Advanced Analytics and Applications

Project Report

Authors:

Christopher Göbeler (Student ID: 7376835)

Supervisor: Univ.-Prof. Dr. Wolfgang Ketter

Co-Supervisor: Janik Muires

Department of Information Systems for Sustainable Society

Faculty of Management, Economics and Social Sciences

University of Cologne

23.07.2025

Eidesstattliche Versicherung

Hiermit versichere ich an Eides statt, dass ich die vorliegende Arbeit selbstständig und ohne die Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten und nicht veröffentlichten Schriften entnommen wurden, sind als solche kenntlich gemacht. Die Arbeit ist in gleicher oder ähnlicher Form oder auszugsweise im Rahmen einer anderen Prüfung noch nicht vorgelegt worden. Ich versichere, dass die eingereichte elektronische Fassung der eingereichten Druckfassung vollständig entspricht. Die Strafbarkeit

einer falschen eidesstattlichen Versicherung ist mir bekannt, namentlich die Strafandrohung gemäß § 156 StGB bis zu drei Jahren Freiheitsstrafe oder Geldstrafe bei vorsätzlicher Begehung der Tat bzw. gemäß § 161 Abs. 1 StGB bis zu einem Jahr Freiheitsstrafe oder Geldstrafe bei fahrlässiger Begehung.

Christopher Göbeler

Köln, den 23.07.2025

Abstract

This report presents an integrated analytical approach for optimizing the deployment and operation of an autonomous electric ride-hailing service in urban environments, specifically using Chicago as a case study. Leveraging a comprehensive dataset of approximately nine million taxi trips from January 2024 to June 2025, supplemented by external sources such as weather data, points of interest, and public holidays, the study conducts an in-depth exploration of spatiotemporal taxi demand patterns.

The predictive modeling component evaluates and compares Support Vector Machines (SVMs) and Neural Networks (NNs) at multiple temporal and spatial resolutions. Findings indicate that NNs significantly outperform SVMs, particularly at higher granularities, due to better computational efficiency and ability to handle spatial embeddings.

Additionally, the report addresses the challenge of smart charging for an electrified autonomous taxi fleet using reinforcement learning (RL). Various RL algorithms including SARSA, Q-Learning, Double Q-Learning, and Deep Q-Network (DQN) are implemented and evaluated. The results clearly demonstrate the superior performance of the DQN algorithm, effectively balancing the costs associated with charging against the risk of insufficient battery capacity at departure.

Contents

List of Figures

List of Tables

This report presents the modeling and analysis conducted as part of the Advanced Analytics and Applications Group Project, with full source code and data processing workflows available at this GitHub repository. All Code as well as the following sections in this Report were created by Christopher Göbeler.

1 Introduction

The rapid evolution of smart mobility solutions is fundamentally transforming urban transportation landscapes worldwide. Major cities are witnessing the rise of innovative ride-hailing platforms fueled by advancements in autonomous driving, electrification, and data-driven decision making.

As the Data Science Team at Bane & Wayne Partners (BWP), we aim to provide actionable insights and solutions to support our client's strategic objective of launching a fully autonomous ride-hailing platform in the United States. Our primary mission is to analyze historical urban mobility data from Chicago using taxi trip data as a proxy for potential ride-hailing operations. Combining this historical data with external factors, such as weather and point-of-interest information, allows us to gather comprehensive understanding of demand patterns and operational requirements. The scope of this report encompasses an end-to-end analysis approach: from collecting and preparing relevant datasets, performing spatial and temporal analytics, developing predictive models with the use of support vector machines and neural networks, to designing intelligent charging strategies for electric vehicles using reinforcement learning techniques. Through this multidisciplinary approach, our aim is to deliver data-driven recommendations that will enable our client to successfully navigate the challenges and opportunities of the US ride-hailing market.

2 Data Description

The principal dataset employed in this study is the publicly available Chicago Taxi Trips dataset, encompassing roughly nine million taxi trips within the city of Chicago from January 2024 until today. We focus on data from 2024 through June 2025 in our analysis.

Each observation in the dataset corresponds to a single taxi journey and is characterized by a range of attributes that provide both temporal and spatial context. Key variables include the trip start and end timestamps, which allow for precise temporal analysis at an hourly or even finer resolution. Trip duration and distance are recorded for each ride, offering further insights into travel behaviors and demand profiles. Spatial information is available through the identification of both pickup and dropoff locations, which are encoded at the census tract and community area level. While this level of spatial granularity is somewhat coarser than exact geographic coordinates, it preserves rider privacy while still being sufficient for city-wide mobility analysis. Financial attributes such as fare amount, tips, tolls, and surcharges are documented for each trip, enabling the assessment of economic aspects and demand behavior of trips and taxi operations. While the sheer size of the dataset allows for robust statistical and machine learning analyses, there are important limitations to consider,mainly in the realm of spatial resolution. The restriction to census tracts necessitates further discretization and disaggregation, when more granular spatial analysis is required. In summary, the Chicago Taxi Trips dataset for 2024–2025 provides a comprehensive and temporally resolved record of taxi activity in one of the United States' largest cities. Its breadth and structure make it highly suitable for examining spatio-temporal demand patterns to develop predictive models relevant to optimally consult on the operational problems of an autonomous, fully electrified ride-hailing fleet.

2.1 External Data Collection

In addition to the primary taxi trip dataset, a range of external data sources were integrated to enrich the analytical framework and enhance the predictive power in the later modelling stages. Recognizing that urban mobility is influenced by a variety of contextual factors, the study incorporated high-resolution weather data, point-of-interest (POI) information, and public holiday calendars.

Historical weather data for the city of Chicago was obtained via the *Weatherbit API*. This data source offers comprehensive meteorological observations, including variables such as temperature, precipitation, wind speed, and humidity at hourly intervals. The inclusion of weather features enables the analysis to account for exogenous effects on travel demand, which are particularly relevant in a city known for its highly variable weather conditions.

To further contextualize spatial patterns in taxi demand, point-of-interest data was extracted from *OpenStreetMap* (OSM). The OSM dataset provides detailed geographic information regarding the location and categorization of various urban amenities, such as restaurants, bars and tourist attractions. POI data can be a significant determinant of urban mobility flows, as it directly affects the spatial distribution of trip origins and destinations.

Additionally, data on public holidays and special events was incorporated using a Python package capable of generating holiday calendars for the United States and the city of Chicago specifically. The presence of holidays and major events is known to induce marked deviations in regular mobility patterns and thus is essential for constructing robust predictive models.

The integration of these external data sources allows for a more comprehensive and nuanced modeling approach, facilitating the identification of key drivers of taxi demand and supporting the development of operational strategies tailored to varying exogenous conditions. By combining these data streams with the primary taxi trip records, the study achieves a higher degree of explanatory and predictive accuracy in the context of urban ride-hailing services.

3 Descriptive Analysis

Before diving deeper into more complex analyses it is important to consider some baseline statistics about ride-hailing services and demand behavior. The scope of these statistics is reduced to rides where all information is present and does not include the full dataset of January 2024 to June 2025 in all cases. The most frequently observed pickup tract is associated with Chicago's main airport, while the downtown area emerges as the predominant destination for drop-offs. Notably, the single most common routing pattern is characterized by both the origin and destination being the airport, likely representing short rides to hotels closeby. Having a major transportation hub as the central node within a metropolitan network is not surprising and underscores the importance of incorporating locational data into our analysis.

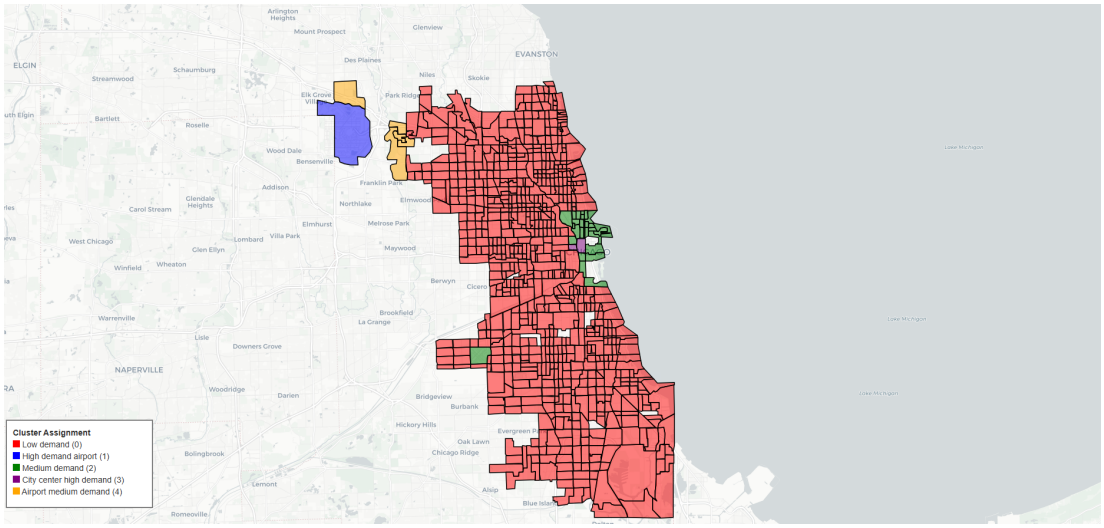

Figure 1: Chicago Census Tracts - Demand Clusters

A temporal disaggregation of revenue per trip demonstrates pronounced monthly and hourly fluctuations. The average revenue per trip ranged from approximately \$26.26 to \$38.02 across the observed period. Monthly averages remain relatively stable, with a modest upward trend in the summer months of 2024. Hourly revenue analysis reveals a clear diurnal cycle, with per-trip revenues peaking during the late night and early morning hours (midnight to 6 AM), where values consistently exceed \$30. The highest average revenue per trip occurs at 5 AM (\$36.10), whereas the lowest is observed at 8 AM (\$23.14). Totals drop during rush hours and rise in later windows likely due to surcharges at those hours.

The cumulative gross revenue generated during the period of analysis is \$274,704,462.68.

This emphasizes the potential for establishing a dominant autonomous ride-hailing fleet, even when considering just a single city. Examination of individual trips highlights the presence of outliers, most notably a single trip with a recorded revenue of $9,999.75.

From the perspective of vehicle-level activity, the taxi with the highest total gross revenue amassed $823,921.39 during the observation period, indicating considerable heterogeneity in fleet utilization. Additionally, the most active taxi, identified by trip frequency, completed 15,277 rides, averaging 30.31 rides per day and generating an average gross revenue of $778.41 per day. For this vehicle, the average trip covered 9.44 miles and lasted approximately 19.26 minutes, suggesting operational efficiency in balancing trip length and duration.

The mean price per mile, aggregated over all trips, is $17.54, reflecting prevailing fare structures and the influence of both fixed and variable charges on ride costs.

Overall, these descriptive findings provide a nuanced understanding not just of spatial and temporal, but also financial dimensions of urban taxi operations in Chicago.

4 Modelling Taxi Demand

To effectively capture the complex dynamics of urban mobility, it is essential to model trip demand at multiple spatial and temporal scales. In this study, predictive models are developed to estimate taxi trip demand at hourly, bi-hourly (2-hour), 6-hourly, and 12-hourly intervals. This temporal stratification is combined with two levels of spatial aggregation: the census tract and the broader community area. Spatial information for each unit is encoded using the centroid coordinates derived from the H3 geospatial indexing system, thereby preserving relative spatial relationships and facilitating effective spatial learning within the models. Since the original Chicago taxi trips dataset supplies locational data only for the pickup and dropoff centroids, there are limited options to go for even finer spatial resolutions. While we decided to disaggregate demand of community areas into census tracts for rides where the more granular spatial information is missing, we did not choose this approach of disaggregation to get an even more granular resolution using h3 hexagons.

Temporal features, including the hour of day and the month, are transformed

using sine and cosine encodings . This ensures a continuous, cyclical representation of time, allowing the models to recognize that, for example, the distance between 11 PM and 1 AM is equivalent to that between 1 AM and 3 AM (see Figure 2). By leveraging these cyclic encodings alongside spatial centroids, the input feature set is structured to reflect the inherent periodicity and adjacency present in recurring daily demand patterns. This methodological framework supports a robust examination of how varying spatio-temporal granularities impact predictive performance and operational insights.

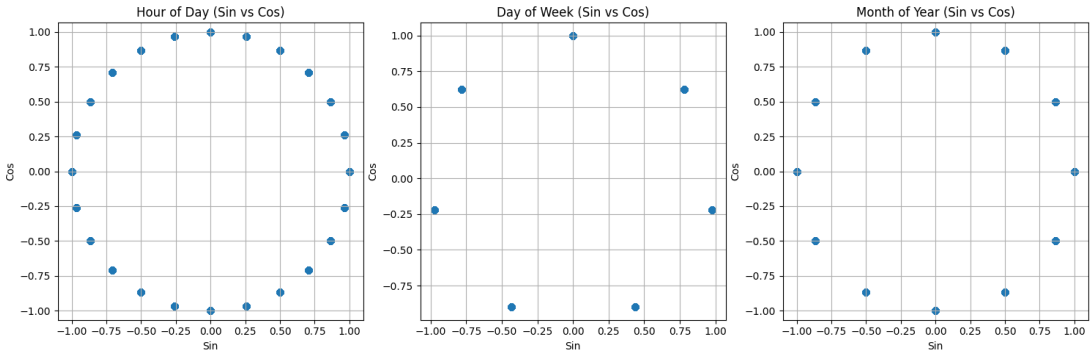

Figure 2: Sine/Cosine Transformation of Temporal Variables

4.1 Support Vector Machine

Initial experiments using traditional Support Vector Machines (SVMs) with exact kernel computations proved infeasible for the large-scale taxi demand datasets, particularly those with fine-grained spatial resolution. The primary challenges stemmed from the large number of observations and the high-dimensional feature space, especially due to one-hot encoding of H3 spatial indices. These factors made full kernel matrix computations prohibitively expensive in both memory usage and training time.

To address this, the modeling approach was redesigned using approximate kernel methods, which enable the use of linear models in transformed feature spaces that emulate the behavior of non-linear kernels. Two kernel types were explored for each dataset: a linear SVM using `LinearSVR` and an RBF kernel approximation using randomized Fourier features via `RBFSampler`. These approximations were combined with stochastic gradient descent regression models to support scalability over large datasets.

This kernel approximation approach allowed all models to be successfully trained across the full set of community area datasets with hourly, 2-hourly, 6-hourly, and 12-

hourly resolutions and limited resource requirements.

4.2 Neural Network

As a further measure to model taxi trip demand under varying spatial and temporal resolutions, a series of feedforward neural networks (NNs) were implemented in PyTorch. These networks are designed to learn complex nonlinear relationships between spatiotemporal features and demand levels, and are evaluated across datasets using different granularities.

Each neural network follows a fully connected feedforward design, with architecture depth determined by a selected complexity level:

Simple: One hidden layer with 32 neurons.

Medium: Two hidden layers with 64 and 32 neurons.

Deep: Three hidden layers with 128, 64, and 32 neurons.

The architecture integrates both numerical and categorical inputs. The categorical spatial feature, `pickup_h3_9`, is encoded using a trainable embedding layer: H3 indices are first integer-encoded and then embedded using a dedicated `nn.Embedding` layer. Embedding dimensions increase with model complexity: 8 (simple), 16 (medium), and 32 (deep). The embedded vectors are concatenated with normalized numerical features before being passed through the network.

ReLU activation is applied after each linear transformation. The final output layer uses a single linear unit to predict continuous taxi trip counts. Each model was trained for 10 epochs using the Adam optimizer with a learning rate of 1×10^{-3}. Mean squared error (MSE) was used as the training loss, more on this choice follows in the chapter on evaluation. Data was loaded in batches using PyTorch `DataLoader` using a batch size of 256.

4.3 Model Evaluation & Recommendation

Model performance was evaluated using several standard regression metrics defined as follows:

Mean Squared Error (MSE):

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Mean Absolute Error (MAE):

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

Mean Absolute Percentage Error (MAPE):

$$\text{MAPE} = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i + \varepsilon}\right|$$

where ε is a small constant added to avoid division by zero.

R-squared (R^2):

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

where $\bar{y}$ is the mean of the observed values.

For the main training metric, the MSE was chosen due to the operational benefits resulting from optimizing this loss: When strategically deploying taxis, it is much more important to capture high demand hotspots accurately than to estimate lower absolute demand with high accuracy. Since the MSE penalizes large deviations more than smaller ones, the algorithm focuses on identifying peaks with higher accuracy.

4.3.1 SVM Performance

Table 1 presents the updated results comparing linear and RBF kernel approximations across all area-level datasets with different temporal resolutions. Across all settings, the linear SVM models consistently outperform their RBF counterparts in terms of predictive accuracy.

For each dataset, the linear model achieves lower mean squared error (MSE) and mean absolute error (MAE), as well as higher R^2 values. The performance gap is especially pronounced in the coarser temporal resolutions (e.g., 6-hourly and 12-hourly), where the RBF models underperform with R^2 values of 0.257 and 0.322, respectively, compared to 0.626 and 0.603 for the linear models. Even for the finest resolution dataset (`area_hourly_h3`), the linear SVM outperforms the RBF approximation by a wide margin: reducing MSE from 1616.57 to 934.53 and improving R^2 from

Table 1: Updated SVM Performance on Area-Level Datasets (Linear vs RBF)

Dataset	**Kernel**	**MSE**	**MAE**	R^2
area_hourly_h3	linear	934.53	9.59	0.580
area_hourly_h3	rbf	1616.57	21.58	0.274
area_2hourly_h3	linear	3051.23	15.81	0.593
area_2hourly_h3	rbf	5495.25	37.98	0.266
area_6hourly_h3	linear	18815.52	34.80	0.626
area_6hourly_h3	rbf	37366.31	97.66	0.257
area_12hourly_h3	linear	60541.52	48.41	0.603
area_12hourly_h3	rbf	103473.34	170.77	0.322

0.274 to 0.580.

It is generally uncommon for a linear kernel to outperform a RBF kernel, as it maps input features into an infinite-dimensional space and is capable of capturing complex, non-linear relationships between features and targets. In theory, the RBF kernel strictly generalizes the linear kernel, meaning it should perform at least as well or better on most datasets, particularly when the underlying data-generating process is non-linear. However, in practice, this theoretical advantage comes at the cost of increased risk of overfitting.

4.3.2 NN Performance

Table 2 summarizes the performance of neural networks of varying complexity across different spatial and temporal aggregation levels. Overall, deeper architectures tend to yield better predictive accuracy for fine-grained datasets, such as `census_hourly_h3` and `census_2hourly_h3`, as evidenced by lower MSE and MAE values and higher R^2 scores. However, the performance gains from deeper networks diminish or even reverse when applied to coarser aggregations.

Table 2: Neural Network Performance across Spatiotemporal Resolutions

Dataset	**Model**	**MSE**	**MAE**	R^2
census_hourly_h3	Simple	4.8943	0.4755	0.8664
census_hourly_h3	Medium	3.3206	0.3142	0.9094
census_hourly_h3	Deep	3.0928	0.3043	0.9156
census_2hourly_h3	Simple	18.9249	0.9318	0.8675
census_2hourly_h3	Medium	13.4256	0.6072	0.9060
census_2hourly_h3	Deep	11.3461	0.5300	0.9205
census_6hourly_h3	Simple	143.0646	2.6273	0.8769
census_6hourly_h3	Medium	100.6256	1.9896	0.9134
census_6hourly_h3	Deep	98.9006	1.3829	0.9149
census_12hourly_h3	Simple	404.3251	4.3624	0.8994
census_12hourly_h3	Medium	321.6382	3.2960	0.9200
census_12hourly_h3	Deep	372.0903	3.0307	0.9075
area_hourly_h3	Simple	234.0589	5.6079	0.8948
area_hourly_h3	Medium	164.9225	4.2772	0.9259
area_hourly_h3	Deep	140.4819	4.2270	0.9369
area_2hourly_h3	Simple	875.1282	9.5230	0.8832
area_2hourly_h3	Medium	551.8194	7.1618	0.9263
area_2hourly_h3	Deep	494.8300	6.8193	0.9339
area_6hourly_h3	Simple	6533.6094	28.5183	0.8701
area_6hourly_h3	Medium	3144.5334	16.7177	0.9375
area_6hourly_h3	Deep	2967.6165	16.3672	0.9410
area_12hourly_h3	Simple	26606.4414	63.3133	0.8256
area_12hourly_h3	Medium	11619.5342	32.5474	0.9239
area_12hourly_h3	Deep	8396.6553	25.2334	0.9450

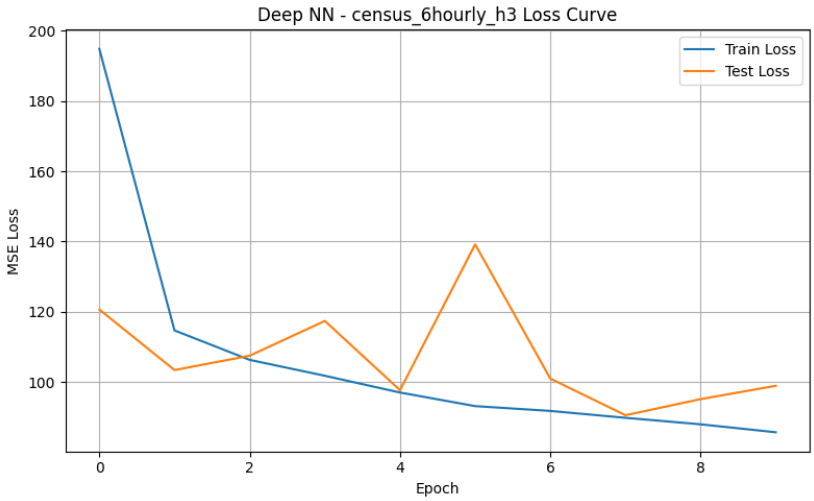

Figure 3: Training and Validation Loss - Dataset agg. to 6hr on Census Tract Level

These findings highlight the importance of aligning model complexity with data granularity: while deeper models can capture more abstract relationships, they are also more prone to overfitting when data volume or variability is limited. Notably, for datasets with higher temporal and spatial aggregation (e.g., `area_6hourly_h3` and `area_12hourly_h3`), the deeper neural networks exhibit signs of overfitting (Figure 3). This is reflected in converging or even increasing test losses over repeated training epochs, despite continued reductions in training loss. The reduced number of training examples in these aggregated datasets exacerbates this issue, making the added model complexity counterproductive.

Conversely, for high-resolution datasets with richer and more varied input patterns, even relatively simple neural networks perform competitively. Simple or medium-complexity model achieves a favorable balance between performance and generalization, suggesting that adding more layers or parameters yields only marginal benefits for this prediction task even at finer resolutions.

4.3.3 Conclusion

The results clearly demonstrate that Support Vector Machines (SVMs) are computationally intensive and do not scale effectively to large datasets, particularly those with high spatiotemporal resolution. As dataset size and dimensionality increase, the training time and memory requirements of SVMs grow substantially, limiting their practical applicability. In contrast, even relatively simple neural networks not only scale more efficiently but also outperform SVMs in predictive accuracy across the evaluated tasks. Neural networks are inherently more adaptable to granular spatial data, as demonstrated by their ability to incorporate spatial embeddings (e.g., H3 hexagon indices) in a compact and informative manner. This allows them to capture fine-grained spatial patterns more effectively than traditional kernel-based methods. However, care must be taken when designing deeper or more complex architectures, as overly expressive neural networks can overfit quickly—especially in settings with limited training data or high variability. Thus, model complexity should be matched carefully to the data characteristics to ensure robust generalization.

5 Smart Charging - RL

A critical operational challenge for electrified taxi fleets lies in optimizing charging strategies to minimize costs while ensuring vehicles have sufficient energy to meet stochastic daily demand. To address this, we conceptualize the smart charging decision as a sequential optimization problem, modeled using reinforcement learning (RL). In particular, we design a simulated environment in which automated agents learn to control the charging process for an electric taxi, subject to operational and physical constraints.

5.1 Problem Description and Environment

In the considered scenario, the taxi driver returns home at 14:00 and departs at 16:00 each day, thereby providing a two-hour charging window, discretized into eight 15-minute intervals. At each interval, the agent selects one of five charging power levels (discrete intervals going from 0 to 22kWh) with and overall battery capacity of 50 kWh. The objective is to achieve a final battery level that meets or exceeds a stochastic energy demand—drawn from a normal distribution (mean: 30 kWh, standard devi-

ation: 5 kWh) at the time of departure. Running out of energy incurs a significant penalty, enforcing the critical operational requirement of reliability.

The environment implemented defines the Markov Decision Process (MDP) as follows:

Action space: Actions correspond to discrete charging power choices, mapped linearly between zero and the maximum rate. **Transition dynamics**: At each step, the chosen power level determines the incremental increase in stored energy, subject to the battery's maximum capacity. **Reward function**: The immediate reward (negative cost) at each step is proportional to the (negative exponential) cost of charging power, with cost coefficients varying over time to represent time-of-use pricing or other dynamic factors. At the final step, a high penalty is incurred if the accumulated energy is insufficient to meet the randomly sampled demand. **Episode termination**: An episode ends when the charging window closes (after eight intervals), at which point the agent's performance is evaluated based on cost minimization and avoidance of energy shortfalls.

5.2 Learning Algorithms

5.2.1 SARSA

To address the intelligent charging control problem in the proposed environment, we implemented a **SARSA (State-Action-Reward-State-Action)** agent. SARSA is an on-policy, model-free reinforcement learning algorithm that iteratively updates action-value estimates based on the current policy. Unlike off-policy methods such as Q-learning, SARSA directly incorporates the agent's exploratory actions into the learning updates, which can be advantageous in environments where safety or risk aversion is important—such as in electric vehicle charging with strict operational constraints.

The SARSA agent discretizes the continuous battery energy state into a finite number of bins, and the time dimension is divided into steps corresponding to each 15-minute interval in the charging window. The state-action value function $Q(s, a)$ is maintained in a three-dimensional array indexed by time step, energy bin, and action (charging power level). The agent selects actions using an ϵ-greedy strategy to balance exploration and exploitation and updates the Q table according to the SARSA update rule:

$$Q(s, a) \leftarrow Q(s, a) + \alpha \left[r + \gamma Q(s', a') - Q(s, a) \right] \tag{1}$$

where α is the learning rate, γ is the discount factor, r is the received reward, and (s', a') are the next state and action.

The agent was trained for 250,000 episodes, leading to a success rate of 94.87%, defined as episodes where the agent successfully accumulated enough battery energy to meet the (randomly drawn) demand at departure. The average reward per episode was -255.94, reflecting the trade-off between energy cost and penalties. A penalty was incurred in approximately 51.31% of episodes, indicating the presence of some risk in the learned policy—primarily in cases of demand outliers or insufficient charging in early exploration phases.

An analysis of the charging strategy in a representative episode shows the agent frequently chooses high charging rates (e.g., 22 kW, the maximum allowed), interspersed with medium or low rates when intermediate energy levels suffice. This policy reflects a risk-averse behavior, aiming to minimize the high penalty associated with undercharging while still accounting for the time-varying cost structure.

Visualizations of the value function and policy map (see Figure 4) further illustrate the agent's learned behavior: higher power actions are favored at low battery states or late time steps, while lower power may be selected when sufficient energy has been accumulated. The value function demonstrates a monotonic increase with both remaining time and stored energy, consistent with the agent's objective.

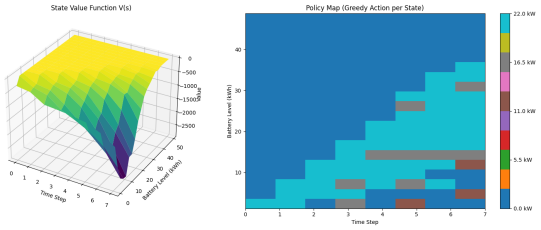

Figure 4: SARSA Value Function & Policy Map

5.2.2 Q-Learning

In addition to the SARSA agent, we implemented a Q-learning agent to solve the intelligent charging task. Q-learning is a widely used off-policy, model-free reinforcement learning algorithm. Unlike SARSA, which learns from the actions actually taken, Q-learning estimates the optimal action-value function independently of the agent's policy by always considering the action with the highest value in the next state.

This property often leads to more aggressive exploitation of learned knowledge and, potentially, faster convergence to the optimal policy.

The Q-learning agent discretizes both the battery energy state and the charging window time into a finite grid, identical to the SARSA setup. The state-action value function $Q(s, a)$ is represented as a three-dimensional table indexed by time step, energy bin, and action (charging power level). Action selection is performed using an ϵ-greedy strategy, where the probability of random exploration (ϵ) is decayed over time to encourage convergence.

The update rule for Q-learning is given by:

$$Q(s, a) \leftarrow Q(s, a) + \alpha \left[r + \gamma \max_{a'} Q(s', a') - Q(s, a) \right] \tag{2}$$

where α is the learning rate, γ is the discount factor, r is the received reward, and $\max_{a'} Q(s', a')$ is the maximum estimated value for the next state across all possible actions.

After training the Q-learning agent for 250,000 episodes, we observed a success rate of 97.13%, indicating that the agent reliably met or exceeded the battery energy required to satisfy stochastic demand at departure. The average reward per episode was -236.22, showing a notable improvement over the SARSA agent in both overall reward and penalty minimization. Penalties were incurred in only 0.03% of episodes, demonstrating the agent's ability to avoid costly undercharging.

In representative episodes, the learned policy exhibited a preference for maximum or near-maximum charging rates in the early steps, similar to the SARSA policy, but demonstrated improved reliability in maintaining sufficient battery charge. The reduced penalty frequency and higher success rate can be attributed to the off-policy nature of Q-learning, which favors optimal decision-making under uncertainty.

Visualization of the value function and the greedy policy map (add graph) confirms that the agent has learned to prioritize aggressive charging when the battery is low or time is limited, while modulating power in cases where sufficient energy has already been accumulated.

These results suggest that Q-learning is highly effective for the intelligent charging control problem under the simulated conditions, providing a strong baseline for future comparison with more advanced or function-approximation-based reinforcement

learning algorithms.

5.2.3 Double Q-Learning

To further address the potential for overestimation bias in value estimates, we implemented a Double Q-learning agent for the smart charging environment. Double Q-learning is an extension of standard Q-learning designed to reduce overoptimistic value estimations by decoupling the selection and evaluation of actions. This is achieved by maintaining two separate Q-tables, Q_1 and Q_2, which are updated in alternating fashion.

The Double Q-learning agent utilizes the same discretization for state (time and battery energy) and action (charging power) as the previous agents. For each learning step, one of the two Q-tables is chosen at random to update. The target for each update is computed using the alternate Q-table, thereby mitigating the upward bias associated with max-based value updates. Formally, the update rules are as follows:

$$Q_1(s,a) \leftarrow Q_1(s,a) + \alpha \left[r + \gamma Q_2(s', \arg\max_{a'} Q_1(s',a')) - Q_1(s,a) \right] \tag{3}$$

$$Q_2(s,a) \leftarrow Q_2(s,a) + \alpha \left[r + \gamma Q_1(s', \arg\max_{a'} Q_2(s',a')) - Q_2(s,a) \right] \tag{4}$$

where α is the learning rate, γ the discount factor, r the received reward, and s', a' denote the next state and action.

Action selection is performed by taking the greedy action with respect to the sum $Q_1 + Q_2$, and exploration is managed using an ϵ-greedy strategy with decaying ϵ.

The Double Q-learning agent was trained for 250,000 episodes. The resulting success rate was 84.00%, with an average reward per episode of -353.68. Penalties, corresponding to failure to meet the energy demand at departure, occurred in 0.16% of episodes. While the penalty frequency remains low, both the success rate and the average reward were less favorable than for the standard Q-learning and SARSA agents in this environment.

Analysis of the learned charging strategy indicates that, as with other agents, the Double Q-learning agent often selects high or maximum charging power in early and mid-charging intervals, but exhibits greater variability in some episodes, potentially reflecting the increased stochasticity in learning updates. The reduction in overestima-

tion bias does not always translate to improved performance in all settings, especially when the state-action space is relatively small.

Visualizations of the combined value function and policy (not shown here) support these findings: the learned policy remains risk-averse, prioritizing higher charging rates when energy reserves are low, but the value surface demonstrates slightly more conservative estimates compared to single Q-learning.

Overall, Double Q-learning offers theoretical advantages in mitigating overestimation but, in this application, did not outperform standard Q-learning. The results nonetheless provide a robust benchmark and additional insight into the stability and efficiency of value-based reinforcement learning algorithms in smart charging applications.

5.3 Evaluation

The smart charging environment simulates an electric taxi that returns home at 14:00 and must be sufficiently charged before departure at 16:00. This gives the agent eight discrete 15-minute time steps to allocate charging power, with the objective of minimizing electricity costs while ensuring that the final energy level meets or exceeds a stochastic energy demand, modeled as $\mathcal{N}(30, 5^2)$ kWh. Charging costs vary over time through an increasing cost coefficient α_t, encouraging the agent to exploit cheaper early time slots and avoid unnecessary charging in later, more expensive intervals. A heavy penalty is applied if the vehicle departs undercharged, making successful energy fulfillment critical.

Table 3: Performance Metrics of RL Agents in the Smart Charging Task

Agent	**Success Rate (%)**	**Avg Reward**	**Avg Penalty**
SARSA	94.93	-255.43	0.51
Q-Learning	97.13	-236.22	0.03
Double Q-Learning	84.00	-353.68	0.16
DQN	97.68	-232.31	0.02

In this context, the Deep Q-Network (DQN) agent demonstrates the best overall performance, achieving a 97.68% success rate and the most favorable average reward of -232.31. Its final action pattern reveals a cost-aware strategy: full charging is prioritized early, with reduced charging in later steps, suggesting the agent has learned to

efficiently accumulate enough energy before costs peak (see Table 3). The Q-Learning agent performs similarly well, reaching a 97.13% success rate and nearly eliminating penalties. However, its average reward is slightly worse than that of DQN, indicating less optimal cost management despite reliable fulfillment of energy demand.

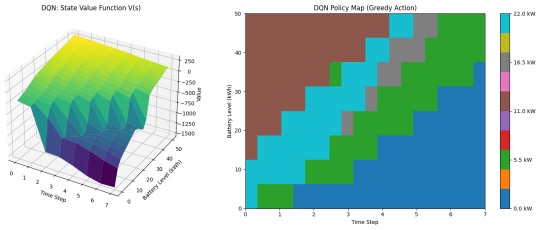

Figure 5: DQN Value Function & Policy Map

The SARSA agent, while attaining a respectable 94.93% success rate, incurs a significantly worse average reward (-255.43) due to a consistently high penalty rate. Its final episode behavior—seven full-power charging steps followed by no charging—suggests a rigid policy that lacks adaptability to varying energy demands and cost dynamics. This may stem from its on-policy learning, which tends to be more conservative and less exploratory. The Double Q-Learning agent performs the worst, with a success rate of only 84.00% and the lowest average reward. While Double Q-Learning is designed to mitigate overestimation bias in value functions, it appears to have struggled in this environment, potentially due to poor convergence or suboptimal exploration in the short time horizon and high-stakes penalty structure.

Table 4: Final Episode Charging Patterns per Algorithm (Power in kW)

Agent	**Final Charging Pattern**
SARSA	[22.00, 22.00, 22.00, 22.00, 22.00, 22.00, 22.00, 0.00]
Q-Learning	[22.00, 22.00, 16.50, 22.00, 22.00, 16.50, 22.00, 22.00]
Double Q-Learning	[22.00, 16.50, 22.00, 22.00, 22.00, 22.00, 16.50, 22.00]
DQN	[22.00, 22.00, 22.00, 22.00, 22.00, 22.00, 16.50, 16.50]

Taken together, the results emphasize that high-performing agents in this task must effectively balance cost-efficient charging with the uncertainty of final energy demand. Overcharging early ensures robustness but may increase costs, while undercharging invites severe penalties. The DQN agent's ability to generalize across these trade-offs underscores the advantage of deep function approximation and experience replay in learning precise temporal strategies for energy management.

6 Discussion & Outlook

This project demonstrates the significant impact that the choice of predictive modeling techniques and spatial resolution has on the operational effectiveness of autonomous ride-hailing fleets in large urban environments. A key insight from our findings is that Support Vector Machines (SVMs), while historically popular for certain regression and classification tasks, fail to scale efficiently to the large datasets necessary for fine spatiotemporal demand prediction. Their reliance on kernel computations becomes a bottleneck as both the number of data points and the dimensionality of spatial features (e.g., H3 hexagons) increase. Even with approximate kernel methods, SVMs require excessive computational resources, making them impractical for real-time, city-scale deployment where high resolution is essential for operational accuracy.

In contrast, even relatively simple neural networks consistently outperformed SVMs on both predictive accuracy and computational efficiency. This superiority is further amplified as spatial and temporal granularity increases. Neural networks offer intrinsic flexibility, particularly when equipped with embedding layers that can effectively encode H3 hexagon indices, allowing for nuanced spatial information to be captured and leveraged during training. This ability to integrate granular spatial data directly into the model architecture means that neural networks not only generalize better but also support the use of the highest feasible spatiotemporal resolution for demand prediction. We recommend prioritizing fine temporal partitioning (at least hourly) to serve demand peaks and enable precise fleet deployment strategies, which is especially critical in busy areas downtown and airport zones.

For future improvements, especially as the platform moves into operational phases, it would be highly desirable to fully map the city of Chicago into H3 hexagons and incorporate actual latitude and longitude data rather than centroids tied to census tracts. This would allow for even finer-grained modeling, minimize spatial aggregation error, and potentially unlock further gains in prediction accuracy and fleet optimization. Moreover, future iterations of the platform could benefit from ingesting real-time, external data sources such as live updates from the public transportation sector on disruptions or delays, as well as comprehensive data on public events. While access to those feeds is currently limited by paywalls and falls outside the project scope, their even-

tual integration would significantly enhance the system's ability to adapt to exogenous shocks and surges in demand.

Regarding the charging strategy for an electric autonomous fleet, this study has demonstrated the value of modeling the problem as a Markov Decision Process (MDP) and applying reinforcement learning techniques to derive smart charging policies. By integrating demand prediction with charging decision-making, autonomous agents can learn to optimize real-time charging schedules, reducing costs and ensuring vehicle readiness. Our results suggest that a hybrid charging infrastructure strategy with private charging stations strategically placed in high-demand, high-competition areas such as the city center and airport, complemented by public charging facilities in other census tracts—would best balance reliability, cost, and operational flexibility.