Proposal Report: Forecasting Bitcoin Transaction Fees

Partner: Trilemma Foundation

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

2025-05-09

Table of contents

1 Executive Summary

This project aims to develop a machine learning-based tool to forecast Bitcoin transaction fees over a 24-hour horizon. Existing tools are limited to short-term forecasts (1-6 blocks), which aren't sufficient for users planning batch transactions or optimizing timing. Our tool fills this gap by providing longer-term forecasts with confidence intervals to show forecast precision and reliability. We will compare baseline models like ARIMA and XGBoost with advanced models such as Prophet, DeepAR, and Temporal Fusion Transformers (TFT) to ensure both accuracy and robustness. We ultimately plan to deploy the tool using AWS Lambda for real-time forecasts using mempool data such as transaction volume, fee rates, and network conditions.

2 Introduction

Bitcoin transaction fees are determined by a market-based mechanism where users compete for limited block space (Easley, O'Hara, and Basu 2019). This causes fees to fluctuate significantly within short timeframes, making it difficult for users to plan batch transactions or optimize timing. Most existing tools offer only short-term forecasts—typically within the next 10 to 60 minutes—which are insufficient for users who need to make decisions over longer time horizons. In addition, fee-related data is often undocumented or inconsistently structured, adding complexity to model development.

To address this gap, our project proposes a machine learning-based tool that forecasts Bitcoin transaction fees up to 24 hours in advance. By leveraging real-time mempool data—including transaction volume, fee rate dynamics, and network conditions—we aim to build a robust and accurate forecasting system. This system not only predicts the expected fees but also provides confidence intervals to help users assess the reliability of the predictions. The tool will be deployed via AWS Lambda to support scalable, real-time access. Our approach involves benchmarking baseline models like ARIMA (Box et al. 2015), Holt-Winter's Exponential Smoothing (Holt 1957; Winters 1960), and XGBoost (Chen and Guestrin 2016) against more advanced models Prophet (Taylor and Letham 2018), DeepAR (Salinas, Flunkert, and Gasthaus 2017), and TFT (Lim et al. 2019) to ensure high predictive performance under volatile network conditions. To support this, we anticipate the need for scalable training environments and extended data horizon—especially for high-capacity models like TFT—and may explore partnerships or cloud-based solutions to achieve this.

3 Dataset and Features

This rich dataset sources data from *mempool.space* and provides the necessary historical context for building accurate fee rate forecasting models.

3.1 Mempool Overview

We use Bitcoin Mempool data (Bitcoin Explorer n.d.), which records hourly time series information about unconfirmed transactions waiting to be confirmed in the blockchain (Figure 1). The mempool acts as a dynamic queue, with transaction size represented by block space and fee ranges indicated by color, where red signals higher fees and green signals lower fees.

Figure 1: Real-time data of Bitcoin Mempool.

3.2 Data Description

Our dataset (Table 1) consists of 11,902 hourly entries and 67 features across several categories relevant to fee prediction, including recommended fee rates, mempool statistics, fee histograms, mining difficulty adjustments, and price data in various fiat currencies. The projected data covers mempool blocks, transaction sizes, and associated fee ranges. Fee rate features include recommended fees for different confirmation targets (fastest, half-hour, hour, minimum, economy) and 37 binned fee categories based on transaction counts. Mempool statistics capture aggregate data like transaction count, virtual size (vsize), and total fees per time step. Difficulty adjustments include the current difficulty epoch, progress percentage, and estimated time to the next retarget event.

Category	Description	Columns	Range
Projected Data	Projected mempool blocks, transaction sizes and their fee ranges	5	Varies by feature / type
Fee Rates	Different types of recommended fee rates and confirmation targets. Fees are also binned by counts.	5 fee rates e.g. fastest 37 binned	Min: 1 sat/vbyte Max: 41 sats/vbyte Median: 2–3 sats/vbyte
Mempool State	Aggregate mempool statistics (count, vsize and total_fee)	3	count: 24–169 vsize: 1.2e4–5.6e7 vbytes fee: 1.2e4–2.3e8 sats
Difficulty Adjustment	Data related to the current difficulty epoch (progress%, estimated retarget)	10	Varies by feature / type
BTC Prices	Current BTC price data in multiple currencies	7	Median: 83,845 USD Max: 92,736 USD

Table 1: Data descriptions by categories.

4 Data Science Techniques

To build reliable predictions of Bitcoin transaction fees, effective data preprocessing into sensible inputs and appropriate model selections are critical.

4.1 Data Preprocessing

First and foremost, we select the *fastestFee* rate, recommended fee for fastest confirmation, under Fee Rates category in Table 1 as the primary response variable, as it serves as a clear benchmark for users prioritizing transaction speed. Given the right-skewed nature of the fee distribution, we apply a logarithmic transformation to improve model performance.

Several other challenges arise during this process. First, we detected outliers that may represent data entry errors or anomalies, such as BTC prices recorded as -1. Second, documentation from the data source is limited. Third, transaction behavior is expected to vary over time, with differences across weekdays and weekends, as well as possible seasonal patterns. Finally, relationships between variables like transaction volume and fee rates may hold important signals that require further exploration.

To address these issues, we intend to first correct outliers and abnormal values through imputation. We also inspect feature definitions and cross-validate them to understand how certain features are computed. We further analyze and visualize the time-series data for temporal patterns and signs of seasonality, subject to the period covered. Then, we compute pairwise correlations across different features (e.g. transaction volume and fee rates) and amongst similar features (e.g. median fee and total fees) to uncover meaningful relationships. Finally, we consider engineering time-based features—such as hour of day and day of week—as well as indicators of mempool congestion and major positive / negative events to enrich the feature space for downstream modeling.

4.2 Forecasting Methodology

The overall feature structure reflects both temporal dependencies and external influences, such as time-of-day effects, network congestion, and market conditions. Since our data contains both time-series patterns and contextual signals from network and market activity, we deliberately combine autoregressive, tree-based, and deep learning models to address different aspects of this complexity.

To tackle the challenges of predicting Bitcoin fee rates—like changing patterns over time, external factors, and non-linear behavior—we start with a few simple but solid baseline models. To understand how things like mempool congestion or the day of the week affect fees, we use **Linear Regression**. It's not meant for forecasting, but it gives us a clearer idea of which external factors are actually influencing the fee rates. Then, to capture how fee rates depend on their own past values over time, we use **ARIMA**, which is well-suited for modeling time-based patterns and serves as a good starting point for temporal forecasting. Since fee rates often show short-term trends and recurring daily patterns, we use **Holt-Winters Exponential Smoothing (HWES)** to capture these seasonal behaviors without needing additional input features. Finally, to deal with more complex nonlinear interactions, we bring in **XGBoost**, a powerful tree-based model that works well with structured data and can handle both historical and external features.

As the project moves forward, we need models that can better deal with the complexity and unpredictability of the data. For example, we want to understand broader patterns like seasonal cycles with holiday effects in fee rates. That's where **Prophet** comes in—it breaks the time series into interpretable parts like trend and seasonality, giving us a clearer picture of the overall structure. While it's more suited for daily data and doesn't respond well to sudden changes, it's still helpful for spotting big-picture patterns. We also need a way to model multiple related time series—like our detailed fee histogram bins—while accounting for uncertainty. **DeepAR** is a good fit for this, since it learns from sequences and produces probabilistic forecasts, which is especially useful in such a volatile environment. Lastly, to make sense of all the moving parts in our high-dimensional dataset, we use the **Temporal Fusion Transformers (TFT)**. It can focus on the most important features at each point in time using attention and variable selection, making it powerful for capturing complex relationships.

That said, it's a heavy model that takes more computation and needs careful tuning to avoid overfitting.

Figure 2 shows the inputs and expected outputs for the respective models.

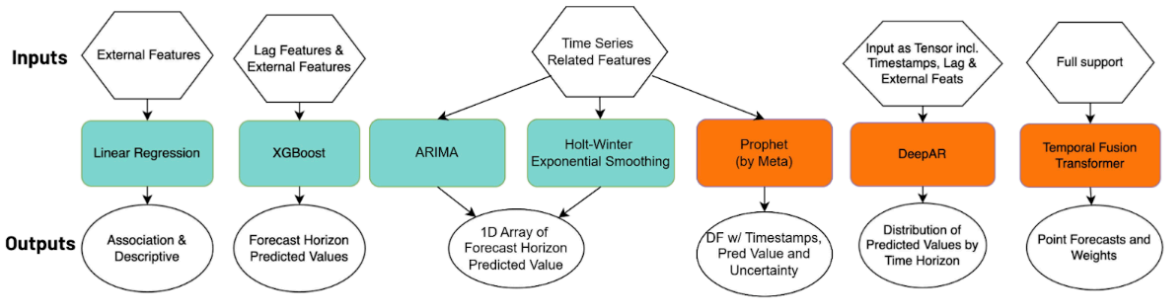

Figure 2: Diagram of model inputs and outputs for baseline and advanced forecasting models.

To assess model performance, we use **Mean Absolute Error (MAE)** and **Root Mean Squared Error (RMSE)** as our primary evaluation metrics through back-testing and a train-test split approach. MAE provides an interpretable average of prediction errors in the same unit as the target variable, while RMSE penalizes larger deviations more strongly, making it useful for identifying models that are sensitive to extreme fluctuations. The goal is to achieve lower error and variance compared to baseline models, ensuring that the model not only predicts accurately but also generalizes well to unseen data. For probabilistic models like DeepAR and TFT, we also evaluate the **calibration of predicted confidence intervals** to ensure that they reliably reflect observed variability in fee rates. This is critical for making informed, risk-sensitive decisions in high-variance network environments.

5 Workflow & Timeline

The following section presents the workflow and provides a detailed project timeline for the duration of the capstone.

5.1 Workflow

The workflow diagram (Figure 3) outlines the project's end-to-end structure, ensuring adherence to best data science practices. We begin with raw Bitcoin mempool data, followed by EDA, feature engineering, and model development. Next, we establish baseline models to set benchmarks and then progress to advanced models for improved accuracy and reliability. Ultimately, the best-performing model will be deployed to AWS for real-time fee prediction.

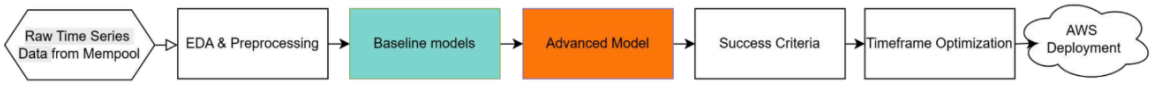

Figure 3: Workflow of the modeling pipeline from data preprocessing to deployment.

5.2 Timeline

Week	**Milestone**
Week 1	Write proposal report, conduct initial EDA, and create a full data specification notebook to document and visualize all available features and the target.
Week 2	Train and evaluate baseline models, including ARIMA, Holt-Winters, and XGBoost; Use Linear Regression for exploratory analysis of external features.
Week 3	Build advanced models such as Prophet, DeepAR, and Temporal Fusion Transformer.
Week 4	Continue building advanced models if needed; test and compare their performance against baselines using standard metrics.
Week 5	Perform timeframe optimization by analyzing model performance across the 24-hour forecast horizon.
Week 6	Integrate modeling pipeline for AWS deployment; finalize deliverables and prepare for final presentation.

Table 2: Weekly Project Timeline

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

References

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.

Easley, David, Maureen O'Hara, and Suman Basu. 2019. "From Mining to Markets: The Evolution of Bitcoin Transaction Fees." *Journal of Financial Economics* 134 (1): 91–109. https://doi.org/10.1016/j.jfineco.2019.02.008.

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.

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.

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.