AI-Driven Vehicle Matching & Dynamic Pricing System

Objective

Design and implement an AI-powered system that recommends the best available vehicle for a ride request based on predicted ETA, dynamic pricing, and user preference. This project blends machine learning, geospatial analytics, and real-world decision modeling.

1. Data Acquisition

Students may collect or generate their own dataset:

- Use real-world data (e.g., OpenStreetMap, Google Maps API, or public ride datasets from Kaggle), OR
- Create synthetic data simulating trips, vehicles, traffic, and demand patterns.

Each record should include: origin_lat, origin_lon, dest_lat, dest_lon, timestamp, vehicle_type, trip_distance, trip_duration, and fare. Optionally simulate time-of-day effects (rush hours), demand surges, and traffic delays.

2. Core ML Tasks

1. ETA Prediction - Predict pickup and trip duration.
   Models: Linear Regression, LightGBM, or simple neural network. Metrics: MAE, RMSE, MAPE.

2. Demand Forecasting - Predict short-term ride demand per region or grid cell.
   Models: Prophet, XGBoost, or LSTM. Metrics: RMSE, MAE.

3. Dynamic Pricing - Compute surge multipliers using predicted demand or supply-demand ratios.
   Approaches: Rule-based baseline + regression or reinforcement learning policy.

4. Vehicle Ranking - Rank available vehicles using ETA, price, and user preference ('fastest', 'cheapest', or 'balanced'). Approach: Weighted scoring or learn-to-rank model.

3. API Layer (Light Integration)

Implement minimal APIs (using Flask or FastAPI):

- POST /vehicles/update - Update vehicle location/status.
- POST /ride/quote - Return top-k recommended vehicles with ETA, cost, and category.

Optionally integrate Google Maps Distance Matrix API or OpenStreetMap routing for real distances and ETAs.

4. Deliverables

1. Dataset (real or synthetic) with description.
2. Model training notebooks or scripts.
3. Saved model artifacts (.joblib / .pkl / .onnx).
4. Inference script or small API service.
5. Evaluation notebook with metrics and plots.
6. README with setup instructions and example usage.
7. Unit tests for data and model components.

5. Evaluation Criteria

- Data realism & feature engineering - 15%
- ETA & demand model accuracy - 25%
- Dynamic pricing logic - 15%
- Vehicle ranking performance - 15%
- Code quality & documentation - 10%
- Reproducibility & clarity - 10%
- Insights & analysis - 10%

6. Tools & Timeline

Tools: Python 3.8+, Pandas, NumPy, Scikit-learn, LightGBM/XGBoost, PyTorch or TensorFlow (optional), Geopandas/H3 for geospatial features, and FastAPI or Flask for demonstration APIs.

Timeline (1 Week):

- Phase 1: Data exploration, feature engineering, and model training.
- Phase 2: Model evaluation, integration into API, and report preparation.

7. Submission & Contact

Submission: GitHub repository link with all deliverables.

For queries: WhatsApp 9886498481 or email shricharan@unloadin.com