

Event-Aware Predictive Route Recommendation Using Random Forest and A* Search

Dr. Sandhiya S

Assistant Professor, Department of Computer Science, Women's Christian College, Chennai, Tamil Nadu, India

Email: sandhiya[at]wcc.edu.in

Abstract: Urban traffic conditions can change considerably during peak hours and around public events. Conventional route selection generally reacts to the traffic condition observed at the time of travel, whereas a predictive approach can estimate likely travel duration before a journey begins. This paper presents a student-level prototype for event-aware route recommendation that combines supervised learning with graph-based path search. A Random Forest regression model is used to estimate travel duration from route origin, destination, distance, travel hour, and an event indicator. The predicted duration is then used as the cost of road segments in an A* search procedure. A small urban road graph representing selected locations is used for demonstration. The implementation consists of a Python-based machine-learning module, a Flask backend, and a React user interface. The project report records an R^2 value of 0.9687 for the main modeling experiment, while a separately generated event-focused dataset produced an R^2 value of 0.9291. These results indicate that the prototype can learn useful relationships in the prepared dataset and can translate predicted travel time into route-selection decisions. The work is intentionally presented as an academic prototype rather than a live city-scale navigation system; real-time traffic feeds, geographical coordinates, larger datasets, and extensive field validation remain future work.

Keywords: Traffic prediction, route recommendation, Random Forest regression, A* search, event-aware routing, smart mobility, machine learning

1. Introduction

Traffic congestion is a recurring challenge in urban transportation. Travel duration is affected not only by the normal variation between morning, afternoon, evening, and night periods, but also by temporary changes associated with concerts, sports events, festivals, public gatherings, and other activities. Reviews of traffic congestion research show that travel time, speed, and level-of-service measures are commonly used to characterize congestion [1]. Route planning research has also explored data-driven optimization and adaptive decision making for urban transport networks [2].

The project considered in this paper addresses a narrower and more practical problem: predicting the travel time of selected routes when the travel hour and presence of an event are known, and then using the prediction to select a route in a small road graph. This modification makes the work appropriate for a postgraduate student project while retaining a meaningful machine-learning and algorithmic component.

The proposed prototype follows a two-stage decision process. First, a Random Forest regression model learns the relationship between route attributes and travel duration. Second, the predicted duration is supplied as an edge cost to an A* search routine. Therefore, the machine-learning component estimates the cost, while the graph-search component selects a path with a comparatively lower predicted cost. This separation also makes the system easier to understand, implement, and extend.

1.1 Problem Statement

A simple route planner may select a path using fixed distance or a static travel-time estimate. Such a strategy does not explicitly represent temporary event-related changes. The problem addressed here is to build a prototype that uses

historical or prepared traffic observations together with an event flag to estimate travel duration and then recommend a route using the estimated duration.

1.2 Objectives

- Prepare a route-level dataset containing time, distance, event, origin, destination, and travel-duration information;
- Train a Random Forest regression model for travel-duration estimation;
- Represent selected urban locations as nodes and connections as graph edges;
- Use predicted travel duration as a dynamic route cost in A* search;
- Classify the resulting route condition into Green, Orange, or Red using project-defined thresholds; and
- Evaluate the prototype using regression performance and functional route-generation results.

2. Related Work

Research on route planning has used optimization algorithms, machine learning, and traffic information to improve transport efficiency. Blessing and Yedgarian discuss data-driven route planning in urban transport and emphasize the value of adaptive approaches for changing traffic conditions [2]. Their work motivates the use of predictive information before selecting a route.

Kumar, Kumar, and Das review traffic-congestion measurement approaches and identify travel time, speed, and level-of-service measures as important categories for congestion analysis [1]. Their review also highlights the importance of improved data collection and modern analytical techniques.

Schneider, Miller, and Holik describe a GIS-based route optimization model for winter-maintenance operations,

showing how network analysis can support operational route planning [3]. Although the application domain differs from urban commuter routing, it demonstrates the usefulness of graph-based optimization for transportation tasks.

Delling, Pajor, and Wagner investigated accelerated multi-modal route planning using access nodes and demonstrated the importance of efficient shortest-path processing for large transportation networks [4]. The present project does not attempt to reproduce large-scale multi-modal routing. Instead, it uses A* on a small academic graph so that the complete prediction-to-routing workflow can be demonstrated clearly.

A key distinction of the present work is the combination of a simple event-aware travel-time prediction model with a small graph-search layer. The objective is not to compete with commercial navigation platforms, but to demonstrate how a learned cost can influence a route-selection algorithm in a manageable student project.

3. Proposed System

The proposed system is an event-aware route recommendation prototype. Its input consists of an origin, destination, travel hour, and event status. The backend prepares the model input, predicts the travel duration for candidate connections, and executes A* search. The final response contains the selected path, estimated travel time, and traffic category.

3.1 System Architecture

- User Interface: A React-based interface accepts origin, destination, hour, and event status.
- Application Layer: A Flask service receives the request and coordinates prediction and route search.
- Prediction Layer: A saved Random Forest regression model estimates travel duration for candidate edges.
- Graph Layer: A small manually defined road graph represents the academic test network.
- Optimization Layer: A* search accumulates predicted travel duration and selects a lower-cost path.
- Output Layer: The system displays the selected path, predicted duration, and traffic category.

3.2 Workflow

The workflow begins when a user selects an origin and destination and enters the travel hour. The event option is set to 0 when no relevant event is assumed and to 1 when an event is expected. The backend obtains the distance associated with each candidate graph edge and passes the features to the trained model. Predicted travel durations become the edge costs used by A*. After the search terminates, the accumulated predicted duration is mapped to a traffic category using the project thresholds.

4. Dataset and Preprocessing

The original project report describes a structured route dataset containing Route ID, origin, destination, date, day of week, time of day, time category, traffic condition, distance, travel

duration, and event-related attributes. The event and time-based data were combined for model preparation. Categorical variables were converted to numerical representations before model training. □ filecite□turn0file0□L783-L811□

For the student-level paper, the feature set is simplified to the variables most directly connected with the proposed prototype: origin, destination, distance, hour, event flag, and travel duration. Origin and destination are one-hot encoded. The event flag is binary, with 0 representing no event and 1 representing an event. The target variable is travel duration in minutes.

4.1 Academic Event-Based Data Generation

The source project also contains a separate Python routine that creates an 800-record event-based dataset. It selects origins and destinations from a small set of Chennai locations, generates distances between 5 and 15 km, selects travel hours between 6 and 22, and adjusts a base travel time for peak periods and event conditions. This dataset is useful for demonstrating the complete machine-learning workflow, but it should be described as generated or simulated academic data rather than as live traffic observations. The original code reports 800 generated records and a model R^2 of 0.929112... for that experiment. □ filecite□turn0file0□L1206-L1328□

4.2 Preprocessing

- Clean column names and standardize feature labels;
- Convert date/time information into usable hour-based features where required;
- Encode categorical origin and destination values using one-hot encoding;
- Combine compatible observations while preserving the event indicator;
- Separate input variables from the travel-duration target; and
- Split the prepared data into training and testing subsets using an 80:20 ratio.

5. Machine Learning and Route Optimization

5.1 Random Forest Regression

Random Forest regression is used because the relationship between travel duration and the selected input variables may not be purely linear. The model constructs multiple decision trees from randomized subsets of the training data and combines their outputs. In the project implementation, 200 trees are used. The input features include encoded origin and destination values, distance, hour, and event flag, while travel duration is the prediction target.

For an unseen route request, the backend creates a feature vector with the same columns used during training. The corresponding route distance, hour, event flag, and one-hot location indicators are assigned before the vector is passed to the saved model.

5.2 A* Route Search

A* is used to search the small road graph. For a node n , the evaluation function is:

$$f(n) = g(n) + h(n)$$

Here, $g(n)$ represents the accumulated predicted travel duration from the starting point to node n , while $h(n)$ is the heuristic estimate from n to the destination. In the present academic implementation, the heuristic is a simple constant. The predicted Random Forest duration is therefore the main cost used to compare candidate paths.

The implementation uses a small graph containing Marina, Chepauk, Nungambakkam, and Kilpauk, with manually specified connections. This deliberately limited graph keeps the project reproducible and makes the effect of changing event conditions easy to observe. It should not be interpreted as a complete representation of Chennai's road network.

6. Implementation

The prototype uses Python with pandas and scikit-learn for data preparation and model training. The trained model is serialized with joblib. A Flask application exposes a route-prediction endpoint. The React interface sends the user request to the Flask service and displays the response. The original implementation confirms this separation between the React interface, Flask backend, Random Forest model, and A* search procedure. [filecite turn0file0 L1345-L1603](#)

6.1 Route Request

The request contains four main values: origin, destination, hour, and event flag. The backend validates the selected origin, runs A*, and returns the optimal path, predicted travel time, and traffic condition. The React interface provides selection controls for these values and presents the resulting path and duration. [filecite turn0file0 L1615-L1815](#)

6.2 Traffic Classification

Predicted Duration	Prototype Category
< 20 min	Green
20-< 40 min	Orange
≥ 40 min	Red

7. Experimental Results and Discussion

The main project experimental reports an R^2 value of 0.9687 on the held-out test portion. The source code shows an 80:20 train-test split and a Random Forest Regressor with 200 estimators, followed by evaluation using the model score. The reported score is 0.9687548146632367. [filecite turn0file0 L1174-L1198](#)

A second experiment uses the separately generated 800-record event-based dataset. With the same general Random Forest approach and an 80:20 split, the reported R^2 is 0.9291121876234291. This result is useful as an additional academic check, but because the data were generated using predefined rules, it should not be interpreted as evidence of

real-world traffic prediction accuracy.
[filecite turn0file0 L1214-L1328](#)

Experiment	Data Setting	Reported R^2
Main project model	Prepared/combined project dataset	0.9687
Event-focused model	800-record generated academic dataset	0.9291

The results indicate that the selected features contain useful information for estimating the prepared travel-time values. The route-search component then converts those estimates into a decision. When an event flag changes the predicted cost of one or more connections, A* can prefer another available path. This demonstrates the central concept of the project: prediction and optimization can be connected through a common travel-time cost.

However, the high R^2 values should be interpreted within the limits of the dataset. A strong score on a small or synthetically generated dataset does not establish performance under live traffic, weather variation, accidents, road closures, or large-scale network conditions. Therefore, the contribution of this paper is the working prototype and integration of the two techniques, rather than a claim of production-grade navigation accuracy.

8. Discussion

The prototype demonstrates three practical observations. First, event information can be represented as a simple feature without redesigning the entire prediction pipeline. Second, predicted travel duration provides an intuitive bridge between machine learning and graph algorithms because it can be directly used as an edge cost. Third, a full-stack implementation makes the concept accessible to end users without requiring them to interact with the machine-learning model directly.

The student-oriented design also has deliberate limitations. The graph is manually defined, the heuristic is simple, the event representation is binary, and the dataset does not provide continuous GPS or sensor streams. These limitations make the system easier to implement and explain, but they prevent direct comparison with commercial navigation systems.

9. Limitations and Future Enhancement

- Expand the graph to include more road segments and geographically meaningful coordinates.
- Replace the binary event flag with event type, expected attendance, start/end time, and distance from the event location.
- Integrate live traffic or map APIs after appropriate API and data-use permissions are obtained.
- Add weather, road incidents, day-of-week, and historical speed information.
- Compare Random Forest with Gradient Boosting or XGBoost using the same train/test protocol.
- Use a geographically meaningful A* heuristic instead of the constant academic heuristic.

- Evaluate the system on independently collected real-world observations and report MAE, RMSE, and R^2 together.
- Deploy the prototype on a cloud platform for controlled multi-user testing.

10. Conclusion

This paper presented a compact event-aware route recommendation framework that combines Random Forest regression with A* search. The machine-learning component estimates travel duration from route and contextual variables, while the graph-search component uses the estimates to select a path. The prototype was implemented using Python, Flask, and React and was tested on the prepared project data. The reported R^2 values of 0.9687 for the main experiment and 0.9291 for the separately generated event-focused experiment show that the prepared data contain strong predictive patterns. At the same time, the results are interpreted as academic prototype results rather than real-world navigation accuracy. The proposed structure provides a clear foundation for future work involving larger datasets, geographical road networks, real-time data, richer event descriptions, and more rigorous validation.

References

- [1] M. Kumar, K. Kumar, and P. Das, "Study on road traffic congestion: A review," in *Recent Trends in Communication and Electronics*, pp. 230–240, 2021, doi: 10.1201/9781003193838-43.
- [2] M. Blessing and V. A. Yedgarian, "Optimizing Route Planning for Enhanced Network Efficiency in Urban Transport Systems," 2024.
- [3] W. Schneider, T. M. Miller, and W. A. Holik, *Route Optimization*, Report FHWA/OH-2016-20, Ohio Department of Transportation, Nov. 2016.
- [4] D. Delling, T. Pajor, and D. Wagner, "Accelerating Multi-modal Route Planning by Access-Nodes," in *Proc. 17th Annu. Eur. Symp. Algorithms (ESA)*, *Lecture Notes in Computer Science*, vol. 5757, pp. 587–598, 2009, doi: 10.1007/978-3-642-04128-0_53.
- [5] L. Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5–32, 2001.
- [6] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.