overwhelming amount of information available online. Traditional Travel Recommendation Systems (TRSs) have been developed to address this challenge, but most of them either focus only on accuracy or function as black-box models, reducing transparency and user trust.

This project presents a Decision Tree-based Tourist Recommendation System implemented as a Flask web application. The system allows users to upload datasets, preprocess data using Pandas to handle missing values, and apply Label Encoding to convert categorical features into numerical form. A Decision Tree Classifier is then trained to generate recommendations based on user preferences. To ensure transparency, the system also provides a graphical visualization of the trained model using Graphviz.

The system delivers accurate recommendations in real time while maintaining ease of use through its web interface. The integration of interpretability and machine learning makes this approach practical for the tourism domain.

### INTRODUCTION

Choosing a tourist destination has become increasingly difficult for travelers because of the massive amount of information spread across the internet, travel platforms, and social media. With so many options and sources, people often experience information overload, making it challenging to identify places that truly match their interests, preferences, budget, or available time.

Although traditional Travel Recommendation Systems (TRSs) try to reduce this burden—typically using collaborative or content-based filtering techniques—they still face several drawbacks:

- **Low accuracy**, often caused by poor handling of varied and complex datasets.
- **Limited user-friendliness**, especially for people who are not familiar with technical systems.
- **Poor interpretability**, meaning users cannot clearly see or understand the reasoning behind the suggestions.

Therefore, there is a clear need for a recommendation system that is not only accurate but also simple to use and transparent in its decision-making process, enabling personalized and trustworthy travel recommendations.

### PROPOSED SYSTEM

The system is made to be easy to use and understand. It works in these steps:

1. **Upload Data** Users upload a CSV file with tourist information.
2. **Clean Data** The system removes missing values and changes text into numbers.
3. **Train Model** A Decision Tree is trained because it is simple and easy to understand.
4. **Get Recommendation** The user enters their choices, and the model suggests the best place to visit.
5. **Show Result** The system shows the suggestion and a picture of the Decision Tree to explain how the choice was made.

---

## SYSTEM REQUIREMENTS AND SPECIFICATIONS

### FUNCTIONAL REQUIREMENTS

#### 1. Upload Data

- User uploads a CSV file with tourist information.

#### 2. Clean Data

- System removes missing values.
- Text is changed into numbers.

#### 3. Train Model

- A Decision Tree is trained using the cleaned data.

#### 4. Give Recommendation

- User enters their choices.
- The model suggests the best place to visit.

#### 5. Show Results

- The system shows the suggested place.
- A picture of the Decision Tree explains the decision.

---

### NON-FUNCTIONAL REQUIREMENTS

- **Easy to use:** The website is simple for anyone.
- **Fast:** The system gives answers quickly.
- **Reliable:** Same input gives the same output every time.
- **Easy to update:** Code is organized in parts.
- **Works anywhere:** Runs on Windows, Linux, and macOS.
- **Clear:** Shows how it makes decisions

---

### HARDWARE REQUIREMENTS

- Intel i3/i5 processor
- 4 GB RAM (8 GB better)
- 512 MB free space
- Normal monitor

---

### SOFTWARE REQUIREMENTS

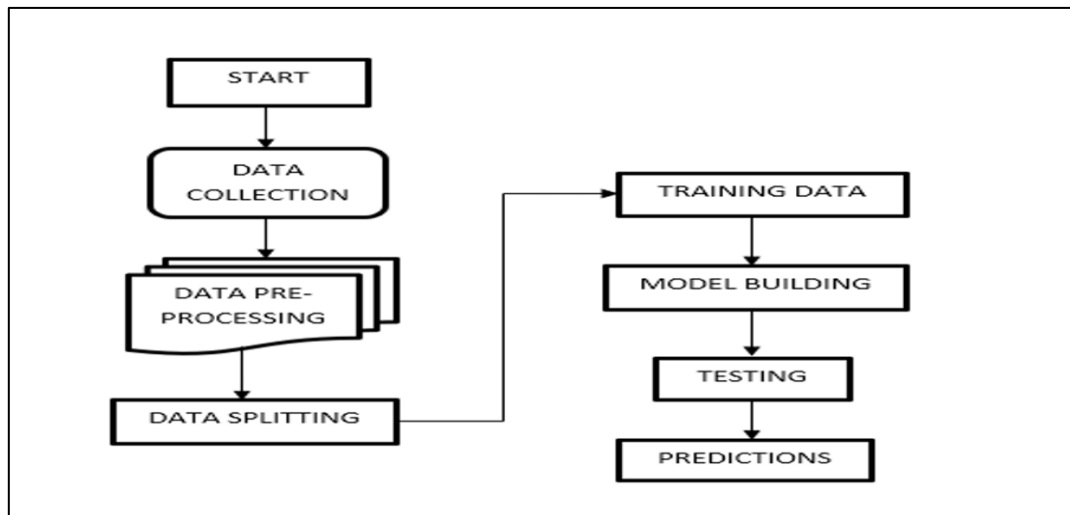
- Windows / Linux / macOS
- Python 3
- Flask
- Pandas, Scikit-learn, Graphviz
- VS Code / PyCharm / Jupyter
- Chrome / Edge / Firefox browser

## LITERATURE SURVEY

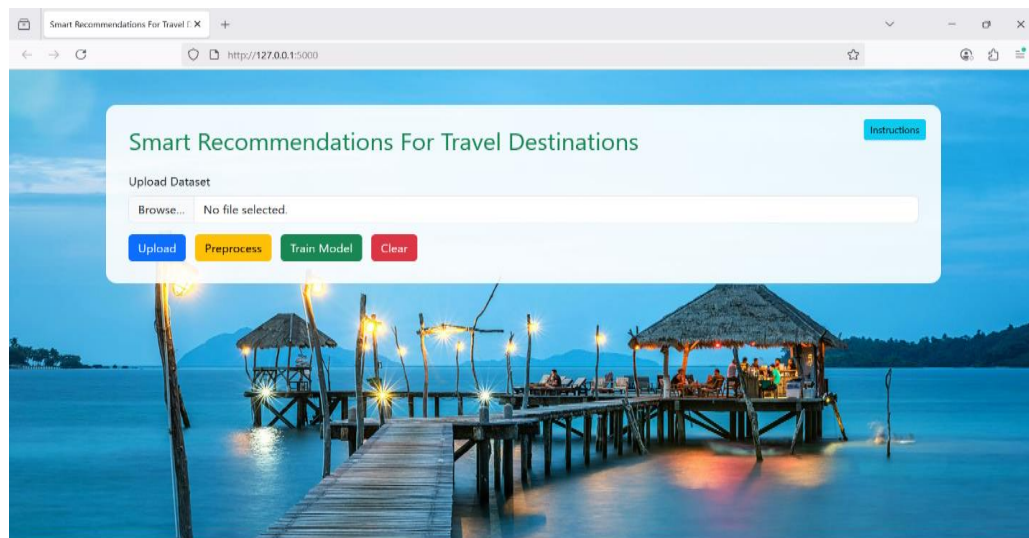
Recommendation Systems (RS) are widely used today in many areas such as online shopping, movies, healthcare, and tourism. Because so much information is available online, tourists often get confused and find it hard to choose the right place to visit. To solve this problem, researchers have created Travel Recommendation Systems (TRSs) that help users make better travel choices.

TRSs mainly use three types of methods: Collaborative Filtering, Content-Based Filtering, and Hybrid Models. Recently, simple machine learning models like Decision Trees have also been used because they are easy to understand and clearly show how recommendations are made.

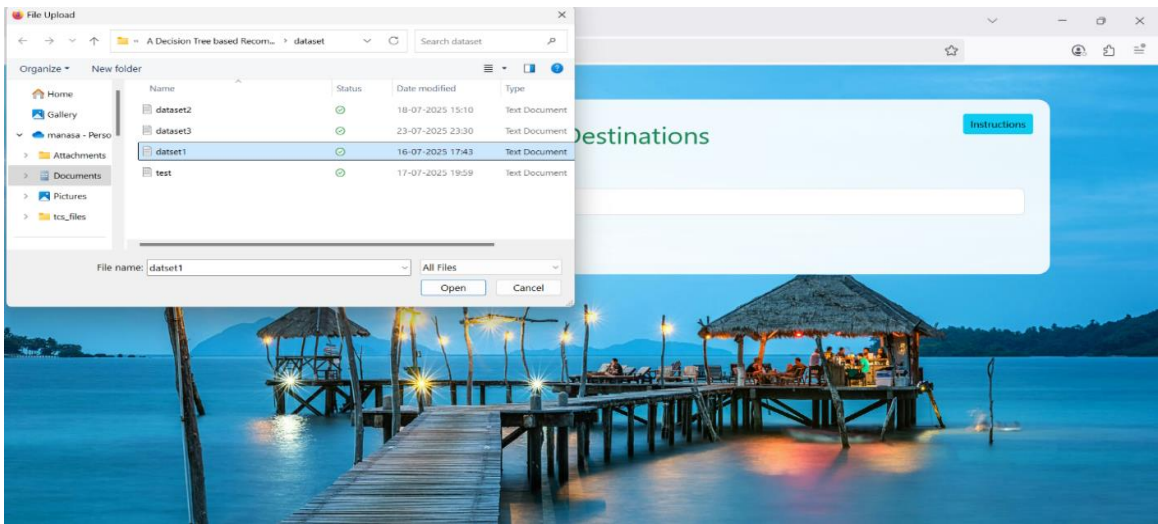
## SYSTEM DESIGN



## IMPLEMENTATION AND RESULTS



Home Page of the Tourist Recommendation System



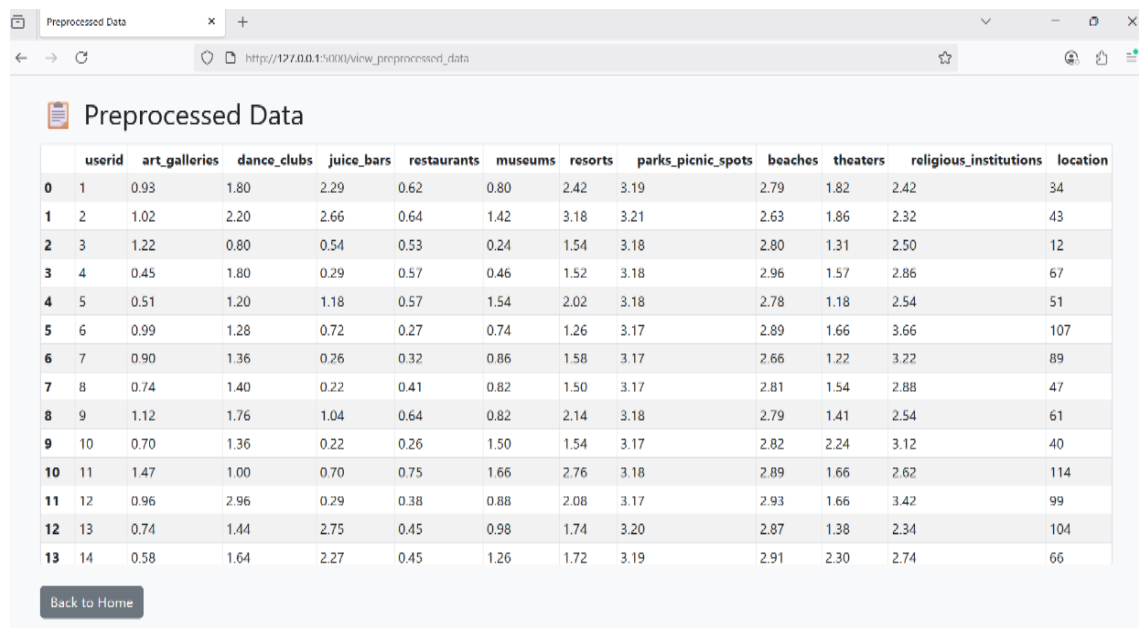
Dataset Upload Screen

Uploaded Dataset

	userid	art_galleries	dance_clubs	juice_bars	restaurants	museums	resorts	parqs_picnic_spots	beaches	theaters	religious_institutions	
0	1	0.91	1.80	2.29	0.62	0.80	2.42	3.19	2.79	1.82	2.42	Amsterdam Heining
1	2	1.02	2.20	2.66	0.64	1.42	3.18	3.21	2.63	1.86	2.32	Amsterdam Jachthaven ybur
2	3	1.22	0.80	0.54	0.53	0.24	1.54	3.18	2.80	1.31	2.50	Amsterdam Burt Huumstra Kud
3	4	0.45	1.80	0.29	0.57	0.46	1.52	3.18	2.96	1.57	2.86	Amsterdam Ruigoord Ker
4	5	0.51	1.20	1.18	0.57	1.54	2.02	3.18	2.78	1.18	2.54	Amsterdam Inetja In De Alk
5	6	0.99	1.28	0.72	0.27	0.74	1.26	3.17	2.89	1.66	3.66	Amsterdam eukenple
6	7	0.90	1.36	0.26	0.32	0.86	1.58	3.17	2.66	1.22	3.22	Amsterdam The Roya
7	8	0.74	1.40	0.22	0.41	0.82	1.50	3.17	2.81	1.54	2.88	Amsterdam Kermis Ubur
8	9	1.12	1.76	1.04	0.64	0.82	2.14	3.18	2.79	1.41	2.54	Amsterdam Nederlandse Bugh
9	10	0.70	1.36	0.22	0.26	1.50	1.54	3.17	2.82	2.24	3.12	Amsterdam Uburg vulgens mij
10	11	1.47	1.00	0.70	0.75	1.66	2.76	3.18	2.89	1.66	2.62	Amsterdam sdorperpold
11	12	0.96	2.96	0.29	0.18	0.88	2.88	1.17	2.91	1.66	1.42	Amsterdam Voethalclub Nieuw
12	13	0.74	1.44	2.75	0.45	0.98	1.74	3.20	2.87	1.38	2.34	Amsterdam aarlemmerw
13	14	0.58	1.64	2.27	0.45	1.26	1.72	3.19	2.91	2.30	2.74	Amsterdam Rondje Ubur

Back to Home

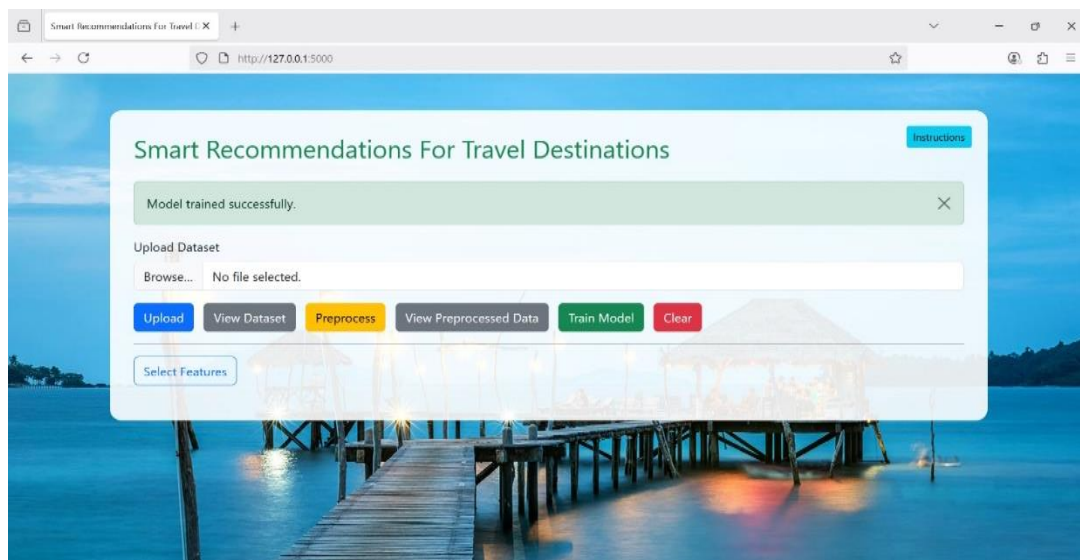
Confirmation of Dataset Upload



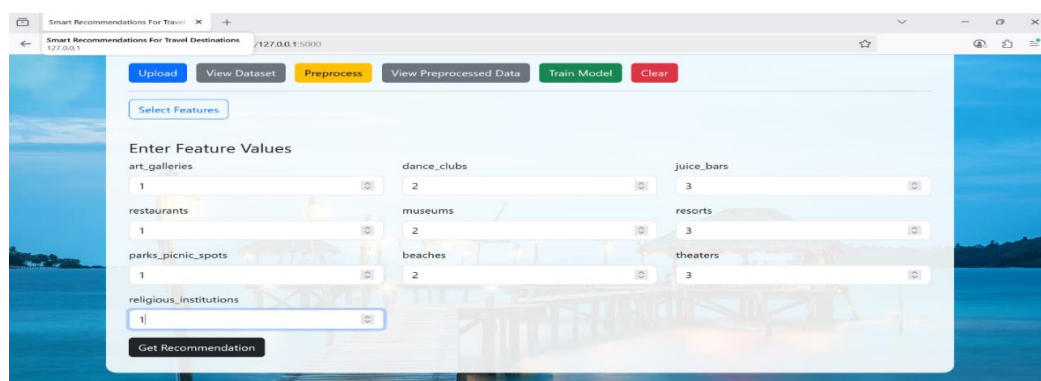
The screenshot shows a web browser window with the title 'Preprocessed Data'. The address bar displays 'http://127.0.0.1:5000/view\_preprocessed\_data'. The main content area features a table with 13 columns: 'userid', 'art\_galleries', 'dance\_clubs', 'juice\_bars', 'restaurants', 'museums', 'resorts', 'parks\_picnic\_spots', 'beaches', 'theaters', 'religious\_institutions', and 'location'. The table contains 14 rows of data, indexed from 0 to 13. Below the table is a 'Back to Home' button.

	userid	art_galleries	dance_clubs	juice_bars	restaurants	museums	resorts	parks_picnic_spots	beaches	theaters	religious_institutions	location
0	1	0.93	1.80	2.29	0.62	0.80	2.42	3.19	2.79	1.82	2.42	34
1	2	1.02	2.20	2.66	0.64	1.42	3.18	3.21	2.63	1.86	2.32	43
2	3	1.22	0.80	0.54	0.53	0.24	1.54	3.18	2.80	1.31	2.50	12
3	4	0.45	1.80	0.29	0.57	0.46	1.52	3.18	2.96	1.57	2.86	67
4	5	0.51	1.20	1.18	0.57	1.54	2.02	3.18	2.78	1.18	2.54	51
5	6	0.99	1.28	0.72	0.27	0.74	1.26	3.17	2.89	1.66	3.66	107
6	7	0.90	1.36	0.26	0.32	0.86	1.58	3.17	2.66	1.22	3.22	89
7	8	0.74	1.40	0.22	0.41	0.82	1.50	3.17	2.81	1.54	2.88	47
8	9	1.12	1.76	1.04	0.64	0.82	2.14	3.18	2.79	1.41	2.54	61
9	10	0.70	1.36	0.22	0.26	1.50	1.54	3.17	2.82	2.24	3.12	40
10	11	1.47	1.00	0.70	0.75	1.66	2.76	3.18	2.89	1.66	2.62	114
11	12	0.96	2.96	0.29	0.38	0.88	2.08	3.17	2.93	1.66	3.42	99
12	13	0.74	1.44	2.75	0.45	0.98	1.74	3.20	2.87	1.38	2.34	104
13	14	0.58	1.64	2.27	0.45	1.26	1.72	3.19	2.91	2.30	2.74	66

### Preprocessed Data Ready for Training

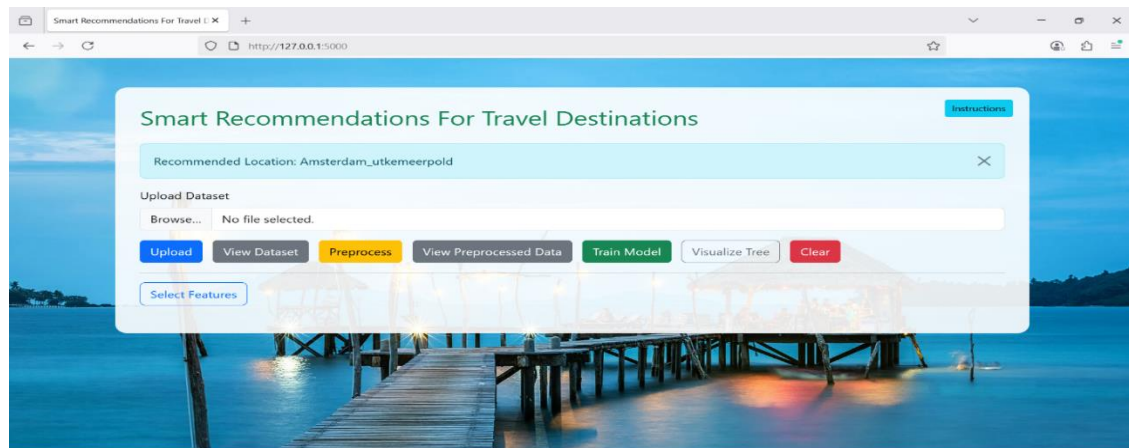


### Training Model Interface

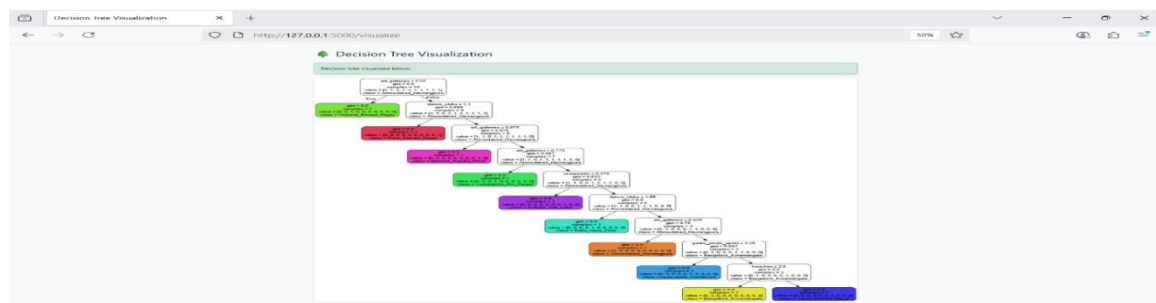


The screenshot shows a web browser window with the title 'Smart Recommendations For Travel Destinations'. The address bar displays 'http://127.0.0.1:5000'. The main content area features a 'Select Features' button. Below this, there is an 'Enter Feature Values' section with input fields for various features: 'art\_galleries' (1), 'restaurants' (1), 'parks\_picnic\_spots' (1), 'religious\_institutions' (1), 'dance\_clubs' (2), 'museums' (2), 'beaches' (2), 'juice\_bars' (3), 'resorts' (3), and 'theaters' (3). A 'Get Recommendation' button is located at the bottom of the form. The background of the interface shows a scenic view of a wooden pier extending into a body of water at dusk.

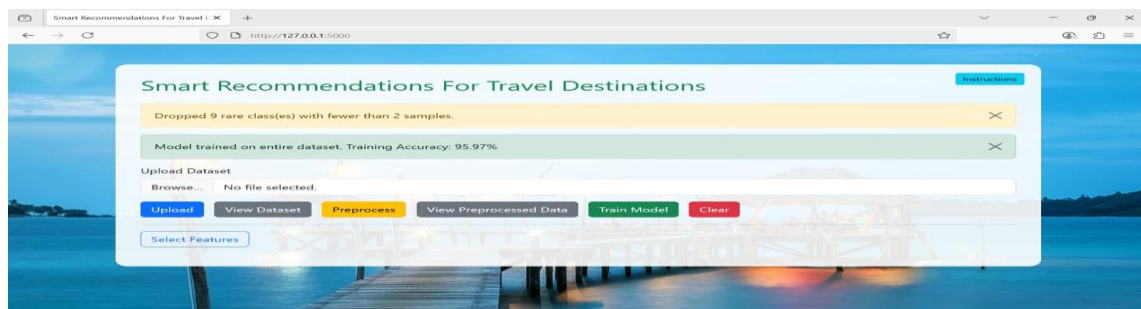
### Input Selection for Prediction



Recommendation Output Screen



Decision Tree Visualization Using Graphviz



Model Accuracy Screen

## CONCLUSION AND FUTURE SCOPE

This project created a Tourist Recommendation System that uses a Decision Tree and works through a Flask web application. The system solved the main problems seen in older travel recommendation systems:

- **Too much information:** The machine learning model looks at the data automatically and suggests suitable places to visit.
- **Hard to use:** The web interface is simple, so anyone can upload a dataset, train the model, and get recommendations easily.
- **No clear explanation:** The system shows the Decision Tree using Graphviz, helping users see how the recommendation was made.

The system also showed good results:

1. Data was cleaned properly using Pandas and Label Encoding.
2. The Decision Tree model gave accurate and dependable predictions.
3. The Flask app made the system easy to use and quick to respond.
4. The tree diagram helped users trust the recommendations.

Overall, the system is accurate, easy to use, and clear in how it makes decisions, making it useful for real travel recommendation needs.

---

**REFERENCES**

---

- Adomavicius & Tuzhilin (2005). Next generation recommender systems.
- Ricci et al. (2015). Recommender Systems Handbook.
- Jannach et al. (2010). Recommender Systems.
- Quinlan (1986). Induction of decision trees.
- Breiman et al. (1984). CART.
- Shani & Gunawardana (2011). Evaluating recommender systems.
- Flask Documentation (2025).
- McKinney (2010). Pandas.
- Pedregosa et al. (2011). Scikit-learn.
- Graphviz Documentation (2025).