Smart Travel Advisor – Final Report

---

1. Introduction

The Smart Travel Advisor project aims to provide a user-friendly solution for flight fare prediction using natural language queries. With the growing demand for intelligent and interactive applications in travel tech, this system bridges the gap between traditional ML models and modern user interfaces by integrating a trained scikit-learn model into a Streamlit app, all packaged using Docker for seamless deployment.

---

2. Problem Statement

Users often want quick and simple answers to questions like:

"What is the flight cost from Hyderabad to Goa on July 15 via Air India?"

The challenge is to parse such queries, extract relevant structured data (like cities, date, airline, time), transform them into a machine-readable format, and deliver accurate fare estimates instantly.

---

3. Dataset & Preprocessing

- **Source: A publicly available dataset of airline flight prices**
- **Size: ~10,000 rows**
- **Features:**
    - **Categorical: Airline, Source, Destination, Total_Stops**
    - **Temporal: Date_of_Journey, Dep_Time, Arrival_Time**
    - **Numerical: Duration (converted to minutes)**

Key Preprocessing Steps:

- **Missing values dropped**
- **Time-related features extracted (hour, minute, day, month)**
- **Duration parsed to minutes**
- **Categorical values label-encoded using fallback to "Unknown"**

Encoders and model were saved using joblib and reused during inference.

---

4. Model Development & Evaluation

- **Algorithm: RandomForestRegressor (chosen for robustness and interpretability)**
- **Train/Test Split: 80/20**

- **Metrics:**
    - **MAE (Mean Absolute Error): ~1100**
    - **R^2 Score: ~0.89**

The model was trained to handle both seen and unseen combinations of flight routes and airlines, thanks to flexible encoding.

Exported model file:

models/flight_price_model_rf.joblib

Encoders:

models/Airline_encoder.pkl

models/Source_encoder.pkl

models/Destination_encoder.pkl

models/Total_Stops_encoder.pkl

---

5. Streamlit App Architecture

The front end was built using Streamlit. It:

- **Accepts a natural language query from the user**
- **Parses query using regular expressions and rule-based mapping**
- **Converts extracted features to structured DataFrame**
- **Loads encoders & model**
- **Returns a formatted prediction**

Example input:

"How much does it cost to go to Pune from Bangalore on August 5?"

App output:

"Estimated Flight Fare: ₹7125.39"

Error handling, unknown values, and fallback mechanisms are built-in for robustness.

---

6. Dockerization & Deployment

The entire project is containerized using Docker for easy deployment on any system or cloud provider.

Dockerfile Summary:

- **Base image: python:3.10-slim**
- **Installed dependencies via requirements.txt**

- **Exposes Streamlit on port 8501**
- **Uses .env for secure API keys (OpenAI, if integrated later)**

Build and Run:

docker build -t smart-travel-app .

docker run --env-file .env -p 8501:8501 smart-travel-app

.dockerignore ensures only necessary files are copied:

.venv/

notebooks/

data/

***.csv**

***.ipynb**

***.xlsx**

***.pdf**

__pycache__/

---

7. Challenges Faced

- **Long Docker build times due to unnecessary files (fixed with .dockerignore)**
- **Torch and GPU packages pulled accidentally (resolved by cleaning requirements.txt)**
- **Fallback logic for unseen cities/airlines required extra handling**
- **Streamlit's SessionState fails in bare mode (avoided by using streamlit run properly)**

---

8. Future Enhancements

- **Deploy on Streamlit Cloud, HuggingFace Spaces, or AWS Lambda**
- **Add conversational memory and chat history**
- **Provide SHAP/LIME explanations for predictions**
- **Human-in-the-loop feedback**
- **Runtime feedback loop**
- **Modular design using Python classes**
- **Automated pipeline/workflow**

---

9. Conclusion

The Smart Travel Advisor demonstrates how a traditional ML model can be made intuitive and deployable using modern tools. The integration of Streamlit, Docker, and simple NLP parsing allows it to serve as a base for intelligent travel applications, virtual assistants, or price prediction tools.

This solution is robust, extendable, and ready for production use or integration into larger systems.