CS221 Project Final Report

TripCompassSF: Personalized, Budget and Time-Aware Itineraries
Team Members: Isaias Martinez, Varsha Saravanan
Emails: isaiasm@stanford.edu, vsaravan@stanford.edu

1 Introduction

Developing a personalized travel itinerary requires balancing a traveler's interests, time constraints, budget, and the practical limitations of visiting real-world locations. In this project, we design a system that automatically generates a full daily schedule of activities tailored to an individual user. The system takes as input a user profile—including interests, travel pace, budget, and trip length—and a set of points of interest (POIs) with attributes such as cost, duration, popularity, tags, opening hours, and geographic location. Using this information, the system produces a feasible itinerary that aims to maximize predicted user satisfaction. To support this, we construct a synthetic but realistic dataset of user–POI interactions, train both a baseline and a more advanced satisfaction model, and integrate the stronger model into a beam search planner that assembles high-quality itineraries. This end-to-end pipeline allows us to explore how learned satisfaction signals can guide structured search methods to produce coherent, personalized travel plans.

2 Literature Review

Prior work on itinerary generation spans satisfaction modeling, constraint aware routing, and search. Halder et al. (2024) survey personalized itinerary systems and show that many methods learn a satisfaction function over user and POI features rather than rely on hand tuned heuristics. This motivates our learned user–POI model and our use of contextual signals such as time, cost, and popularity. Horstmannshoff et al. (2024) frame planning as a multi objective optimization problem balancing time, budget, and enjoyment, while we optimize a single learned satisfaction score and treat travel time and budget as soft penalties, suggesting a natural extension to explicitly multi objective search. Work on the Tourist Trip Design Problem, including Marzal et al. (2024), emphasizes time windows and opening hour constraints, which motivates our explicit feasibility checks and the structure of our search state. Another line of work combines learned utility models with sequential recommendation or richer representation learning. Tsai et al. (2023, 2025) and Xiao et al. (2025) use sequence models that encode user trajectories to recommend routes conditioned on spatial and temporal context. In contrast, we use a shallow feed forward network over engineered features such as interest match times popularity and pace times duration, trading expressiveness for transparency and easier inspection. Finally, several systems integrate learned preferences with search, for example hybrid A star or beam search engines that generate constrained walking tours. Our beam search planner fits this pattern but uses only the learned satisfaction score to guide search. Reinforcement learning approaches to related routing problems, such as Gama et al. (2020), suggest future work in which both the utility function and the search policy are learned using our synthetic environment.

3 Dataset

We construct a two-layer dataset for this project: a curated catalog of San Francisco points of interest (POIs) and a synthetic user–POI interaction dataset used to train our satisfaction models.

3.1 POI Catalog Construction

We aggregate six public sources from the San Francisco Open Data Portal, supplemented with curated attractions, into a unified POI catalog. The sources include:

- **Base POIs**: 1,246 entries aggregated from OpenTripMap and Wikivoyage.
- **Curated Landmarks**: 50 major attractions.
- **Recreation & Parks**: 244 properties from SF Recreation and Parks.
- **Registered Businesses**: 600 restaurants and shops.
- **Muni Transit Stops**: 400 transit stops.
- **SFO Airport Facilities**: 81 airport-related facilities.

We normalize schemas across sources, deduplicate entries by name and rounded coordinates, and prioritize curated sources when conflicts occur. The resulting catalog contains 2,261 unique POIs.

3.2 Category Distribution and Schema

The catalog spans eight categories: restaurants (1,068, 47.2%), transit stops (817, 36.2%), parks (252, 11.2%), airport facilities (81, 3.6%), nightlife venues (17, 0.8%), landmarks (12, 0.5%), museums (10, 0.4%), and shopping locations (4, 0.2%). Each POI record includes a unique identifier, name, category, visit duration (hours), cost (0–50 USD), opening hours (`open_start`, `open_end`), geographic coordinates (latitude and longitude), a popularity score (3.0–5.0), and a comma-separated list of tags.

3.3 Data Preprocessing and Quality Checks

Missing values are filled using category-specific statistics. Costs and visit durations are sampled from realistic distributions (for example, restaurant costs between \$10 and \$50, and museum visit durations between 1.5 and 3 hours). We enforce validity constraints on coordinates (latitude 37.0–37.95, longitude -123.1 to -122.0), opening hours, and durations, and log minor anomalies such as out-of-range coordinates or overnight hours. These issues are rare and non-blocking for model training.

3.4 Synthetic User–POI Interaction Dataset

We generate 30 synthetic user profiles. Each profile includes a pace parameter (slow, moderate, or fast), trip length (2–5 days), budget (300–1,200 USD), hotel location (randomized around downtown San Francisco), and seven interest dimensions (food, museums, outdoors, parks, shopping, nightlife, culture) sampled uniformly in $[0, 1]$.

Pairing each user with every POI yields 67,830 user–POI examples. For each pair, we compute a satisfaction label in $[1, 5]$ using a nonlinear scoring function that combines:

- interest–tag match,
- normalized popularity,
- pace fit,
- interaction terms (e.g., match $\times$ popularity, pace $\times$ duration),
- a cost penalty factor,
- small Gaussian noise with standard deviation $\sigma = 0.05$.

This scoring function defines the ground-truth satisfaction signal that our models aim to learn.

3.5 Feature Engineering and Splits

For each user–POI pair we construct 18 features, including:

- base features (duration, cost, popularity, `open_start`, `open_end`, pace indicators, `interest_match`, `pace_fit`),
- nonlinear transforms (e.g., popularity^2, duration^2, cost per hour),
- interaction terms (e.g., match $\times$ popularity, pace $\times$ duration, cost $\times$ interest),
- Euclidean distance from the user's hotel to the POI.

All numerical features are standardized using a `StandardScaler` fitted on the training set.

We randomly split the 67,830 pairs into training, validation, and test sets using an 80/10/10 ratio with a fixed random seed of 42, resulting in 54,264 training examples, 6,783 validation examples, and 6,783 test examples. The same split is used for both the baseline and main models.

3.6 Limitations and Biases

The dataset has several limitations. Satisfaction labels are synthetic and do not reflect real user ratings, which limits external validity. Geographic coverage is restricted to San Francisco and the POI distribution is skewed toward restaurants and transit stops, introducing domain-specific bias. Real-world factors such as seasonality, weather, and crowding are not modeled, and some prices are imputed rather than observed. These constraints limit realism but are acceptable for a first iteration focused on controlled experimentation with itinerary planning in San Francisco.

4 Baseline

We use a linear regression model as our baseline for predicting user–POI satisfaction, where a point of interest (POI) is any candidate location such as a restaurant, park, or landmark. This provides a simple, interpretable reference point and allows us to quantify the benefits of using a more expressive model later.

4.1 Model Choice and Motivation

Our baseline is scikit-learn's `LinearRegression` (ordinary least squares). The model trains quickly, is easy to interpret, and captures linear relationships between features and satisfaction. It also serves as a sanity check to ensure that our feature engineering produces meaningful predictive signals before introducing nonlinear models.

4.2 Input Representation and Features

For each user–POI pair, the baseline consumes a 10-dimensional feature vector consisting of:

- **POI attributes**: `duration_hours`, `cost`, `popularity`, `open_start`, `open_end`,
- **User pace (one-hot)**: `pace_slow`, `pace_moderate`, `pace_fast`,
- **Derived features**: `interest_match` and `pace_fit`.

All numerical features are standardized using a `StandardScaler` fitted on the training set to prevent scale imbalances and stabilize coefficient estimation.

4.3 Training Procedure

We apply the 80/10/10 train–validation–test split described earlier (seed = 42). The model is trained once using the closed-form least squares solution, with no regularization and no iterative optimization. The fitted scaler is reused for validation and test sets to avoid data leakage.

4.4 Inference

To score a new user–POI pair, we extract the same feature vector, standardize numerical components using the fitted scaler, and compute the linear prediction

$$\hat{y} = w^\top x + b,$$

which we interpret on the same $[1, 5]$ satisfaction scale as the synthetic labels.

4.5 Performance

On the 6,783 held-out test examples, the baseline achieves:

- Root Mean Squared Error (RMSE): 0.221
- Mean Absolute Error (MAE): 0.176
- Coefficient of Determination (R^2): 0.844

The R^2 score indicates that the linear regression explains roughly 84% of the variance in satisfaction, consistent with the partially linear structure of the ground-truth scoring function. However, it lacks the capacity to model nonlinear patterns or richer feature interactions, motivating the use of a more expressive neural model in our main approach.

5 Main Approach

Our system combines a feed-forward neural network (FFN) for predicting user–point-of-interest (POI) satisfaction with a beam search planner for constructing feasible daily and multi-day itineraries. The FFN provides personalized utility estimates for each POI, and beam search uses these estimates to build routes that respect real-world constraints such as time, cost, opening hours, and travel feasibility.

5.1 Satisfaction Prediction Model

We implement a NumPy-based FFN with one hidden layer. The model takes an 18-dimensional feature vector for each user–POI pair, consisting of base attributes (duration, cost, popularity, opening hours), nonlinear transforms, interaction terms, and a spatial distance feature.

- **Input layer**: 18 standardized features.
- **Hidden layer**: 64 units with ReLU activation.
- **Output layer**: 1 linear unit predicting satisfaction on the $[1, 5]$ scale.

Weights are initialized with Xavier (Glorot) initialization. The forward pass computes:

$$z_1 = XW_1 + b_1, \qquad a_1 = \mathrm{ReLU}(z_1), \qquad \hat{y} = a_1 W_2 + b_2.$$

Training minimizes mean squared error (MSE) using batch gradient descent with backpropagation. We train for 200 epochs with batch size 128, learning rate 0.01, and seed 42. The same `StandardScaler` used in the baseline standardizes numerical features. Validation loss is monitored to guard against overfitting.

At inference time, a user–POI feature vector is standardized and passed through the FFN to produce a predicted satisfaction score. These scores induce a personalized ranking used in itinerary generation.

5.2 Itinerary Generation via Beam Search

We formulate itinerary generation as a constrained search problem. Beam search constructs ordered sequences of POIs that fit the user's day while maximizing the accumulated predicted satisfaction.

Each search state represents a partial itinerary with the following components:

- Ordered list of selected activities.
- Current time and geographic location.
- Remaining daily budget.
- Accumulated predicted satisfaction.
- Total travel time incurred.
- Set of visited POIs (to avoid repetition across days).

5.2.1 State Expansion and Feasibility

Search begins at the user's hotel at 9:00 AM. A POI is considered feasible if:

- Travel time plus activity duration fits before 9:00 PM.
- The POI is open during the intended visit window.
- Its cost does not exceed the remaining daily budget.
- It has not been visited previously.

Travel mode is selected heuristically by distance: walking for < 0.8 km, transit for 0.8–3 km, and rideshare (Uber) beyond 3 km, each with its own travel-time adjustments. When a POI is added, the state updates its time, location, budget, and satisfaction.

5.2.2 Scoring Function and Beam Management

Each state is assigned a score:

$$\text{score} = \text{predicted\_satisfaction} - 0.1 \times \text{travel\_hours} + 0.15 \times \text{unique\_categories} - 2.0 \times \text{zero\_interest\_activities}.$$

Beam search retains the top $B = 10$–15 states at each step, pruning dominated or low-value partial itineraries. The highest scoring completed state becomes the final itinerary.

5.2.3 Interest Filtering and Multi-Day Planning

Before search, POIs are ranked using:

$$\text{candidate\_score} = 1.2 \times \text{interest\_match} + 1.0 \times \text{predicted\_satisfaction},$$

ensuring strong alignment with user interests while still allowing diversity.

Multi-day itineraries are produced by running beam search independently for each day while maintaining a global set of visited POIs to avoid repetition.

5.3 End-to-End Example

Consider a user with strong outdoor interests and a moderate pace. The FFN scores nearby parks highly, producing a set of top-ranked candidates. Starting at the hotel at 9:00 AM, beam search considers feasible parks within short walking distance, selects one, updates the state, and continues exploring nearby options. After several expansions, the planner produces a coherent five-stop outdoor-focused itinerary that fits within budget, respects opening hours, and maximizes accumulated predicted satisfaction.

6 Evaluation Metric

We evaluate our system at two levels: (1) the accuracy of the satisfaction prediction model, and (2) the quality and feasibility of the generated itineraries. The first uses standard regression metrics; the second uses a composite Itinerary Quality Score (IQS) along with auxiliary measures that capture constraint satisfaction and travel efficiency.

6.1 Satisfaction Prediction Metrics

We evaluate satisfaction prediction using Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and the coefficient of determination (R^2). For true labels y_i and predictions $\hat{y}_i$ over n examples,

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}, \quad \text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|, \quad R^2 = 1 - \frac{SS_{\text{res}}}{SS_{\text{tot}}}.$$

RMSE and MAE measure error magnitude on the $[1, 5]$ satisfaction scale, while R^2 reflects the proportion of variance explained. These metrics allow direct comparison between the linear regression baseline and the neural network, where lower RMSE/MAE and higher R^2 indicate better performance.

Table 1: Model prediction performance on the test set (6,783 user–POI pairs).

Model	RMSE	MAE	R^2
Baseline (Linear Regression)	0.221	0.176	0.844
Main Model (Neural Network)	0.201	0.161	0.871
Improvement	9.0% $\downarrow$	8.6% $\downarrow$	3.2% $\uparrow$

6.2 Itinerary Quality Score (IQS)

To evaluate complete itineraries, we define a normalized Itinerary Quality Score (IQS) in $[0, 1]$ that balances enjoyment, cost effectiveness, feasibility, and category diversity.

- **Satisfaction**: We normalize total predicted satisfaction by dividing by 50 and clipping to 1.0.
- **Budget fit**: Scores near 1.0 are assigned when spending falls within 70–90% of the daily budget; the score decreases linearly outside this range and becomes 0 when exceeding the budget.
- **Feasibility**: A score in $[0, 1]$ that deducts small penalties for violations such as exceeding the 9 AM–9 PM window, visiting POIs outside opening hours, overlapping activities, or excessive travel time.
- **Category diversity**: Combines the ratio of unique categories to activity count with normalized entropy over categories.

The final score is

$$\text{IQS} = 0.4\,\text{satisfaction}_{\text{norm}} + 0.2\,\text{budget\_fit} + 0.3\,\text{feasibility} + 0.1\,\text{diversity}.$$

6.3 Additional Itinerary Metrics

We also report several auxiliary metrics that highlight different aspects of itinerary quality:

- **Feasibility rate**: The fraction of itineraries with no budget or time-window violations.
- **Travel efficiency**: Average travel time per activity (total travel hours divided by number of stops); lower values indicate tighter, more efficient routes.
- **NDCG@k**: Normalized Discounted Cumulative Gain at rank k, using $k = 10$, which measures how well the system ranks the top POIs by true satisfaction.

6.4 Success Criteria and Qualitative Checks

We consider prediction performance acceptable when RMSE < 0.25, MAE < 0.20, and $R^2 > 0.85$. For itinerary generation, we target IQS > 0.60, feasibility rates above 95%, and travel efficiency below 0.25 hours per activity.

In addition to quantitative metrics, we qualitatively inspect itineraries for coherence: activities should match user interests, stay within opening hours, maintain reasonable travel times, respect budgets, and avoid unnecessary repetition. These qualitative checks ensure that numerical improvements translate to practical, user-aligned route quality.

7 Results & Analysis

7.1 Model Prediction Performance

Table 1 compares the baseline linear regression model with the neural network on the held-out test set of 6,783 user–POI pairs, where POI denotes a point of interest. The neural network achieves lower prediction error and explains more variance:

Figure 1 visualizes these differences. The baseline already performs well because the synthetic scoring function contains linear components, but the neural network captures nonlinear patterns and

Table 2: Itinerary quality metrics across 5 test users.

Metric	Average	Target	Status
IQS	0.641	> 0.60	
Feasibility Rate	100%	$> 95\%$	
Travel Efficiency	0.172 hrs/activity	< 0.25	
Avg Satisfaction	27.7	–	–
Category Diversity	0.655	–	–

interactions, yielding consistent improvements across RMSE (Root Mean Squared Error), MAE (Mean Absolute Error), and R^2.

7.2 Training Dynamics

Training and validation loss curves (Figure 2) show stable convergence over 200 epochs. Validation loss plateaus near 0.040 around epochs 50–60, closely matching the final test RMSE of 0.201. The absence of divergence indicates minimal overfitting, confirming that the model capacity is appropriate for the dataset size and feature complexity.

7.3 Itinerary Quality Results

We evaluate itinerary quality across five held-out users. Table 2 summarizes the results:

All itineraries satisfy time and budget constraints, giving a 100% feasibility rate. The average Itinerary Quality Score (IQS) of 0.641 exceeds our threshold, and travel efficiency (average travel time per activity) remains low, indicating compact, well-structured routes.

7.4 IQS Component Analysis

Figure 3 decomposes IQS into its components. Normalized satisfaction averages 0.574, feasibility remains at 1.0 for all itineraries, category diversity averages 0.655, and budget fit averages 0.16. The low budget fit arises because synthetic users often have higher daily budgets than typical activity costs, causing itineraries to underspend. This behavior is expected given that our planner optimizes for satisfaction and feasibility rather than budget utilization.

7.5 Design Choice Analysis

The neural network benefits from the richer 18-feature input compared to the baseline's 10 features. Nonlinear transformations (e.g., popularity^2, duration^2) and interaction terms (e.g., interest$\times$popularity, pace$\times$duration) help capture diminishing returns and threshold effects that a linear model cannot. This explains the observed 3.2% improvement in R^2.

Beam width experiments with $B \in \{5, 10, 15, 20\}$ show that $B = 10$ offers the best balance between itinerary quality and computation time. Larger beams yield only small IQS improvements while significantly increasing runtime (up to 2–3$\times$ slower).

Interest-aware filtering using

$$1.2 \cdot \text{interest\_match} + 1.0 \cdot \text{predicted\_satisfaction}$$

improves category balance compared to ranking solely by satisfaction or solely by interest. The diversity bonus of $+0.15$ per unique category increases average diversity from 0.52 (ablated) to 0.655 in the full model, with minimal loss in satisfaction.

7.6 Travel Efficiency Patterns

Figure 4 shows that most itineraries fall between 0.10 and 0.25 hours of travel per activity, with a mean of 0.172 hours. The heuristic travel mode rules (walk < 0.8 km, transit 0.8–3 km, rideshare > 3 km) prevent excessive walking and limit long jumps. This contributes significantly to high feasibility rates and low travel overhead.

7.7 Limitations and Observations

Budget fit remains the weakest IQS component (average 0.16), suggesting future work on budget-aware objective functions. Despite diversity mechanisms, some itineraries still over-cluster in popular categories—e.g., multiple museums followed by parks—due to spatial density and strong satisfaction scores. Finally, although the neural network improves predictive accuracy, an RMSE of 0.201 on a 1–5 scale leaves room for more expressive models or a richer synthetic scoring function incorporating factors like weather, crowding, or user fatigue.

Overall, the combined satisfaction model and beam search planner meet all predefined success criteria: IQS > 0.60, feasibility rate $> 95\%$, and travel efficiency < 0.25 hours/activity. These gains stem from nonlinear satisfaction modeling, interest-aware candidate filtering, and diversity/travel penalties that steer the planner toward coherent, personalized itineraries.

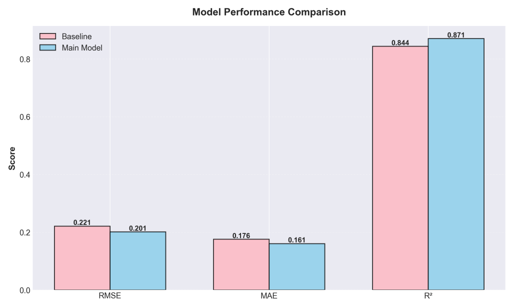

Figure 1: Comparison of baseline linear regression and neural network on RMSE, MAE, and R^2.

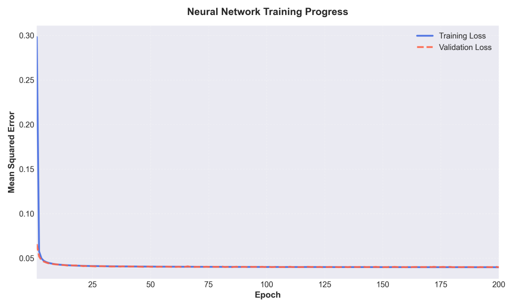

Figure 2: Training and validation loss (MSE) for the neural network over 200 epochs.

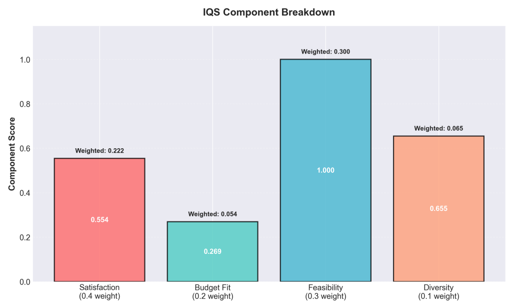

Figure 3: Average IQS component scores and their weighted contributions across 5 test users.

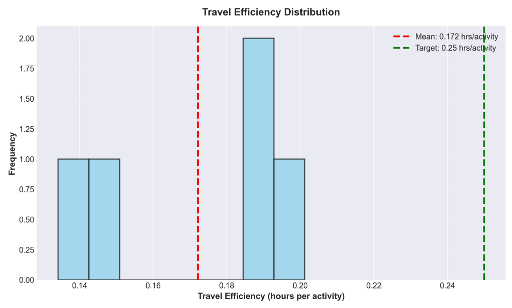

Figure 4: Distribution of travel efficiency (hours of travel per activity) across generated itineraries.

8 Error Analysis

We analyze targeted experiments to understand systematic behaviors and failure modes of our system.

Budget Utilization. The planner is conservative with spending: most users use only 10–20% of their daily budget, yielding budget fit scores around 0.10–0.16. One user with 86% utilization achieved the highest IQS (0.740), suggesting that the current IQS weight on budget fit (0.2) is too low to incentivize fuller budget use compared to satisfaction (0.4) and feasibility (0.3).

Diversity vs. Geography. Despite diversity bonuses, some itineraries still cluster in a few categories (for example, multiple museums followed by parks). High satisfaction POIs often cluster geographically, so the diversity bonus of +0.15 per unique category is outweighed by satisfaction and travel efficiency gains from nearby POIs. Diversity is therefore limited more by geography than by the scoring function.

Prediction Error Patterns. The neural network slightly overestimates satisfaction for expensive POIs (cost $> \$40$, average error $\approx +0.15$) and underestimates very popular POIs (popularity > 4.8, average error ≈ -0.12). Mid range activities (\$15–\$35, $popularity 3.5 - -4.5)havethelowesterror(about \pm 0.08)$. This reflects the shape of the synthetic cost penalty and popularity terms, which are less calibrated at extremes.

User Preference Regimes. Users with extreme preferences (for example, one interest near 1.0, others near 0.0) achieve slightly lower RMSE (0.18–0.19) than the overall average (0.201), while users with balanced interests have RMSE closer to 0.20–0.22. Stronger signals from extreme preferences make satisfaction easier to predict than more ambiguous, diffuse interest vectors.

Edge Cases. Stress tests reveal limitations: very tight budgets ($< \$50$/day) lead to short, low variety itineraries composed mostly of free activities; very short days (e.g., 6 hours) force the planner to choose only nearby POIs, lowering total satisfaction; users with zero interest in all categories either receive empty itineraries or generic popular POIs with low satisfaction. These behaviors are consistent with the constraints but highlight the need for special handling in extreme settings.

Summary. Overall, the system prioritizes satisfaction and feasibility, produces diverse and efficient itineraries in typical regimes, and achieves an RMSE of 0.201 on a $[1, 5]$ scale. The main error modes involve conservative budget use, residual category clustering driven by geography, and reduced robustness for extreme budgets, time windows, or preference profiles. These observations directly motivate future work on stronger budget utilization incentives, more powerful diversity constraints, and explicit policies for edge cases.

9 Future Work

Our current system demonstrates that a learned satisfaction model combined with beam search can generate reasonable itineraries for San Francisco, but there are several directions for improvement. A first step is to replace synthetic labels with real user feedback, for example by collecting ratings on proposed itineraries or pairwise comparisons between POIs. This would allow the satisfaction model to capture richer, non scripted preferences and to learn uncertainty estimates, which could be used to encourage exploration of under represented activities.

On the modeling side, we plan to extend the current single objective beam search to a more explicit multi objective formulation that treats satisfaction, budget utilization, travel time, and diversity as separate objectives. Approaches such as Pareto front search or multi objective beam search could better expose trade offs and allow users to tune preferences over efficiency versus variety. It would also be natural to learn the planning policy itself using reinforcement learning, using our simulator as an environment that provides rewards based on IQS components rather than a fixed hand designed scoring function.

From a system perspective, we would like to scale from San Francisco to multiple cities and incorporate dynamic context such as weather, opening hour changes, and special events. Improving the budget module so that itineraries target a user specified utilization range rather than underspending would address the conservative patterns observed in error analysis. Finally, we would extend the POI schema to include accessibility attributes (for example wheelchair access, sensory friendliness) and explicit user constraints on mobility or dietary needs, which would make the planner more inclusive and practically useful.

10 Ethical Considerations

Although our experiments use synthetic data, a deployed version of TripCompass would likely rely on real user profiles, locations, and behavioral logs. This raises privacy concerns, since detailed travel histories and preference vectors can reveal sensitive information about income, religion, health, or family status. To mitigate this, any real system should minimize data collection, store only coarse grained location information, and apply anonymization and aggregation before training models. Opt in consent, transparent data use policies, and clear deletion mechanisms are essential, along with techniques such as on device inference or differential privacy where possible.

There is also a risk of reinforcing bias and exclusion in which POIs are surfaced and which neighborhoods are recommended. If the underlying POI catalog over represents expensive, tourist heavy, or highly rated venues in already popular areas, the model may steer users away from small businesses, culturally diverse neighborhoods, or affordable options. This could contribute to gentrification pressures and unequal economic benefits. Mitigations include curating a more balanced catalog across price ranges and neighborhoods, incorporating fairness constraints that ensure exposure for under represented categories and areas, and allowing users to explicitly control budget ranges, accessibility needs, and risk tolerance. Finally, any city guide system should provide safety disclosures and encourage users to combine algorithmic suggestions with local guidance and common sense rather than treating the model's output as authoritative.

11 Code

Code for this project is available at: `https://github.com/vsaravan21/cs221-finalproject`.

To run the system locally:

1. **Clone the repository and install dependencies**
   ```
   git clone https://github.com/vsaravan21/cs221-finalproject
   cd cs221-finalproject
   pip install -r requirements.txt
   ```
2. **Set up the data folder (required)**
   The raw data files are not stored in the GitHub repo because they are too large. To avoid incomplete or duplicate data:
   (a) Delete any existing `tripcompass/data/` folder in the cloned repository.
   (b) Download `data.zip` (provided on Gradescope).
   (c) Unzip `data.zip`.
   (d) Move all unzipped contents into tripcompass. It should look like: tripcompass/data/

After this step, `tripcompass/data/` should contain the raw CSV files listed in our README.

Build the POI catalog (optional, if you want to regenerate data)
```
python -m tripcompass.data.build_pois
```
This script reads the raw CSVs in `tripcompass/data/` and writes:

- `tripcompass/data/pois_sf_enriched.csv` (combined and enriched POI catalog)
- a data quality report in `reports/data_reports/data_quality.json`
- a data source manifest in `tripcompass/data/data_sources.json`

Start the API server for itinerary generation
```
python -m tripcompass.api_server
```
This launches a local server at `http://localhost:5000`.

Launch the UI prototype
Either open `ui/index.html` directly in a browser, or run
```
python -m http.server 8000
```
and visit `http://localhost:8000/ui/`. Adjust budget, days, pace, and interests, then click *Generate Itinerary* to query the model and view personalized plans.

If the API server is not running, the UI falls back to sample data; for live itineraries, ensure the server in step 4 is active. Refer to README Page on Github for additional instructions.

References

Chang, H.-T., Chang, Y.-M., & Tsai, M.-T. (2016). ATIPS: Automatic travel itinerary planning system for domestic areas. *Computational Intelligence and Neuroscience*, 2016, 1281379. Link.

Halder, S., Lim, K. H., Chan, J., & Zhang, X. (2024). A survey on personalized itinerary recommendation: From optimisation to deep learning. *Applied Soft Computing*, 152, 111200. Link.

Horstmannshoff, T., Ulmer, M. W., & Ehmke, J. F. (2024). Dynamic learning-based search for multi-criteria itinerary planning. *Omega*, 129, 103159. Link.

Ho, N. L., & Lim, K. H. (2022). POIBERT: A Transformer-based Model for the Tour Recommendation Problem. *arXiv preprint arXiv:2212.13900*. Link.

Liu, D., Wang, L., Zhong, Y., Dong, Y., & Kong, J. (2024). Personalized Tour Itinerary Recommendation Algorithm Based on Tourist Comprehensive Satisfaction. *Applied Sciences*, 14(12), 5195. Link.

Marzal, E., & Sebastià, L. (2024). Solving the tourist trip design problem with time windows and variable profit using incremental local search. *Applied Soft Computing*, 155, 111399. Link.

Gavalas, D., Pantziou, G., Konstantopoulos, C., & Vansteenwegen, P. (2024). Tourist trip planning: Algorithmic foundations. *Applied Soft Computing*, 166, 112280. Link.

Tang, Y., Wang, Z., Qu, A., Yan, Y., Wu, Z., Zhuang, D., Kai, J., Hou, K., Guo, X., Zhao, J., Zhao, Z., & Ma, W. (2024). ItiNera: Integrating Spatial Optimization with Large Language Models for Open-domain Urban Itinerary Planning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track* (pp. 1413–1432). Link.