Final Report: Forecasting Bitcoin Transaction Fees

Partner: Trilemma Foundation

Jenny (Yuci) Zhang, Tengwei Wang, Ximin Xu, Yajing Liu

2025-06-25

Table of contents

1 Executive Summary

Bitcoin transaction fees are highly volatile and event-driven, with annual spending exceeding billions and single-day spikes reaching over $78 million (Harper 2024). Most existing tools provide only short-horizon, heuristic-based estimates for the next few blocks (<1 hour), offering limited foresight for users aiming to optimize transaction cost or timing. This project addresses that gap by forecasting fee volatility up to 24 hours ahead, with the primary goal of identifying high-volatility periods in the fastestFee tier and a secondary goal of estimating fee magnitude within those windows.

Multiple modeling approaches were explored, including classical time series methods, tree-based models, and deep learning architectures. Traditional models captured seasonality but failed to anticipate sharp spikes. The Temporal Fusion Transformer (TFT) demonstrated the best performance, capturing complex dependencies and improving both RMSE and a custom volatility-sensitive loss by 25–35%. The final product is a modular forecasting system that combines narrative-driven notebooks with executable model pipelines and open-source infrastructure for ongoing development. While limitations remain—such as limited historical data, reliance on reactive features, and the computational cost of deep learning— the system provides a practical foundation for long-horizon fee forecasting and contributes to advancing infrastructure in the Bitcoin ecosystem.

2 Introduction

In the Bitcoin network, transaction fees fluctuate sharply due to congestion, shifting incentives, and user behavior. These spikes are driven by irregular, event-based shocks — such as NFT inscription surges, exchange batching, or market volatility — rather than recurring patterns (Harper 2024). Fee data is marked by abrupt jumps and heavily influenced by off-chain events, making short-term fee selection a major challenge for users and infrastructure providers (Wang et al. 2025).

Existing tools like Bitcoin Core's estimatesmartfee (Bitcoin Core Docs n.d.) and Mempool.space offer short-term, reactive guidance based on recent block data (Bitcoin Explorer n.d.), typically covering only the next 1–6 blocks. These methods are opaque, insensitive to external drivers of volatility, and provide no insight beyond the immediate horizon. Most public tools and research treat fee prediction as a static regression problem, overlooking the timing and structure of volatility — and leaving a critical planning gap unaddressed (Li et al. 2020).

2.1 Target Objectives & Data Motivation

To make the forecasting problem tractable, we refined the partner's original question into concrete data science objectives. Rather than predicting exact fee values, we framed the task as a short-term time series forecasting problem, with a focus on identifying periods of high volatility and fee spikes.

Our primary objective was to detect upcoming windows of elevated transaction fees within a 24-hour horizon. For end users, wallets, and exchanges, this information enables better timing decisions and cost optimization—aligning more closely with real-world needs than pure point prediction. A secondary goal was to estimate fee magnitudes within those volatile windows, recognizing that even approximate directional forecasts are valuable in practice. This framing also reflects the broader challenge of modeling volatile series, where standard methods often fail to capture irregular, event-driven spikes (Taylor and Letham 2018).

We used high-frequency snapshots of the Bitcoin mempool collected over two months in early 2025. This dataset reflects real-time network activity and captures key signals related to congestion, mining, and market conditions. Prior work highlights network congestion and transaction complexity as critical drivers of transaction fees (Fan and Liu 2020). Our exploratory analysis revealed patterns such as daily cycles, short-term dependencies, and sudden spikes—reinforcing our focus on volatility-aware forecasting.

Ultimately, these refined objectives balance technical feasibility with stakeholder value, enabling a meaningful and targeted modeling strategy.

Our project addresses this gap by reframing fee prediction as a volatility-first, time-sensitive forecasting problem. For longer horizons, the timing and shape of fee spikes are often more actionable than precise point estimates. To capture these dynamics, we evaluate a diverse set of models: Holt-Winters Exponential Smoothing (HWES) (Holt 1957; Winters 1960), SARIMA (Box et al. 2015), XGBoost (Chen and Guestrin 2016), Prophet (Taylor and Letham 2018), DeepAR (Salinas, Flunkert, and Gasthaus 2017), and the Temporal Fusion Transformer (TFT) (Lim et al. 2019). These represent a progression from classical time series models to tree-based regressors and deep learning architectures. We assess them using both standard metrics (e.g., RMSE, MAPE) and a custom composite loss designed to penalize deviation from spike shape, timing, and volatility.

We find that TFT significantly outperforms other models, improving key evaluation metrics by 25–35% over baseline approaches. This demonstrates its ability to model non-periodic fee behavior and support forward-looking decision-making. The remainder of this report is organized as follows: Section III outlines the data science techniques used, including preprocessing, model selection, and evaluation strategy. Section IV presents the data product and results, detailing model performance, intended use, and the system's extensibility. Section V concludes with key takeaways, limitations, and recommendations for future development.

3 Data Science Techniques

3.1 Dataset Overview

We used data collected from mempool.space via their public API, capturing snapshots of the Bitcoin mempool every 5 minutes between March and May 2025. Each snapshot represents the network state at a given moment and includes metrics such as mempool congestion, transaction volume, block production, mining difficulty, and BTC price. These snapshots form a time series dataset that reflects both blockchain-level activity and market demand conditions.

The mempool serves as a waiting room for unconfirmed Bitcoin transactions. Users compete to have their transactions included in upcoming blocks by attaching fees, and when congestion rises, miners prioritize higher-fee transactions. This dynamic makes mempool data a strong signal for fee forecasting.

Figure 1: Visual illustration of Bitcoin mempool congestion over time (source: mempool.space).

3.2 Feature Preprocessing

To prepare the data, we conducted exploratory correlation analysis between features and the target variable fastestFee, which confirmed that several predictors contained useful signals. This was not used for formal feature selection, but helped validate feature relevance.

To prevent target leakage, we removed a few features (e.g., hourFee) that had extremely high correlation with the target (over 0.97). These were excluded from all models except DeepAR and TFT, where model constraints made removal more complex.

We also applied several feature engineering steps. We first examined the distribution of fastestFee and found it to be highly right-skewed (see Figure 2), with most values clustered at the low end and a few extreme spikes. This heavy skew can hinder model stability and violate common assumptions.

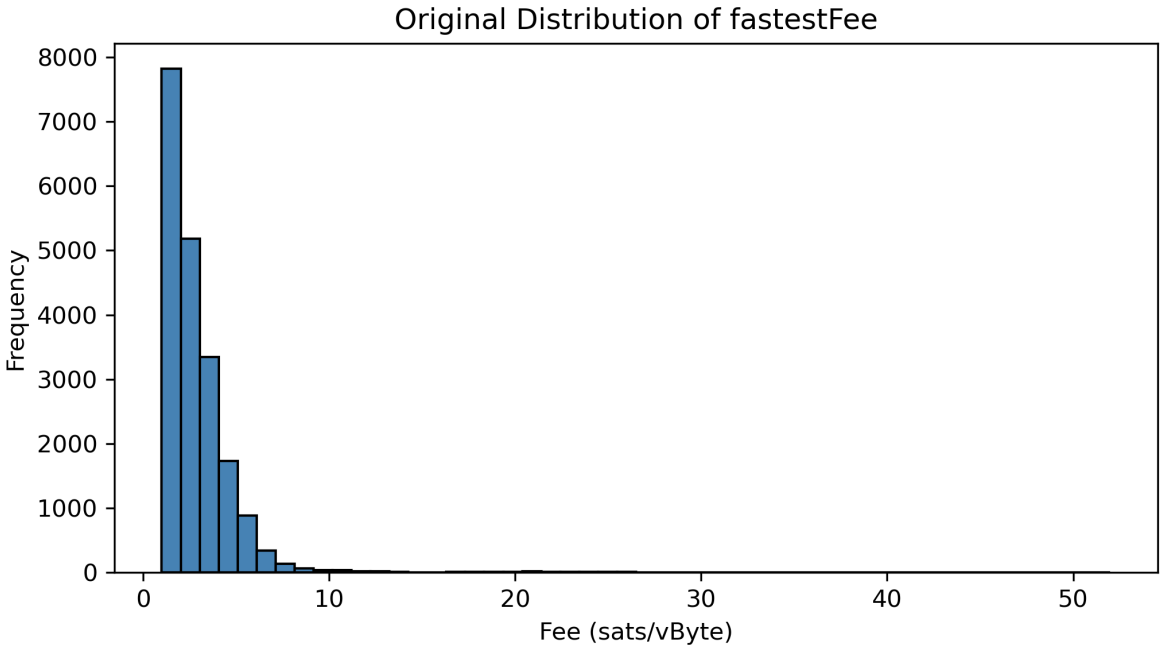

Figure 2: Original distribution of fastestFee shows strong right skew.

To mitigate this, we applied a logarithmic transformation to fastestFee, which compresses large values and helps stabilize variance across the series—making it more suitable for forecasting. Second, we resampled the original 5-minute data into 15-minute intervals to reduce noise and enhance short-term signal clarity.

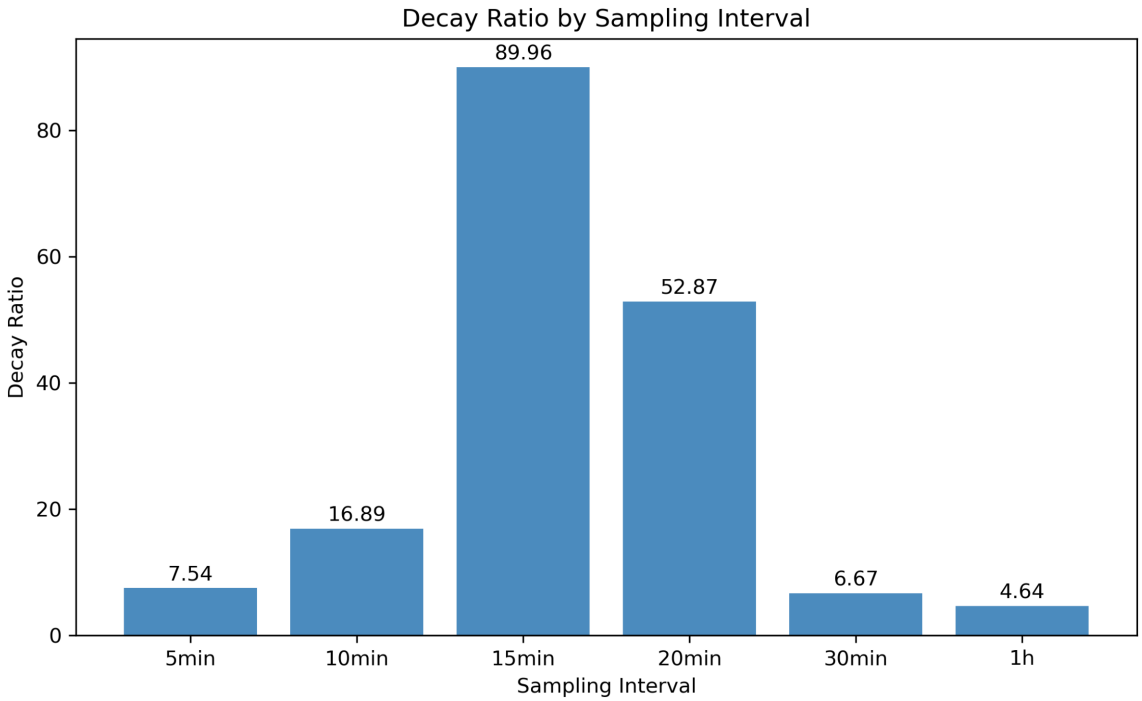

Figure 3: Decay ratio by sampling interval. Higher ratios indicate stronger AR(1)-like structure and better short-term predictability.

This choice was motivated by decay ratio analysis, which evaluates the signal strength of short-term temporal dependencies. As shown in Figure 3, the 15-minute interval yielded the strongest AR(1)-like pattern among tested frequencies, supporting its selection for resampling. Finally, we created lagged variables and rolling aggregations to help models capture short-term temporal dependencies. Finally, we created lagged variables and rolling aggregates to help models capture temporal dependencies. The final feature set retained most original variables except those flagged for leakage, balancing completeness with modeling integrity.

3.3 Models

To understand and anticipate Bitcoin transaction fee rate dynamics, we implemented a sequence of models. Each of the models were selected for its ability to address specific limitations observed in the previous ones. Below, we walk through these choices, results, and where each model fell short.

3.3.1 Dummy Model (Global Median)

We began with a simple dummy model shown in Figure 4 that always predicted the global median transaction fee rate. Although it had no predictive power, it served as a baseline to measure improvements from more sophisticated approaches. This model completely ignored any temporal or external structure in the data, but offered a useful starting point to quantify how difficult the prediction task really was.

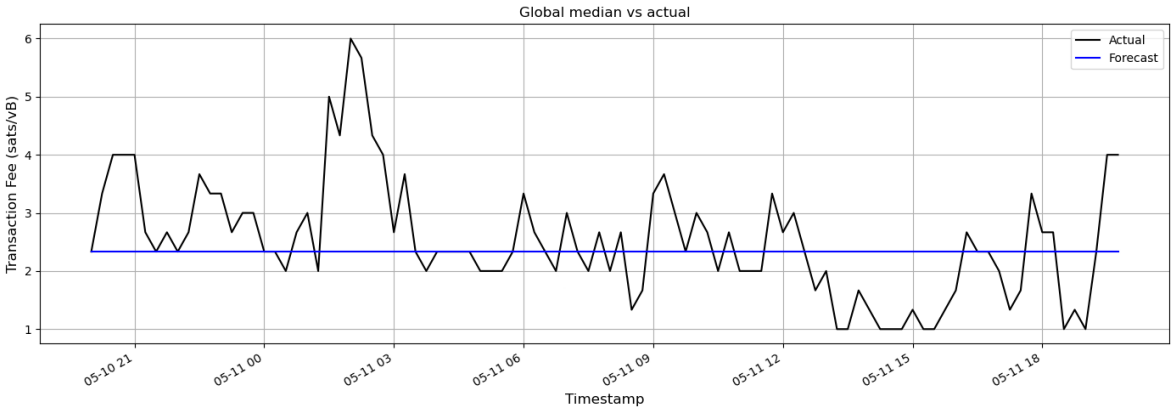

Figure 4: Global median vs actual values

3.3.2 Holt-Winters Exponential Smoothing (HWES)

Given the presence of regular daily cycles in the data, Holt-Winters Exponential Smoothing was a natural next step. As it was shown in Figure 5, it captured seasonality fairly well and improved upon the dummy model. However, analysis of the residuals revealed persistent autocorrelation, suggesting that the model failed to account for important lag effects or hidden patterns beyond periodic behavior.

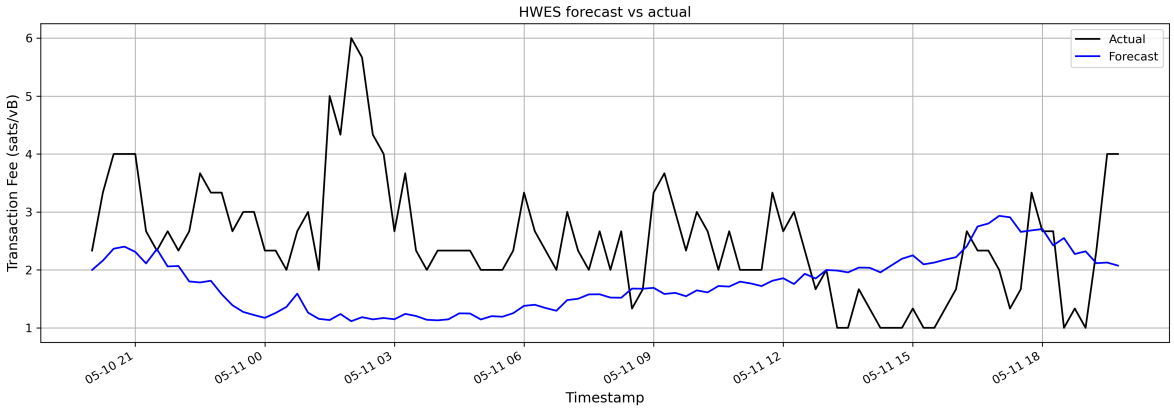

Figure 5: Forecasting results of HWES vs actual values

3.3.3 SARIMA

To address the temporal dependencies missed by HWES, we adopted SARIMA, which supported autoregressive and seasonal differencing components. The results were shown in Figure 6. Because of its capacity to learn from prior values, SARIMA produced a better short-term fit. However, its univariate nature prevented us from including important exogenous features like transaction count, mempool congestion, or size distributions. That limited its practical usefulness in a multi-variable environment.

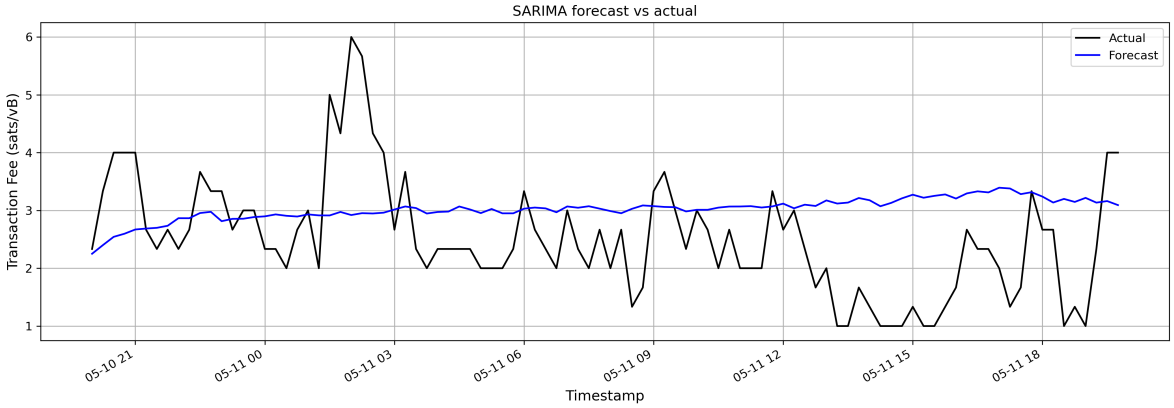

Figure 6: Forecasting results of SARIMA vs actual values

3.3.4 XGBoost

We then turned to XGBoost, a powerful tree-based model capable of incorporating a broad array of features. Through Figure 7, this model significantly expanded our input space and enabled non-linear interactions among variables. While its numerical performance improved, especially on average error metrics like MAPE, the model still struggled with volatility. It produced smooth and flat outputs that failed to capture sudden fee spikes. However, those spikes were precisely the events most critical for users.

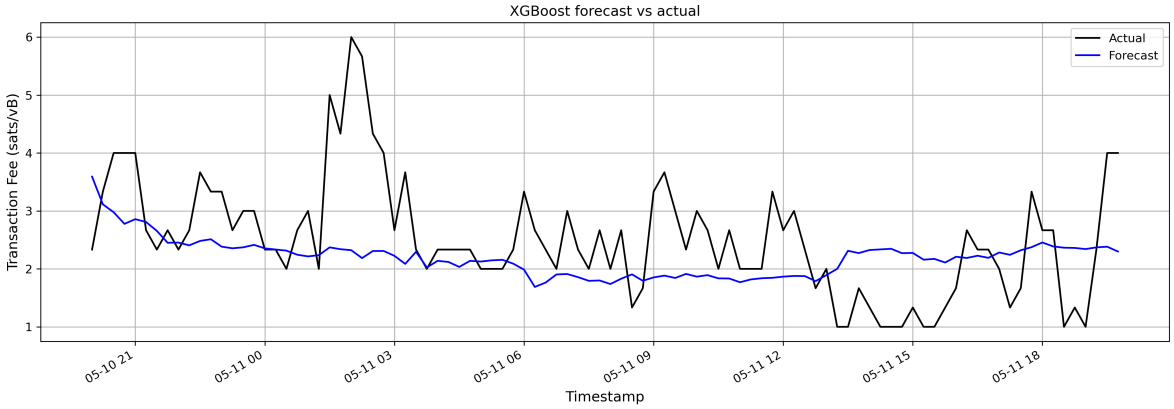

Figure 7: Forecasting results of XGBoost vs actual values

3.3.5 Prophet

We explored Facebook's Prophet model to take advantage of its flexibility in modeling seasonality, changepoints, and custom events. Prophet brought in useful priors for time series with irregular behavior and offered a simple interface for integrating domain knowledge. Unfortunately, as the results shown in Figure 8, it still smoothed over the spikes and underperformed in capturing real-time fee volatility. Its strength in trend estimation did not translate well to our highly reactive use case.

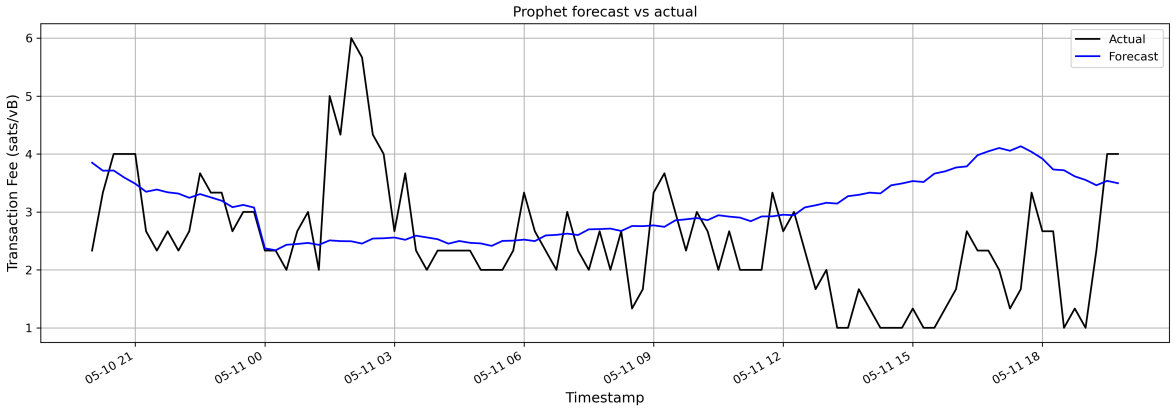

Figure 8: Forecasting results of Prophet vs actual values

3.3.6 DeepAR

To try more dynamic modeling, we implemented DeepAR, which was an LSTM-based autoregressive forecasting model. In theory, DeepAR should have leveraged temporal context more effectively and handled sequential data better. However, the outputs in Figure 9 were unstable and noisy. The results often failed to align with real-world fee movements. The model demonstrated limited generalizability, and its probabilistic forecasts often appeared more random than informative.

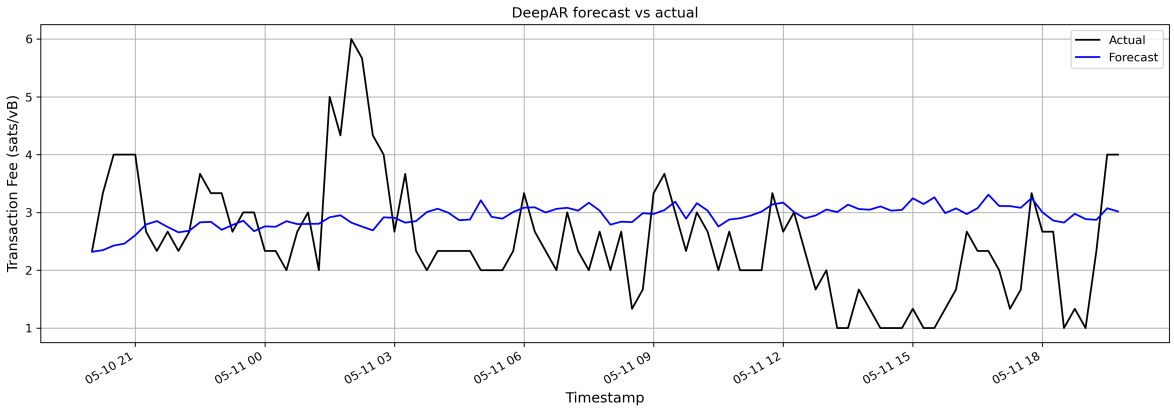

Figure 9: Forecasting results of DeepAR vs actual values

3.3.7 Temporal Fusion Transformer (TFT)

Our final and most advanced model was the Temporal Fusion Transformer(TFT). TFT was designed to integrate static covariates, time-varying features, attention mechanisms, and variable selection into a unified deep learning architecture. Among all models, TFT came closest to capturing both overall volatility and individual spike events. In Figure 10, it successfully learned temporal dependencies, responded to feature relevance dynamically, and produced the most realistic forecasts. While computationally expensive and complex to tune, its performance and interpretability made it the strongest candidate for this forecasting task.

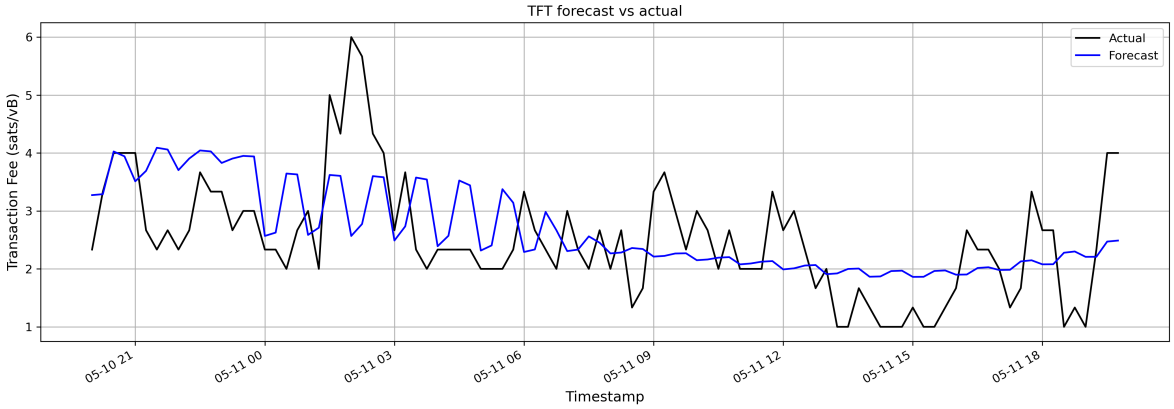

Figure 10: Forecasting results of TFT vs actual values

3.3.8 What Might Have Worked Better

3.3.8.1 Current Limitation

Most models react to spikes after they happen due to:

- Lagged exogenous features.
- Absence of true leading indicators.
- Volatile, abrupt, and irregular nature of spikes.

3.3.8.2 Potential Improvement

A more forward-looking architecture could be created via multi-stage forecasting, e.g.:

- Train models to predict key exogenous signals (e.g., mempool congestion, pending transaction counts).
- Use their predicted values as inputs into the main fee prediction model.

3.3.8.3 Why Not Implemented

- Partner's feedback during mid-project phase discouraged premature inclusion of predicted external signals.
- Additional models would introduce forecast compounding errors.
- Partner had tried social sentiment modeling in previous work (e.g., scraping tweets, news) but results were unsatisfactory.
- Team chose to focus on maximizing what could be done within the current data window and feature scope.

3.4 Evaluation Metrics

We used multiple metrics to evaluate the forecasting models, selected based on both standard practice and alignment with stakeholder needs. The Mean Absolute Percentage Error (MAPE) served as the primary metric due to its interpretability and widespread use in forecasting tasks. In addition, we used Root Mean Squared Error (RMSE) to assess each model's ability to capture spikes, since RMSE penalizes large errors more heavily—an important consideration for volatile fee behavior.

To better capture the characteristics of Bitcoin fee spikes, we designed a custom composite loss function for model training. This loss combines three components: base error (MAE), volatility mismatch (standard deviation loss), and spike timing error (deviation error), which is calculated as: Total Loss = Base Loss + Std Loss + Deviation Error. While traditional losses like MAE or MSE tend to smooth over volatility, our formulation explicitly rewards models that preserve sharp changes and dynamic patterns. A breakdown of each term is shown in Table Table 1.

Component	**Base Loss**	**Std Loss**	**Deviation Error**
Calculation	`y_pred - y_true`	`std_pred - std_true`	`(y_pred - ȳ_pred) - (y_true - ȳ_true)`
Captures	Raw error	Overall volatility mismatch	Dynamic (pointwise) pattern mismatch
Relevance to Spikes	Underweights spikes	Penalizes smoothing	Captures spike timing

Table 1: Breakdown of custom loss function components.

This design was inspired by volatility-sensitive training regimes used in interpretable deep time series models such as the Temporal Fusion Transformer (Lim et al. 2019).

3.5 Stakeholder Impact & Potential Ethical Concerns / Limitations

Our evaluation metrics and modeling choices were designed to align with stakeholder priorities. For end users, forecasting the timing of high-fee periods is more valuable than precise predictions, as it enables them to avoid costly transaction windows. Wallet developers use these forecasts to provide timely fee suggestions, while miners may use them to optimize revenue.

However, deploying such tools raises ethical concerns. Users may over-rely on forecasts, mistaking them for guarantees. To address this, outputs should communicate uncertainty and limitations. There is also a risk of manipulation—public forecasts could be exploited to game mempool behavior. In addition, unequal access to forecasting tools may reinforce disparities. Promoting transparency and open access helps mitigate these risks.

Some broader concerns remain unaddressed, including fairness across user groups, potential shifts in miner incentives, and susceptibility to coordinated manipulation. Future work should consider these dimensions through equity-aware metrics and policy-informed design.

4 Data Product and Results

4.1 Overview of the Data Product

This data product directly supports Trilemma Capital's mission of serving industry talent and advancing Bitcoin infrastructure through data science, both educational and technical. Its design intentionally tailors to three core audiences. General users and institutions can rely on the 24-hour forecasts to plan transactions and reduce fee costs. Learners and educators receive a transparent, step-by-step walkthrough of Bitcoin-fee forecasting and time-series methodology. Industry experts and partners see infrastructure-grade modeling practice embodied in a modular pipeline and well-documented repository. The product is purposefully modular. Jupyter notebooks guide users through EDA, modeling decisions, and final TFT results. Python scripts implement a structured pipeline for reproducible experiments and easy re-training on new data. Finally, the open-source GitHub repository—with clear documentation—enables collaboration, scalability, and long-term extensibility.

4.2 Results

Model comparison tables and the forecast plots illustrate how predictive fidelity improves as we progress from classical statistics to deep learning. Baselines such as HWES and SARIMA track the day-level seasonal drift but miss the sharp dips and spikes that dominate the fee landscape, as shown in Figure 11.

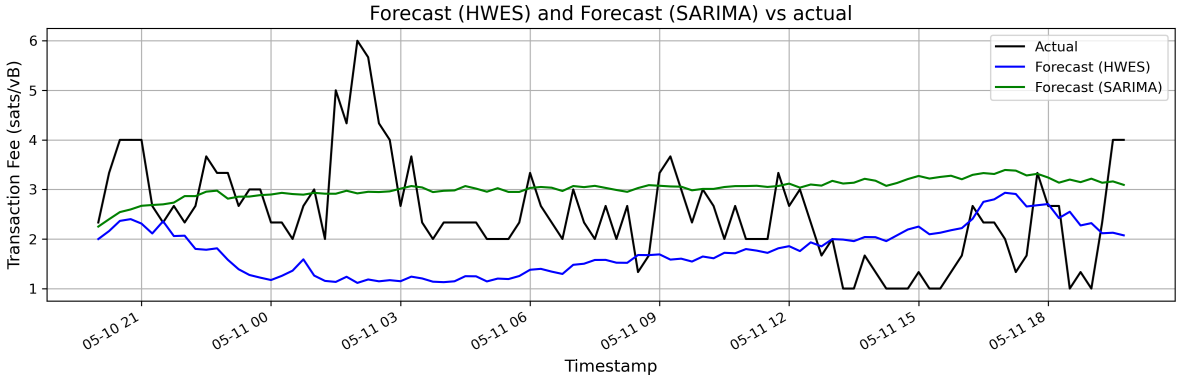

Figure 11: HWES and SARIMA forecasts vs actual fee (test set)

Prophet, with its flexible trend and built-in seasonality terms, improves the global fit but still smooths over most intraday spikes, yielding a custom loss of about 2.43 and an RMSE just about 1.34. XGBoost, which ingests engineered lags and mempool signals, pushes average error lower than any of the purely statistical models. This model yields an RMSE of 1.04, but continues to understate high-frequency volatility, as shown in Figure 12. The custom loss of XGBoost is 2.20, indicating that while the model captures general trends, it still struggles to fully account for the sharp fee spikes and intraday volatility present in the data.

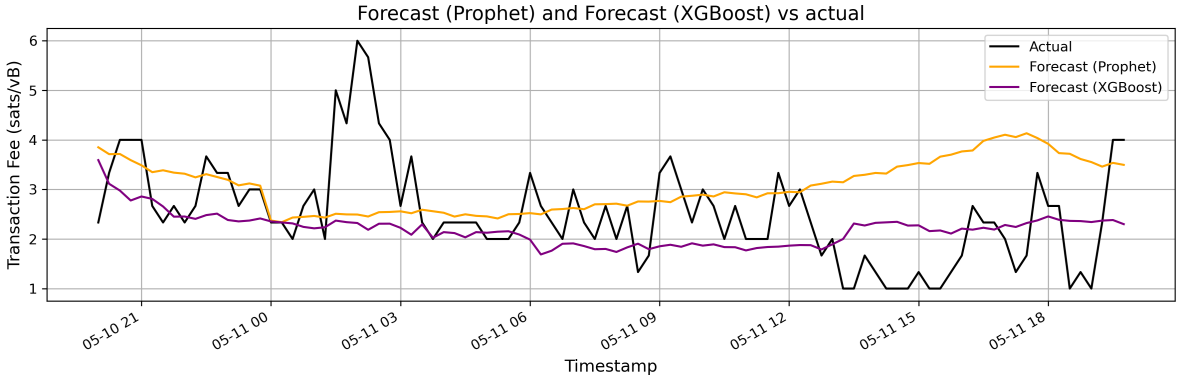

Figure 12: Prophet and XGBoost forecasts vs actual fee (test set)

The neural models represent a meaningful shift in modeling capacity. While DeepAR introduces sequential awareness through recurrence, it falls short of Prophet and XGBoost in short-term fee tracking, with a custom loss of 2.55 and RMSE of 1.15. It is the TFT that best synchronises with both the amplitude and timing of sudden fee surges (darkcyan versus black traces in Figure 13).

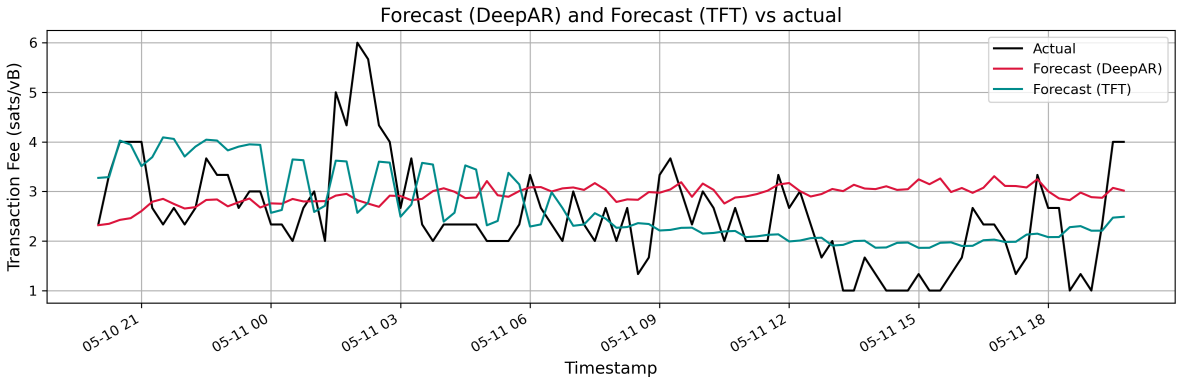

Figure 13: DeepAR and TFT forecasts vs actual fee (test set)

Quantitatively, TFT reduces the bespoke volatility-aware loss from about 2.20, the lowest among the baselines, to 1.78, representing a 19% improvement. It also brings RMSE down from the 1.04–1.44 range to 0.94, a relative gain of 9%. MAE, MAPE, and distribution-shape penalties follow the same pattern, confirming that TFT strikes the strongest balance between overall accuracy and sensitivity to congestion-driven shocks.

4.3 Strength and Competitive Edge

The product's chief strength is that it predicts rather than reacts. Where mainstream tools like estimatesmartfee or Mempool.space publish heuristics for the next few blocks, this project delivers reproducible code that extends the horizon to an entire day and outputs tiered curves—fastest, half-hour, hourly, economy, and minimum—so users can weigh urgency against cost. The product is fully transparent: every notebook, script, and utility function is openly shared on GitHub and documented with detailed inline comments. This openness allows peers to audit modeling assumptions, reproduce plots, and validate performance metrics with ease. Its modular design further reinforces this transparency. Developers can experiment freely—by changing the loss function, introducing new input features, or retraining a specific model—without disrupting the broader pipeline. The educational structure of the notebooks sets this product apart. It not only walks newcomers through key time-series modeling concepts and trade-offs but also provides experienced users with a clear path to automated, command-line execution. Together, these qualities establish Trilemma not as a provider of short-term heuristics, but as a builder of forward-looking, evidence-based infrastructure for the Bitcoin ecosystem.

4.4 Limitations

Despite its strengths, the product faces several limitations. First, the models do not yet incorporate real-time external signals, such as exchange outflows or policy announcements. This is making them less responsive to abrupt market events that drive sudden fee spikes. Second, The loss function uses fixed weights, which limits its ability to adapt dynamically to changing market regimes. Third, predictive intervals are not yet implemented, which may reduce confidence for risk-sensitive users. Fourth, the deep learning models, particularly transformers, require GPU acceleration or cloud infrastructure, which can raise the barrier for experimentation and increase the cost of continuous deployment. Forecast performance will also degrade over time unless models are periodically retrained to reflect evolving network conditions. Lastly, while the notebooks are well-documented and educational, the absence of a live dashboard or public API limits accessibility for non-technical users, who would need to develop a custom interface to integrate the forecasts into practical workflows.

4.5 Future Directions

These limitations above, however, reflect intentional trade-offs made to balance flexibility, transparency, and cost. Keeping each model in a separate script makes training costs transparent and allows teams to execute only the components they need. This modular design supports flexible development but comes at the expense of centralized orchestration, such as a unified Makefile. Looking ahead, enhancements like adaptive loss weighting, uncertainty quantification, and real-time signal ingestion could improve resilience to edge cases. However, each would introduce added computational cost and complexity. While a real-time API remains an appealing long-term goal, the high cost of hosting transformer models continuously makes it impractical today. In the meantime, scheduled batch forecasts offer a pragmatic balance between accuracy, interpretability, and operational efficiency.

5 Conclusion and Recommendations

This project set out to address the challenge of forecasting Bitcoin transaction fees—a notoriously volatile and difficult-to-predict signal shaped by network congestion, user incentives, and unpredictable events. Recognizing the limitations of existing tools that offer only near-term, reactive guidance, we reframed the problem as one of volatility forecasting over a 24-hour horizon. Through extensive experimentation across statistical models, tree-based learners, and deep sequence models, we found that modern deep learning architectures—particularly the Temporal Fusion Transformer—are well-suited to capturing irregular spike patterns. The resulting system moves beyond average-point estimation to provide actionable foresight, helping users, wallets, and infrastructure providers make more informed decisions around transaction timing.

Despite these gains, several constraints remain. The dataset used spans just two months, limiting exposure to rare but impactful phenomena. Many features that help explain fee behavior—such as mempool congestion or block composition—are available only with a lag, reducing our ability to predict true leading signals. Furthermore, while deep models like TFT are powerful, they require regular retraining and high computational resources, which may constrain deployment or accessibility. Our custom loss function, though aligned with business needs, remains static and could be better tuned to balance volatility sensitivity and accuracy.

Looking forward, future work could explore alternate data sources and modeling strategies that address these constraints. First, longer historical coverage or the inclusion of off-chain indicators may improve predictive range. Second, deeper exploration and adaptive tuning of the custom loss function may further align model behavior with user pain points around volatility and timing. Third, future approaches might benefit from hybrid pipelines—where intermediate signals (e.g., mempool congestion or block inclusion rates) are predicted first and then fed into the final fee forecasting model. Finally, probabilistic forecasting and uncertainty quantification could add value once magnitude prediction improves.

6 Appendix

6.1 Terminology

Term	Definition
Bitcoin	Unit of currency is called "bitcoin" with a small b, and system is called "Bitcoin," with a capital B. "bitcoin" is a virtual currency (cryptocurrency) designed to act as money and a form of payment outside the control of any one person, group, or entity (i.e. decentralized).
Bitcoin Address	"1DSrfJdB2AnWaFNgSbv3MZC2m74996JafV" An encoded base58-check version of a public key 160-bit hash consists of a string of letters and numbers. Think of it analogous to an email address when sending someone an email.
Blockchain	A decentralized digital ledger that records transactions across a network of computers, making it transparent, immutable, and resistant to tampering. Technology used by Bitcoin.
Fees	The sender of a transaction often includes a fee to the network for processing the requested transaction. Most transactions require a minimum fee of 0.5 mBTC (millibitcoin) = 0.0005 BTC. Typical unit measurement in satoshi/bytes.
Hash	A function that converts an input of letters and numbers into an encrypted output of a fixed length. The hash is irreversible, meaning it cannot be decrypted back to the original input. Hashes are used in Bitcoin to create blocks and verify transactions.
Mempool	The bitcoin Mempool (memory pool) is a collection of all transaction data in a block that have been verified by Bitcoin nodes, but are not yet confirmed.
Mining / Miner	A process/network node that finds valid proof of work for new blocks, by repeated hashing.
Node	Refers to blockchain stakeholders and their devices that keep a copy of the distributed ledger and serve as communication points within the network. Major purpose is to verify the validity of the transactions within a particular blockchain.
Proof-of-Work	A piece of data that requires significant computation to find; In bitcoin, miners must find a numeric solution to the SHA256 algorithm that meets a network-wide target, the difficulty target.

Satoshi	The smallest denomination of bitcoin that can be recorded on the blockchain. 1 Bitcoin is equivalent to 100 million satoshis, named after the creator of Bitcoin, Satoshi Nakamoto.

Table 2: Key Terms and Definitions in Bitcoin and Blockchain (Alphabetically Ordered)

References

Bitcoin Core Docs. n.d. "Estimatesmartfee." https://developer.bitcoin.org/reference/rpc/estimatesmartfee.html.

Bitcoin Explorer. n.d. "Mempool.space." https://mempool.space/.

Box, George E. P., Gwilym M. Jenkins, Gregory C. Reinsel, and Greta M. Ljung. 2015. *Time Series Analysis: Forecasting and Control*. 5th ed. Wiley.

Chen, Tianqi, and Carlos Guestrin. 2016. "XGBoost: A Scalable Tree Boosting System." *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785–94. https://doi.org/10.1145/2939672.2939785.

Fan, Ying, and Xiaotong Liu. 2020. "The Determinants of Bitcoin Transaction Fees: Evidence from the Mempool." *Finance Research Letters* 35: 101289. https://doi.org/10.1016/j.frl.2019.101289.

Harper, Colin. 2024. "Bitcoin Transaction Fees Hit Record Levels After Halving — Here's Why." https://www.forbes.com/sites/colinharper/2024/04/22/bitcoin-transaction-fees-hit-record-levels-after-halving---heres-why/.

Holt, Charles C. 1957. "Forecasting Seasonals and Trends by Exponentially Weighted Moving Averages." *International Journal of Forecasting* 20 (1): 5–22. https://doi.org/10.1016/0169-2070(94)00023-X.

Li, Xiaoqi, Peng Jiang, Ting Chen, Xiapu Luo, and Qiaoyan Wen. 2020. "A Survey on the Security of Blockchain Systems." *Future Generation Computer Systems* 107: 841–53. https://doi.org/10.1016/j.future.2017.08.020.

Lim, Bryan, Sercan O. Arik, Nicolas Loeff, and Tomas Pfister. 2019. "Temporal Fusion Transformers for Interpretable Multi-Horizon Time Series Forecasting." *arXiv Preprint arXiv:1912.09363*. https://arxiv.org/abs/1912.09363.

Salinas, David, Valentin Flunkert, and Jan Gasthaus. 2017. "DeepAR: Probabilistic Forecasting with Autoregressive Recurrent Networks." *arXiv Preprint arXiv:1704.04110*. https://arxiv.org/abs/1704.04110.

Taylor, Sean J., and Benjamin Letham. 2018. "Forecasting at Scale." *The American Statistician* 72 (1): 37–45. https://doi.org/10.1080/00031305.2017.1380080.

Wang, Minxing, Pavel Braslavski, Vyacheslav Manevich, and Dmitry Ignatov. 2025. "Bitcoin Ordinals: Bitcoin Price and Transaction Fee Rate Predictions." *IEEE Access*. https://doi.org/10.1109/ACCESS.2025.3541302.

Winters, Peter R. 1960. "Forecasting Sales by Exponentially Weighted Moving Averages." *Management Science* 6 (3): 324–42. https://doi.org/10.1287/mnsc.6.3.324.