Time-frequency analysis for non-stationary signals

Yu Luo, Chen Qian

June 15, 2025

Abstract

In this project, we investigate two signal processing methods — Wavelet Packet Transform (WPT) and Pseudo Wigner–Ville Distribution (PWVD) — and apply them to two distinct types of time-series data: stock price signals and EEG (electroencephalogram) recordings. The objective for the stock data is to predict the direction of future price movements, while the goal for the EEG data is to classify abnormal (pathological) brain activity versus healthy signals. By applying both methods to these datasets with differing predictive tasks, we aim to evaluate and compare their effectiveness across financial and biomedical domains.

1 Introduction

When signals become non-stationary—i.e., when their covariance structure changes over time—traditional signal models (in time signal) such as autoregressive (AR) models and spectral analysis in the frequency domain often fail to adequately capture their dynamic characteristics. This is due to the fact that non-stationary signals exhibit more intricate and dynamic behavior across both the time and frequency domains. On the other hand, time-frequency analysis is particularly well-suited for analyzing such signals, as it leverages information from both domains simultaneously. Motivated by this fact, we present an approach based on time-frequency analysis, specifically designed to extract pattern information from non-stationary signals. In what follows, we begin with an introduction to forecasting financial data, followed by a discussion on classifying EEG signals.

1.1 Forecasting financial data

In the financial market, there are many different kinds of financial assets. For example, stocks, securities, commodities, derivatives, and ETFs (Exchange-Traded Fund that is a popular investment vehicle that pools money from investors to buy a basket of securities like stocks, bonds, or commodities having a common underlying category like food or internet). There are also indices of the market that can represent overall market change, like $S\&P$ 500 and Dow Jones. One indicator that is crucial to the traders in the market is the sign change of a financial signal. A change from + to − indicates price falling, which is a signal of selling this financial asset in hand; an opposite sign change represents beginning of increase of the price, which is a buying sign for the traders. One common prediction standard in the quantitative trading field is to predict increase/decrease of a given price signal at the next check point, which may be the next data point, or n data points forward from the current time. Therefore, in this project, for each financial asset, we compute the sign of price change of a given period (controlled by parameter n = prediction_step) with 1 being an increase in stock price, and 0 being a decrease

$$y_t = \begin{cases} 1, & x_{t+n} - x_t > 0 \\ 0, & x_{t+n} - x_t \leq 0, \end{cases}$$

and use this as the standard to calculate accuracy rate of our models on price signal prediction.

We sample from three major categories of financial assets: individual stocks, exchange-traded funds (ETFs), and market indices. To ensure diversity, we select assets from various industries and sectors, aiming to assess model performance across different market conditions. Among the stocks, we include both well-established companies and high-growth startups. Established firms typically exhibit more stable price movements, while startup stocks tend to be significantly more volatile. Similarly, ETFs and

market indices are generally expected to follow smoother and more predictable trends. Based on these characteristics, we anticipate higher prediction win rates for ETFs and indices compared to startup stocks.
Below are the names of all financial signals we sample, and their full name for the reader to get a sense of which field it belongs to:

- **Indices:** SPY (Standard & Poor 500), QQQ (Invesco QQQ Trust), DIA (Dow Jones Industrial Average)
- **ETFs:** VTI (Vanguard Total Stock Market ETF), IWM (iShares Russell 2000 ETF), XLK (Technology Select Sector SPDR Fund), ARKK (ARK Innovation ETF), SMH (VanEck Semiconductor ETF), XLF (Financial Select Sector SPDR Fund), XLV (Health Care Select Sector SPDR Fund)
- **Stocks:**
  - Big companies: AAPL (Apple Inc.), MSFT (Microsoft Corporation), JPM (JPMorgan Chase & Co.), GOOGL (Alphabet Inc.), AMZN (Amazon.com, Inc.), NVDA (NVIDIA Corporation), UNH (UnitedHealth Group Incorporated), HD (The Home Depot, Inc.)
  - Startups: PLTR (Palantir Technologies Inc.), RIVN (Rivian Automotive, Inc.), UBER (Uber Technologies, Inc.), SNOW (Snowflake Inc.), COIN (Coinbase Global, Inc.), ROKU (Roku, Inc.), AFRM (Affirm Holdings, Inc.), DKNG (DraftKings Inc.)

Each dataset includes adjusted close prices over a half-hour window for 60 days. Since the stock market only open 6.5 hours a day (to public), each price signal contains $13 \times 60 = 780$ data points. We calculate the win rate, the accuracy score, of each financial signal by

$$\text{Win rate}(\{y_t\}) = \frac{\Sigma y_{tpred} * y_{ttrue}}{|y_t|} \tag{1}$$

1.2 Classifying EEG signals

Electroencephalography (EEG) is a non-invasive technique for recording the brain's electrical activity via electrodes placed on the scalp. Developed by Hans Berger in 1924, EEG captures voltage fluctuations generated by synchronized neuronal firing. While it offers a safe and accessible way to monitor brain function, its spatial resolution is limited due to signal distortion from the skull and surrounding tissues, and it cannot reach deeper brain structures without invasive methods.

Despite these limitations, EEG provides valuable insights through neural oscillations, or brain waves, which reflect different mental and physiological states. Beta waves are associated with focused attention and motor control and have been linked to disorders like Parkinson's disease. Alpha waves emerge during calm wakefulness and have shown connections to creativity and emotional regulation. At the opposite end, delta waves (0.5–4 Hz) dominate deep non-REM sleep and are crucial for tissue repair, hormone balance, and memory consolidation. Disruptions in delta activity can lead to poor sleep, fatigue, and cognitive issues, and abnormal delta presence during wakefulness may indicate neurological pathology.

Accurate classification of EEG signals is vital for diagnosing conditions such as epilepsy and other neurological disorders. In this project, we propose a binary classification method combining wavelet packet transform and convolutional neural networks (CNNs), aiming to enhance the interpretability and performance of EEG-based diagnostic tools.

Specifically, WPT extends the traditional wavelet transform by allowing both high- and low-frequency coefficients to be further decomposed, resulting in a richer and more flexible time-frequency representation. This approach constructs a complete binary tree of subspaces, enabling finer analysis of high-frequency components compared to standard wavelets. Motivated by the need to capture subtle structures in signals—especially those with complex frequency content—wavelet packets offer improved adaptability and resolution. Their advantages include better energy compaction, customizable basis selection, reduced computational cost, and enhanced performance in applications such as feature extraction for classification tasks. We will demonstrate these advantages numerically in Section 4.2.

The rest of the project is organized as follows. Section 2 reviews the mathematical background of the wavelet packet transform and the PWVD. Section 3 introduces the two procedures we developed.

Section 4 presents the real data analysis by using our proposed methods. Note that, for the reader's convenience, all tables are presented at the end of the project due to their length.

2 Background for PWVD and WPT

2.1 PWVD

The forecasting method employed in this study is grounded in time-frequency analysis, particularly through the use of the **Pseudo Wigner–Ville Distribution (PWVD)**, a well-established technique in non-stationary signal processing. This section situates the proposed method in the context of existing literature and explains the formulation and computational implementation of the approach.

Traditional time series models such as ARIMA, GARCH, or Kalman filtering often assume stationarity or apply linear projections that may fail to capture *frequency-domain behaviors* in asset prices [Tsay, 2005]. With the increased availability of high-frequency data, *non-stationary spectral methods* have gained interest in financial research. Among these, the Wigner–Ville Distribution (WVD) stands out as a quadratic time-frequency representation capable of revealing instantaneous frequency content of signals with high resolution [Cohen, 1989].

$$\mathrm{WVD}(t,f) = \int_{-\infty}^{\infty} x\left(t+\frac{\tau}{2}\right) x^*\left(t-\frac{\tau}{2}\right) e^{-2\pi i f\tau}\, d\tau \tag{2}$$

However, WVD suffers from severe cross-term interference when applied to multi-component signals. The *Pseudo Wigner–Ville Distribution (PWVD)* addresses this issue by introducing a smoothing kernel $h(\tau)$, typically Gaussian or Hamming, to reduce spurious components while preserving concentration in the time-frequency domain [Boashash, 1992a]. The PWVD is defined as:

$$\mathrm{PWVD}(t,f) = \int_{-\infty}^{\infty} h(\tau)\, x(t+\tau)\, x(t-\tau)^*\, e^{-2\pi i f\tau}\, d\tau \tag{3}$$

with $x(t)$ being the input signal. In this project, we choose the Hamming signal for its better localization performance. The Hamming kernel is defined as

$$h(\tau) = \begin{cases} 0.54 + 0.46\, cos(\frac{2\pi\tau}{N}), & \tau \le N/2 \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

where N is length of the signal. Lastly, we have $e^{-2\pi i f\tau}$ applies a Fourier transform in lag τ, converting the autocorrelation into frequency information.

This form retains the quadratic nature of the WVD but enhances interpretability, making it particularly well-suited for financial signals where overlapping patterns are common [Boashash, 1992b].

2.2 WPT

In the literature, the wavelet packet is defined by the equations as follows (see [Wickerhauser, 1991] for more detail)

$$u_{2k}^{(j)}(t) = \sqrt{2}\sum_m h(m) u_k^{(j)}(2t-m), \tag{5}$$

and

$$u_{2k+1}^{(j)}(t) = \sqrt{2}\sum_m g(m) u_k^{(j)}(2t-m). \tag{6}$$

Here, $n = 0, 1, 2, \ldots$, and $k = 0, 1, \ldots$. Moreover, $u_0^{(0)}(t)$ denotes the scaling function $\phi(t)$, and $u_1^{(0)}(t)$ denotes the base wavelet function $\psi(t)$. The superscript (j) in equations (5) and (6) indicates the j-th level of the wavelet packet basis, with 2^j wavelet packet bases present at level j. $h(m)$ and $g(m)$ in (5) and (6) are coefficients for the wavelet. For instance, if we use Haar wavelet in WPT, we have $h(0) = h(1) = 1/\sqrt{2}$, $h(m) = 0, m > 1$, and $g(0) = g(1) = -1/\sqrt{2}$, $g(m) = 0, m > 1$ (see [Daubechies, 1992] for more detail).

We introduce 2 properties of WPT. For simplicity, we do not provide the proof. The first is shift orthogonality. If $\{u_n^{(j)}(t)\}_{n\in\mathbb{Z}}$ is the set of wavelet packet bases obtained from the scaling function $u_0^{(0)}(t) = \phi(t)$ of an orthogonal base wavelet, then these bases hold the property of shift orthogonality:

$$\left\langle u_n^j(t),\, u_n^j(t-k)\right\rangle = \delta_k, \quad k \in \mathbb{Z}. \tag{7}$$

where $\langle\cdot\rangle$ denotes the inner product operation. The symbol δ_k represents a Dirac function. The second property is orthogonal relationship between $u_{2n}^{(j)}(t)$ and $u_{2n+1}^{(j)}(t)$ as follows

$$\left\langle u_{2n}^{(j)}(t),\, u_{2n+1}^{(j)}(t)\right\rangle = 0. \tag{8}$$

Once we have the definitions in (5) and (6), a recursive algorithm can be designed to implement the wavelet packet transform (WPT) for signal decomposition. For the reader's convenience, we refer to [Mallat, 1999].

3 Methods construction

In this section, we introduce two model construction procedures along with their corresponding classification methods. We begin with the PWVD method combined with a random forest classifier, follow by the WPT method using a CNN classifier.

3.1 PWVD and RandomForest Classifier

In Section 2.1, equation (3) presents a continuous-time formulation of the PWVD. However, in practice, it is impossible to work with signals of infinite length. Therefore, in actual implementations, we adopt the discrete version of the PWVD:

$$\mathrm{PWVD}[t,f] = \sum_{m=-M}^{M} h[m]\cdot x[t+m]\cdot x^*[t-m]\cdot e^{-\frac{2\pi i f m}{N}} \tag{9}$$

where $h[m]$ is the symmetric hamming kernel function, $x[t]$ is the input signal, N is number of frequency bins, which is defaulted to equal length of the input signal, and $m \in [-M, M]$ is the effective lag range that smooths the resulting time-frequency distribution, where M is chosen to be 32. Thus, for each input price signal of length N, we obtain a matrix in $\mathbb{R}^{N\times N}$, where the rows represent frequency bins and the columns correspond to time steps. In the following sections, we describe in detail how we use PWVD matrix to achieve signal prediction.

3.1.1 Moving Window Framework

To apply PWVD effectively in forecasting tasks, we adopt a *moving window approach*. Let n denote the number of steps ahead we want to forecast. That is, given the current time t, we aim to predict the sign change in the price of the financial asset at time $t+n$. Define $\{x_t\}_{t=1}^T$ to be the price signal. We divides $\{x_t\}_{t=1}^T$ into overlapping windows of length $K = r\cdot \mathrm{len}(\{x_t\})$, where $r :=$ window ratio as one input of our algorithm. Each window starting at time t and ending at $t+K$, shifting forward by one timestep. That is, we will have a list of moving windows $\{\{x_t\}_{t=1}^K, \{x_t\}_{t=2}^{T+1}, \ldots, \{x_t\}_{t=T-K}^T\}$.

3.1.2 Random Forest Classifier

We adopt the Random Forest (RF) classifier to predict the sign of price movement based on the PWVD matrix computed over each moving window. Given the complexity and non-stationarity of financial signals, RF offers a compelling balance between interpretability, robustness, and predictive performance.

For each windowed segment of the price signal, we compute the corresponding PWVD matrix—a time-frequency representation encoding the signal's local energy distribution. This matrix is then used as input to the Random Forest model, which outputs a binary prediction indicating whether the price will increase or decrease after a fixed forecasting horizon.

Our choice of Random Forest stems from a series of failed or suboptimal attempts with alternative approaches:

Failure of Inverse PWVD Reconstruction. One natural but ultimately infeasible idea was to directly predict the rows of the PWVD matrix and reconstruct the original signal via inverse transformation. However, unlike the classical Wigner–Ville Distribution, which maintains bilinear structure and phase coherence, the PWVD applies smoothing kernels (e.g., Hamming windows) that distort phase information in the time-lag domain. This smoothing process results in the loss of crucial high-frequency phase details, making reconstruction mathematically ill-posed. Specifically, the PWVD only preserves second-order moments of the form $x(t+\tau/2)^* x(t-\tau/2)$, which are insufficient to recover the full signal.

Lack of Visually Discernible Patterns. Visual inspection of the PWVD matrices reveals no clear patterns or structures that can be readily translated into handcrafted features. As shown in Figure 1, the matrices resemble abstract textures rather than interpretable spectrograms, motivating the need for machine learning-based feature extraction.

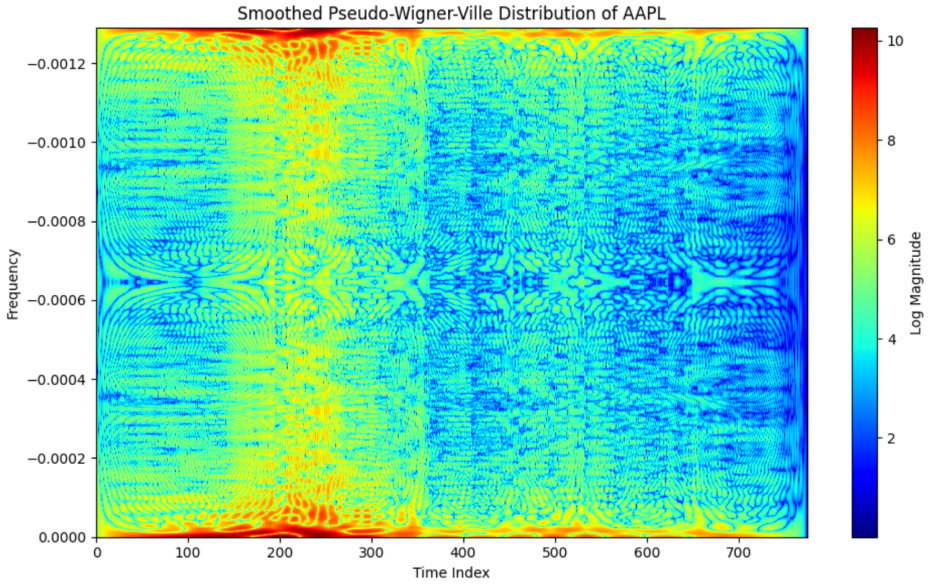


Figure 1: PWVD matrix of Apple's stock price signal. No obvious visual features can be manually extracted.

Why Random Forest? Given these limitations, we turn to Random Forest—a robust ensemble learning method capable of handling high-dimensional, noisy input data such as PWVD matrices. It excels in generalization, mitigates overfitting through feature bagging and tree averaging, and offers partial interpretability via feature importance scores. Moreover, Random Forest classifiers require minimal preprocessing and are less sensitive to hyperparameter tuning, making them well-suited for our large and heterogeneous financial dataset.

In conclusion, our final pipeline integrates PWVD for localized time-frequency decomposition and Random Forest for supervised classification. For each window of the signal, the model intakes the corresponding PWVD matrix and outputs a binary prediction indicating the future direction of price change.

3.2 Binary classification procedure

3.2.1 Feature construction based on WPT

In this subsection, we present our classification procedure based on the Wavelet Packet Transform (WPT) and Convolutional Neural Network (CNN). Let $s(t)$ denote the non-stationary signal, $g(t)$ the low-pass (scaling) filter, and $h(t)$ the high-pass (wavelet) filter. In this project, we choose Daubechies wavelet. Given a fixed positive integer J, the WPT recursively decomposes both the low-frequency and

Algorithm 1 Binary classification procedure.

Input: The training time series $\{\boldsymbol{y}_{k_1}\}_{1\le k_1\le N_1}$ and $\{\boldsymbol{z}_{k_2}\}_{1\le k_2\le N_2}$ with known labels, and the testing time series $\boldsymbol{s}$ with unknown class, integer J, parameters for CNN, and the type of wavelet.

Step 1: For the training time series $\{\boldsymbol{y}_{k_1}\}_{1\le k_1\le N_1}$ and $\{\boldsymbol{z}_{k_2}\}_{1\le k_2\le N_2}$, compute time-frequency coefficient matrix $\Phi^y_{k_1}(\cdot,\cdot)$ and $\Phi^z_{k_2}(\cdot,\cdot)$ based on (10) and (11).

Step 2: Train the CNN model based on $\Phi^y_{k_1}(\cdot,\cdot)$ and $\Phi^z_{k_2}(\cdot,\cdot)$.

Step 3: For the testing time series $\boldsymbol{s}$, compute time-frequency coefficient matrix $\Phi_s(\cdot,\cdot)$ by using WPT.

Step 4: Apply $\Phi_s(\cdot,\cdot)$ to the trained CNN model to classify the class of $\boldsymbol{s}$.

high-frequency components at each level $1 \le j \le J-1$. Moreover, let $x_k^{(j)}(t)$ represent the signal at node k and level j, with the initial condition $x_0^{(0)}(t) = s(t)$. Then, at each level j, each node $x_k^{(j)}$ is decomposed into two subnodes at level $j+1$ as follows

$$x_{2k}^{(j+1)}(t) = \sum_m g(m) \cdot x_k^{(j)}(2t-m), \tag{10}$$

and

$$x_{2k+1}^{(j+1)}(t) = \sum_m h(m) \cdot x_k^{(j)}(2t-m). \tag{11}$$

This decomposition continues recursively until $j = J-1$, producing a full binary tree of filtered and downsampled signals. Subsequently, at level j, there are 2^j nodes, i.e., when $j=0$, we have $x_0^{(0)}$. When $j=1$, we have $x_0^{(1)}, x_1^{(1)}$, finally, when $j=2$, we get $x_0^{(2)}, x_1^{(2)}, x_2^{(2)}, x_3^{(2)}$ and so on. Each node corresponds to a specific frequency band, offering a time-frequency resolution. Finally, we collect all nodes from each level of decomposition and take their absolute values to construct a two-dimensional, real-valued time-frequency coefficient matrix, denoted as $\Phi(\cdot,\cdot)$. Interpreting $\Phi(\cdot,\cdot)$ as an "image", it is natural to apply a convolutional neural network (CNN) for classification, leveraging convolutional filters to extract hierarchical features from the time-frequency "image". For details on the CNN process, we refer the reader to [O'shea and Nash, 2015] for simplicity.

3.2.2 Classification procedure using CNN

We now proceed to introduce the classification procedure. Consider two classes of signals: $\boldsymbol{y}_{k_1}$, where $1 \le k_1 \le N_1$ from class 1, consisting of N_1 subjects, where the k_1-th signal is $\boldsymbol{y}_{k_1} = (y_{k_1,i})_{i=1}^n \in \mathbb{R}^n$; and $\boldsymbol{z}_{k_2}$, where $1 \le k_2 \le N_2$ from class 2, consisting of N_2 subjects, where the k_2-th signal is $\boldsymbol{z}_{k_2} = (z_{k_2,i})_{i=1}^n \in \mathbb{R}^n$. We assume that all signals have the same length n. Moreover, the numbers of subjects in the two classes, N_1 and N_2, need not be large or balanced. This setting is well suited to CNN-based classification, which can effectively extract features even from limited and imbalanced data. Finally, we assume that both $\boldsymbol{y}_{k_1}$ and $\boldsymbol{z}_{k_2}$ are generated from non-stationary signal processes. Recall equations (10) and (11). Given a fixed integer J for both classes, we apply the Wavelet Packet Transform (WPT) to each signal, resulting in a time-frequency coefficient matrix for each one, denoted as $\Phi^y_{k_1}(\cdot,\cdot)$ and $\Phi^z_{k_2}(\cdot,\cdot)$ for two classes respectively. We then apply the CNN model to these matrices to train a classifier for the classification task. The overall procedure is summarized in Algorithm 1.

4 Real data analysis

In this section, we present the application of Wavelet Packet Transform (WPT) and Pseudo Wigner–Ville Distribution (PWVD) to two distinct types of time-series data: financial price signals and EEG recordings. For the financial dataset, we combine each time-frequency transform of a window of price signals with a Random Forest classifier to predict the direction of future price changes. For the EEG dataset, we apply both WPT and PWVD in conjunction with Convolutional Neural Networks (CNNs) to classify abnormal versus healthy brain activity. In addition to analyzing each method's performance within its respective task, we conduct a cross-comparison of PWVD and WPT across both domains to assess their relative strengths and generalizability. This section aims to provide a comprehensive

evaluation of how well each time-frequency representation supports downstream learning tasks in financial forecasting and biomedical signal classification.

4.1 Financial Data Analysis

In this section, we evaluate the forecasting performance of two time-frequency methods—Pseudo Wigner–Ville Distribution (PWVD) and Wavelet Packet Transform (WPT)—on a diverse collection of financial assets, as introduced in Section 1.1. We will use 55% as the standard qualifying win rate to evaluate our results. Our objective is twofold: (1) compare model performance across different types of financial assets, and (2) compare the predictive accuracy of the PWVD-based and WPT-based frameworks.

4.1.1 Performance Across Asset Categories

To understand the influence of asset type on forecasting performance, we categorize the financial instruments into three groups: ETFs, indices, and stocks. Within stocks, we further differentiate between large-cap (e.g., AAPL, MSFT) and startup firms (e.g., PLTR, COIN). Table 5 presents the mean win rates for each asset under both PWVD and WPT, while Table 6 summarizes the corresponding maximum win rates.

Across the board, established companies and major ETFs exhibit higher and more stable win rates than startups, consistent with the hypothesis that assets with lower volatility are easier to model and predict. For instance, Microsoft (MSFT) achieved a mean win rate of 0.7261 under PWVD and 0.7155 under WPT, with a maximum win rate of 1.0 for both methods. In contrast, startup stocks like Palantir (PLTR) and Coinbase (COIN) show greater volatility and consequently lower average predictability. PLTR has mean win rates of 0.6141 (PWVD) and 0.5724 (WPT), while COIN scores even lower at 0.5769 and 0.5498 respectively.

Looking at ETF performance, SMH (VanEck Semiconductor ETF) and XLK (Technology Select Sector SPDR Fund) post relatively higher win rates—around 0.62 to 0.65—suggesting that technology-related funds maintain enough structural stability for useful prediction. Market indices such as SPY and QQQ show lower performance, with QQQ reaching a mean win rate of 0.6291 (PWVD) and 0.6184 (WPT), which is unexpected. One plausible reason might be the indices is a sum of all market forces. They contain too many forces such that the algorithm cannot decompose successfully under the given resolution (# freq_bins = 780) Therefore, it is closest to pure random walk.

The observed trend confirms that prediction accuracy is positively correlated with market maturity and liquidity of the asset. ETFs and indices, which represent diversified or averaged signals, tend to filter out idiosyncratic noise and provide more predictable trends.

Table 5 lists mean win rates across different financial assets. Applying the 55% threshold, we observe distinct patterns across asset classes:

- Large-cap stocks: Most qualify. MSFT (72.61%), GOOGL (62.37%), AMZN (63.71%), HD (62.82%), and NVDA (61.72%) exceed the threshold using PWVD, with similar performance under WPT. These stocks are more stable and exhibit clearer structural trends, aiding prediction.
- Startups: Mixed results. Assets like AFRM (69.4%) and PLTR (61.41%) qualify under PWVD, but others like SNOW (57.58%) and COIN (57.69%) barely meet the threshold. The marginal ones show volatile behavior; while occasionally predictable, their high variance nature reduces average accuracy.
- ETFs and Indices: Broadly consistent. ETFs like XLK (64.72% & 62.62%), SMH (62.69% & 62.19%), and ARKK (60.31% & 60.68%) qualify, while others like XLF (59.26% & 56.41%) and XLV (59.8% & 56.36%) just pass. Among indices, SPY (60.5%) and QQQ (62.91%) qualify, while DIA (57.83%) is marginal. These aggregate instruments tend to be less noisy and reflect broader market trends.
- Failing assets: A few assets—particularly under WPT—fail to reach the 55% mark. Examples include NVDA (54.47%) and COIN (54.98%) under WPT, suggesting that WPT sometimes under-represents fine frequency modulations essential in volatile assets.

In summary, most large-cap stocks and diversified instruments meet or exceed the 55% benchmark, while a subset of volatile startup stocks fall short, especially under WPT.

4.1.2 PWVD vs. WPT: A Comparative Evaluation

Next, we compare the overall predictive performance of the PWVD and WPT models. According to summary Tables 3 and 4, the average win rate across all assets and parameter settings is 0.6152 for PWVD and 0.5995 for WPT. Although the margin is modest, PWVD demonstrates superior performance on average. Furthermore, PWVD's maximum win rate reaches 1.0 for multiple assets such as MSFT, UBER, COIN, and AFRM, indicating its ability to capture signal structures under ideal parameter settings. WPT, while also achieving a perfect score for MSFT and AFRM, tends to produce lower maximums on average. Upon examining the actual prediction sequences for MSFT, UBER, COIN, and AFRM, we observe that the true price direction signals for MSFT and AFRM consist entirely of upward movements (i.e., sequences of 1's). This suggests that both PWVD and WPT models are capable of accurately forecasting persistent bullish trends or sustained price increases. In contrast, the true sequences for UBER and COIN exhibit non-monotonic behavior, involving alternating signs. In these more complex, less predictable scenarios, the PWVD model demonstrates superior accuracy, indicating its enhanced ability to capture intricate temporal and spectral patterns within non-stationary financial signals.

Another critical difference lies in computational efficiency. The mean computation time for PWVD predictions is approximately 229.57 seconds, with high variance across parameter sets, whereas WPT only requires an average of 7.01 seconds. This substantial speed advantage makes WPT attractive for real-time applications or large-scale datasets, though at a slight cost in predictive power. If the quantitative trader prefers train and save parameter type of model, then PWVD is better given its high win rates. If the practitioners favor less computation cost, then they should choose WPT given its fast computational speed.

The standard deviations in win rates further reveal model behavior: PWVD has a slightly higher standard deviation (0.0473) than WPT (0.0461), suggesting that PWVD is more sensitive to parameter tuning. In practice, this makes PWVD potentially more powerful but also more brittle—demanding more careful hyperparameter selection.

Lastly, asset-level comparison also shows that PWVD often surpasses WPT on volatile assets. For example, NVDA achieves a mean win rate of 0.6172 under PWVD compared to just 0.5447 under WPT, and PLTR follows a similar trend. This indicates that PWVD, with its finer time-frequency resolution, may be better suited to detect transient, high-frequency structures in volatile price signals.

In summary, the financial data analysis illustrates two key takeaways. First, prediction performance is strongly tied to the type and volatility of financial assets: large-cap stocks and ETFs consistently outperform startup stocks. Second, the PWVD-based method shows marginally superior forecasting performance compared to WPT, particularly for high-volatility assets, albeit with significantly higher computational cost. Depending on the intended application—whether accuracy or speed is prioritized—either model may be preferred.

4.1.3 Hyperparameter Sensitivity Analysis

To further understand model behavior, we analyze how different combinations of hyperparameters influence predictive performance in both the PWVD and WPT frameworks. Tables 1 and 2 report win rate statistics over a grid of parameter triplets of the form (n, r, α), where:

- n: prediction step (number of time steps into the future),
- r: window ratio (proportion of the signal used in each sliding window),
- α: test ratio (fraction of data reserved for testing).

We highlight several key patterns from the parameter sweeps:

1. Prediction Step (n) Effects. As n increases from 2 to 12, the mean win rates generally improve for both methods, particularly when r and α are fixed at high values. For example, in PWVD, mean win rates increase from around 0.54 at $n = 2$ to a peak of 0.7473 at $n = 12$, $r = 0.9$, $\alpha = 0.2$. A similar trend is observed in WPT, where the mean win rate rises to 0.7253 under the same setting. This suggests that both models benefit from forecasting further into the future—possibly due to smoothing over short-term noise—but at the cost of increased variance.

2. Window Ratio (r) Effects. Higher window ratios consistently yield better mean win rates. In both PWVD and WPT, moving from $r = 0.7$ to $r = 0.9$ often produces a noticeable increase in accuracy. This is intuitive: a larger window provides more historical context for each prediction, allowing the model to capture long-range temporal dependencies and spectral patterns more effectively. For instance, in PWVD, the mean win rate for $n = 8$ increases from 0.5735 at $r = 0.7$, $\alpha = 0.4$ to 0.7363 at $r = 0.9$, $\alpha = 0.2$. Similarly, in WPT, we see a mean win rate increase from 0.5583 to 0.6923 over the same settings.

3. Test Ratio (α) Effects. The test ratio α plays a critical role in determining model performance by directly controlling the amount of training data available. A lower α implies a larger training set, which enables more robust model learning and often leads to significantly improved predictive accuracy. This effect is clearly visible across both PWVD and WPT results. For instance, with parameters $n = 10$, $r = 0.9$, the PWVD method achieves a mean win rate of 0.7143 at $\alpha = 0.2$, but this value drops to 0.6332 at $\alpha = 0.4$. Similarly, WPT reaches 0.7005 at $\alpha = 0.2$ but only 0.6277 at $\alpha = 0.4$. These results highlight that increasing the test ratio can substantially degrade performance, underscoring the importance of preserving a sufficiently large training window in time-frequency forecasting tasks.

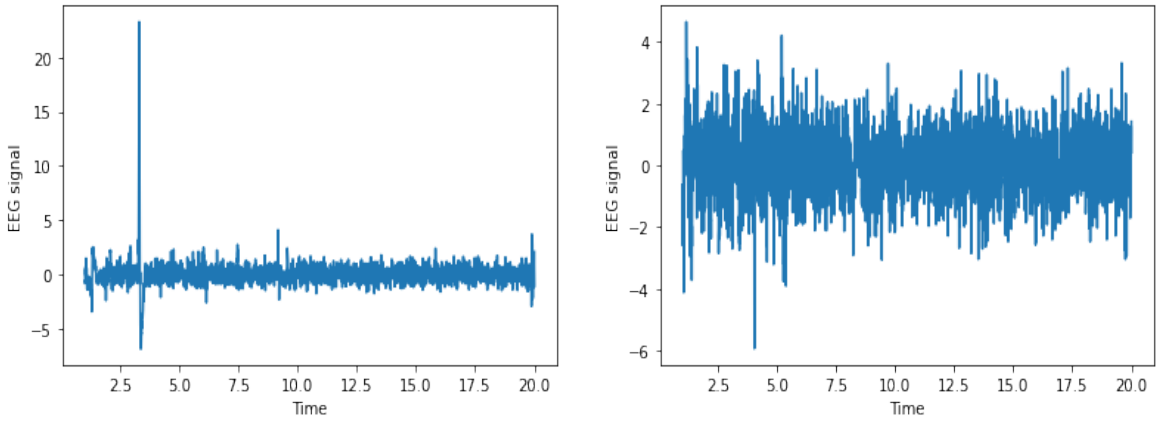
Win rates for both PWVD and WPT models depend significantly on careful hyperparameter tuning. Larger prediction steps and higher window ratios consistently enhance performance, especially when paired with ample training data. PWVD shows more resilience across parameter choices, with a broader range of effective settings. WPT requires more careful selection, but can still exceed PWVD under well-chosen parameters, with lower computational cost.

To further understand model behavior, we analyze how different combinations of hyperparameters influence predictive performance in both the PWVD and WPT frameworks. We adopt a 55% win rate threshold as the industry-accepted standard for meaningful prediction. Any configuration achieving above this benchmark is considered **qualifying**, while those below are considered **non-qualifying**.

PWVD Findings. For the PWVD-based forecasting model, performance is strongly influenced by the choice of hyperparameters, particularly the prediction step n and the window ratio r. At lower values of n, such as $n = 2$, the model frequently fails to exceed the industry-standard 55% win rate threshold. For example $(2, 0.7, 0.2)$, which yields a mean win rate of only 53.68%. The likely cause is that short-term predictions are more vulnerable to market noise and micro-structural fluctuations that obscure informative patterns in the time-frequency domain. As the prediction horizon increases, the model has more opportunity to capture underlying trends rather than high-frequency noise, resulting in substantial improvements. Notably, for $n \geq 6$ and $r \geq 0.8$, most parameter configurations consistently surpass the 55% benchmark. The best-performing PWVD setup—$(12, 0.9, 0.2)$—achieves a mean win rate of 74.73%, reflecting the model's strength in extracting long-term spectral structures when given sufficient historical context and training data. We examined this hyperparameter pair and discovered this is the pair that MSFT, UBER, COIN, and AFRM achieve 100% win rates. In future, if we get to work on this project again, we consider tuning default parameters of the PWVD function like resolution and lag range M to see if win rate can be further enhanced.

WPT Findings. The WPT model exhibits similar sensitivity to hyperparameter settings, but its performance landscape is more irregular. At smaller prediction steps and narrower windows, such as $(2, 0.7, 0.3)$, the WPT model fails to reach the qualifying threshold, with win rates as low as 53.19%. This underperformance is attributed to WPT's hierarchical decomposition being less effective at resolving subtle transient features in short windows or short horizons. However, when provided with richer temporal context and more training data—specifically, for $n \geq 6$, $r \geq 0.9$, and $\alpha \leq 0.3$—the model achieves substantially better accuracy. For instance, the configuration $(10, 0.9, 0.2)$ yields a win rate of 70.05%, and the overall best result of 72.53% is achieved under $(12, 0.9, 0.2)$. These findings suggest that while WPT can rival or even surpass PWVD in ideal settings, it requires more careful parameter tuning to extract discriminative features effectively from non-stationary financial signals.

Conclusion PWVD exhibits greater robustness across the parameter space, with a broader range of qualifying settings. In contrast, WPT can achieve comparable or even superior performance under specific conditions, but requires more precise tuning. Therefore, while both methods benefit from



(a) (b)

Figure 2: Original EEG signals (a) abnormal; (b) normal.

increased prediction horizon and larger window sizes, PWVD demonstrates stronger consistency in crossing the 55% win rate threshold across varying configurations.

4.2 EEG Data

4.2.1 Data exploratory analysis

In this subsection, we introduce the classifcation of Electroencephalography (EEG) based on electrogram of brain, using our proposed procedure as in Section 3.2. We use the TUH Abnormal EEG Corpus dataset [Obeid and Picone, 2016], which contains EEG records that are classified as clinically normal or abnormal. The dataset is part of the Temple University EEG (TUH EEG) database, available at https://isip.piconepress.com/projects/nedc/html/tuh_eeg/.

The dataset contains EEG signal recordings from different electrode positions for each subject, along with metadata and a label indicating whether the subject's EEG is normal or abnormal. In detail, the dataset consists of a training set with 1237 normal and 893 abnormal subjects, and an evaluation set with 148 normal and 105 abnormal subjects. Each subject's record includes 23 EEG signals collected from different electrode positions.

In our analysis, we focused on channel CZ as this channel is centrally positioned over the motor and somatosensory cortex, representing overall brain activity. To ensure consistency and simplicity, we selected 504 subject records, with half normal and half abnormal from the traning dataset. For testing dataset, we used 150 subject records, equally split between normal and abnormal. Moreover, to minimize artifacts, the first minute of each EEG signal recording was discarded, and up to 20 minutes of the remaining was used. Missing values were excluded, amplitude values were clipped to $\pm 800 \mu V$, and each signal was downsampled from 50 Hz to 1 Hz to speed up the computation. Subsequently, each signal was reduced to the length of 4500. Finally, we only used the signal for the classification and ignored the metadata.

After data preprocessing, Figure 2 displays two EEG signals, where (a) is the abnormal signal and (b) is the normal signal. In subplot (a), the abnormal EEG shows a significant spike around 2.5 minutes, followed by increased variability and intermittent high-amplitude activity—suggestive of irregular brain activity, possibly due to seizure or other abnormalities. In contrast, subplot (b) presents a normal EEG with consistent low-amplitude oscillations and minimal variability, indicating stable brain activity. Although a few small spikes appear around 4 minutes, they likely reflect typical stimuli such as yawning. Moreover, while not all abnormal signals resemble subplot (a), the comparison highlights key differences in amplitude and frequency variability between abnormal and normal EEG signals, which are crucial for diagnostic analysis.

4.2.2 Model fitting

We adopt the Python package pywt to apply WPT, tftb for PWVD, and keras for the CNN classifier. For the tuning parameters, we select the Daubechies 9 wavelet for WPT and fix the decomposition

level at 4. For the PWVD method, we use the default parameter settings. Regarding the CNN, we construct the architecture as follows: a Sequential model with three convolutional layers (with 32, 64, and 128 filters, respectively), each followed by ReLU activation and 2×2 max pooling, to extract hierarchical features from the input. The output is then flattened and passed through a dense layer with 128 neurons (ReLU), followed by a final dense layer with a sigmoid activation that outputs the probability of the abnormal class. The model is compiled using the Adam optimizer with binary cross-entropy loss. It is trained for 20 epochs using the WPT-transformed data and for 10 epochs using the PWVD-transformed data.

All parameters for WPT and CNN are chosen in a data-driven. Finally, the model is evaluated on the test set to report classification accuracy. In addition, due to computational limitations, the input signals for the PWVD method are downsampled to a length of 800.

4.2.3 Spectrogram and classification results

The figures 3 and 4 compare spectrograms by using WPT and PWVD for abnormal and normal EEG signals, highlighting their distinct characteristics. In Figure 4, WPT spectrograms provide fine-grained time-frequency localization, with the abnormal signal showing concentrated energy in lower subbands and transient peaks in the specific higher subband, while the normal signal exhibits a more uniform and stable distribution across lower subbands. In contrast, Figure 5 presents the PWVD spectrograms, which primarily capture the global energy distribution. For these 2 signals, We see a clear difference between the abnormal and normal class. However, they make it difficult to identify distinguishing characteristics of the abnormal signal.

The classification accuracy rate for WPT is 81.3%, while for PWVD, it is 62%. This result aligns with the spectrograms of the two methods, as the WPT spectrogram clearly highlights distinct abnormal patterns, particularly in localized frequency subbands. This superior performance of WPT suggests its effectiveness in detecting transient features and specific anomalies that are indicative of abnormal brain activity, which may be less discernible in the broader, more global patterns captured by PWVD. The localized nature of WPT enables better discrimination between normal and abnormal signals, thereby contributing to its higher accuracy. In addition, it is also shown that CNNs have the ability to extract meaningful information from the time-frequency matrix.

In conclusion, these two methods offer complementary insights for analyzing EEG signals, with WPT excelling in providing interpretable, localized details, making it more effective in detecting abnormal patterns. Moreover, WPT offers greater computational efficiency. Overall, this project motivates further exploration of time-frequency analysis techniques for signal classification. In future work, it would be worthwhile to apply additional methods to extract more informative features.

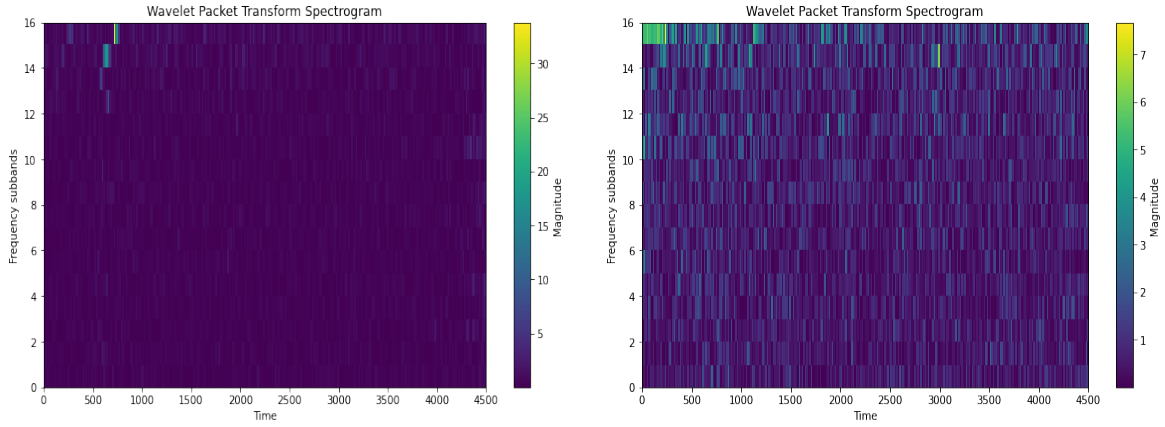
5 Conclusion

This study explores two time-frequency analysis techniques—Pseudo Wigner–Ville Distribution (PWVD) and Wavelet Packet Transform (WPT)—and their applications to financial signal forecasting and EEG-based brain signal classification.

In financial forecasting, PWVD combined with Random Forest classification achieves superior performance, especially for volatile assets. While computationally expensive, PWVD consistently meets or exceeds the 55% win rate threshold across diverse parameter settings. WPT, although faster and more computationally efficient, demonstrates sensitivity to tuning and occasionally underperforms on high-volatility signals.

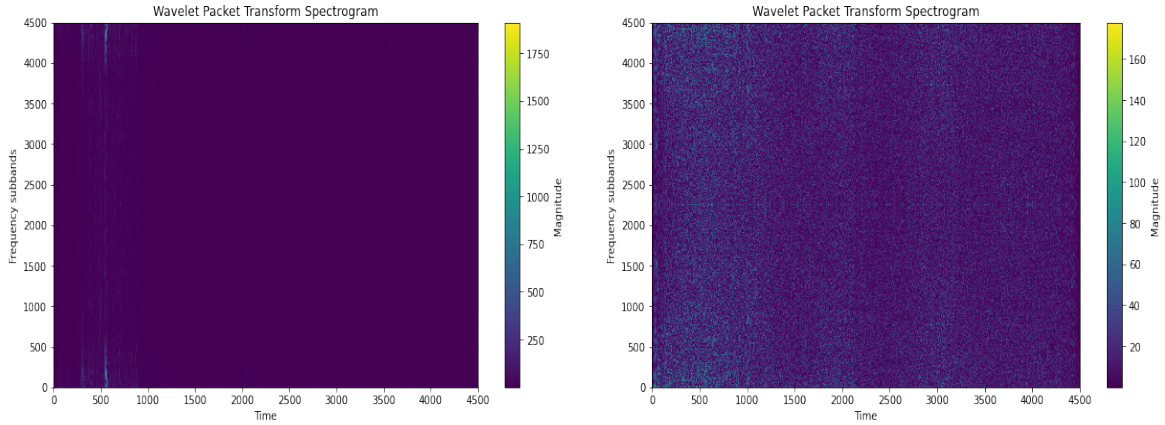
For EEG classification, both PWVD and WPT enable effective feature extraction for convolutional neural network (CNN) models. Despite PWVD's theoretical advantages in frequency resolution, WPT's recursive decomposition structure proves sufficient for high-accuracy classification of abnormal brain signals.

In conclusion, time-frequency methods, when paired with machine learning models, offer versatile and effective solutions for analyzing non-stationary signals. The PWVD excels in accuracy for complex, noisy data, while WPT remains a viable alternative for scenarios where computational cost is a constraint. Future research may integrate both approaches or employ adaptive neural architectures to further enhance model performance.



(a) (b)

Figure 3: Wavelet packet transform spectrogram for EEG signals (a) abnormal; (b) normal.



(a) (b)

Figure 4: PWVD spectrogram for EEG signals (a) abnormal; (b) normal.

References

[Boashash, 1992a] Boashash, B. (1992a). Estimating and interpreting the instantaneous frequency of a signal. i. fundamentals. *Proceedings of the IEEE*, 80(4):520–538.

[Boashash, 1992b] Boashash, B. (1992b). *Time-Frequency Signal Analysis: Methods and Applications*. Longman Cheshire.

[Cohen, 1989] Cohen, L. (1989). Time-frequency distributions—a review. *Proceedings of the IEEE*, 77(7):941–981.

[Daubechies, 1992] Daubechies, I. (1992). *Ten lectures on wavelets*. Society for industrial and applied mathematics.

[Mallat, 1999] Mallat, S. (1999). *A wavelet tour of signal processing*. Elsevier.

[Obeid and Picone, 2016] Obeid, I. and Picone, J. (2016). The temple university hospital EEG data corpus. *Frontiers in Neuroscience*, 10:196.

[O'shea and Nash, 2015] O'shea, K. and Nash, R. (2015). An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*.

[Tsay, 2005] Tsay, R. S. (2005). *Analysis of Financial Time Series*. John Wiley & Sons.

[Wickerhauser, 1991] Wickerhauser, M. (1991). Inria lectures on wavelet packet algorithms.

Parameters	Max	Min	Mean	Time Spent (s)
(2, 0.7, 0.2)	0.7447	0.4681	0.5368	302.8407
(2, 0.7, 0.3)	0.6714	0.4571	0.5467	267.9609
(2, 0.7, 0.4)	0.6452	0.4516	0.5467	273.7547
(2, 0.8, 0.2)	0.6452	0.4516	0.5695	261.7147
(2, 0.8, 0.3)	0.7234	0.4681	0.5655	242.8209
(2, 0.8, 0.4)	0.7097	0.4516	0.554	250.6778
(2, 0.9, 0.2)	0.8125	0.5	0.5986	159.9298
(2, 0.9, 0.3)	0.7826	0.4783	0.5569	140.806
(2, 0.9, 0.4)	0.8065	0.4516	0.5583	136.8724
(4, 0.7, 0.2)	0.7174	0.5	0.6012	345.9085
(4, 0.7, 0.3)	0.7101	0.4638	0.5702	333.176
(4, 0.7, 0.4)	0.6739	0.4674	0.5506	322.4642
(4, 0.8, 0.2)	0.8065	0.4516	0.598	252.1361
(4, 0.8, 0.3)	0.7174	0.4565	0.5811	245.379
(4, 0.8, 0.4)	0.6721	0.459	0.5706	235.7895
(4, 0.9, 0.2)	0.7333	0.4667	0.5897	137.8541
(4, 0.9, 0.3)	0.7391	0.4783	0.592	137.6699
(4, 0.9, 0.4)	0.7	0.4667	0.5885	132.7293
(6, 0.7, 0.2)	0.8261	0.4565	0.607	336.9004
(6, 0.7, 0.3)	0.7971	0.4638	0.612	317.869
(6, 0.7, 0.4)	0.8043	0.4783	0.5807	306.6577
(6, 0.8, 0.2)	0.8667	0.5	0.6205	246.1867
(6, 0.8, 0.3)	0.8222	0.4667	0.5983	237.6013
(6, 0.8, 0.4)	0.8	0.4667	0.5904	233.9873
(6, 0.9, 0.2)	0.8667	0.4667	0.6718	133.2364
(6, 0.9, 0.3)	0.8182	0.4545	0.6066	130.2299
(6, 0.9, 0.4)	0.8966	0.4828	0.618	128.4788
(8, 0.7, 0.2)	0.8478	0.4565	0.6304	326.2015
(8, 0.7, 0.3)	0.7794	0.4706	0.6012	315.7651
(8, 0.7, 0.4)	0.7033	0.4505	0.5735	300.5191
(8, 0.8, 0.2)	0.8667	0.4667	0.6372	241.7412
(8, 0.8, 0.3)	0.8	0.4889	0.6222	233.9543
(8, 0.8, 0.4)	0.8833	0.4833	0.6122	228.5533
(8, 0.9, 0.2)	1.0	0.5714	0.7363	128.7852
(8, 0.9, 0.3)	0.8095	0.5238	0.6429	127.4792
(8, 0.9, 0.4)	0.9286	0.5	0.6703	123.4698
(10, 0.7, 0.2)	0.8889	0.4667	0.6444	323.1122
(10, 0.7, 0.3)	0.7647	0.4559	0.6386	314.4298
(10, 0.7, 0.4)	0.7333	0.4778	0.5932	311.8251
(10, 0.8, 0.2)	0.8333	0.4667	0.6295	257.9262
(10, 0.8, 0.3)	0.8864	0.4545	0.6198	244.4259
(10, 0.8, 0.4)	0.9322	0.4576	0.6206	225.517
(10, 0.9, 0.2)	0.9286	0.5	0.7143	129.9995
(10, 0.9, 0.3)	0.9048	0.4762	0.663	119.6419
(10, 0.9, 0.4)	0.8929	0.4643	0.6332	115.4377
(12, 0.7, 0.2)	0.9556	0.4667	0.6692	332.5985
(12, 0.7, 0.3)	0.806	0.4925	0.6567	319.4869
(12, 0.7, 0.4)	0.7528	0.4607	0.5981	319.1325
(12, 0.8, 0.2)	0.9655	0.4828	0.6698	260.1915
(12, 0.8, 0.3)	0.9091	0.4545	0.6617	240.5781
(12, 0.8, 0.4)	0.931	0.4655	0.6094	227.2123
(12, 0.9, 0.2)	1.0	0.5	0.7473	126.9165
(12, 0.9, 0.3)	1.0	0.5	0.6942	121.7498
(12, 0.9, 0.4)	0.963	0.4815	0.651	128.2913

Table 1: PWVD results

Parameters	Max	Min	Mean	Time Spent (s)
(2, 0.7, 0.2)	0.6596	0.4681	0.5532	11.1711
(2, 0.7, 0.3)	0.6571	0.4571	0.5319	11.6109
(2, 0.7, 0.4)	0.6129	0.4731	0.5347	11.0296
(2, 0.8, 0.2)	0.6774	0.4516	0.5285	7.1881
(2, 0.8, 0.3)	0.6596	0.4681	0.5401	8.0942
(2, 0.8, 0.4)	0.6774	0.4516	0.5515	7.6993
(2, 0.9, 0.2)	0.6875	0.5	0.5817	4.9765
(2, 0.9, 0.3)	0.7826	0.4783	0.5903	4.7739
(2, 0.9, 0.4)	0.6774	0.4516	0.5434	5.4742
(4, 0.7, 0.2)	0.6957	0.4565	0.5543	11.4025
(4, 0.7, 0.3)	0.6522	0.4638	0.5491	10.5727
(4, 0.7, 0.4)	0.6196	0.4565	0.5364	8.6904
(4, 0.8, 0.2)	0.7097	0.4516	0.5707	7.6739
(4, 0.8, 0.3)	0.6957	0.4565	0.5644	6.2343
(4, 0.8, 0.4)	0.7213	0.4754	0.5649	6.5039
(4, 0.9, 0.2)	0.8667	0.4667	0.6128	5.1273
(4, 0.9, 0.3)	0.7391	0.4783	0.5702	4.8116
(4, 0.9, 0.4)	0.7333	0.4667	0.5718	4.844
(6, 0.7, 0.2)	0.8043	0.4565	0.5836	10.4022
(6, 0.7, 0.3)	0.7391	0.4638	0.5825	9.0089
(6, 0.7, 0.4)	0.6304	0.4674	0.5351	7.746
(6, 0.8, 0.2)	0.8	0.4667	0.6064	7.1753
(6, 0.8, 0.3)	0.8222	0.4667	0.6205	6.6041
(6, 0.8, 0.4)	0.6667	0.4667	0.5712	7.0529
(6, 0.9, 0.2)	0.9333	0.4667	0.6538	4.8523
(6, 0.9, 0.3)	0.8636	0.4545	0.6224	4.6703
(6, 0.9, 0.4)	0.8966	0.4828	0.5902	4.0836
(8, 0.7, 0.2)	0.8478	0.4565	0.6254	8.533
(8, 0.7, 0.3)	0.7647	0.4706	0.5894	9.02
(8, 0.7, 0.4)	0.7143	0.4505	0.5583	9.2915
(8, 0.8, 0.2)	0.9333	0.4667	0.6295	7.9247
(8, 0.8, 0.3)	0.8222	0.5111	0.6453	6.3535
(8, 0.8, 0.4)	0.7833	0.4667	0.5833	5.4402
(8, 0.9, 0.2)	0.8571	0.5	0.6923	5.0815
(8, 0.9, 0.3)	0.8571	0.4762	0.6575	4.3844
(8, 0.9, 0.4)	0.9643	0.4643	0.6291	5.1046
(10, 0.7, 0.2)	0.8889	0.4667	0.5855	8.8117
(10, 0.7, 0.3)	0.8088	0.4559	0.6097	8.776
(10, 0.7, 0.4)	0.6889	0.4778	0.5688	8.8616
(10, 0.8, 0.2)	0.8333	0.4667	0.6038	7.7011
(10, 0.8, 0.3)	0.9091	0.4773	0.6329	7.2455
(10, 0.8, 0.4)	0.8136	0.4746	0.6173	6.7365
(10, 0.9, 0.2)	0.8571	0.5	0.7005	4.6631
(10, 0.9, 0.3)	0.9048	0.4762	0.6374	6.302
(10, 0.9, 0.4)	0.9286	0.4643	0.6277	5.0293
(12, 0.7, 0.2)	0.9556	0.4889	0.6274	9.3876
(12, 0.7, 0.3)	0.7313	0.4925	0.6355	7.6394
(12, 0.7, 0.4)	0.7416	0.4607	0.5592	7.371
(12, 0.8, 0.2)	0.931	0.5172	0.6512	6.094
(12, 0.8, 0.3)	0.9318	0.4545	0.6285	5.7031
(12, 0.8, 0.4)	0.7931	0.4655	0.616	5.6346
(12, 0.9, 0.2)	1.0	0.5	0.7253	4.044
(12, 0.9, 0.3)	1.0	0.5	0.6615	4.0088
(12, 0.9, 0.4)	0.9259	0.4815	0.6567	3.7076

Table 2: WPT results

Statistic	Max (win rate)	Min (win rate)	Mean (win rate)	Time Used (s)
Maximum	1.0000	0.5714	0.7473	345.9085
Minimum	0.6452	0.4505	0.5368	115.4377
Mean	0.8181	0.4733	0.6152	229.5662
Std Dev	0.0973	0.0217	0.0473	77.6404

Table 3: PWVD data summary

Statistic	Max (win rate)	Min (win rate)	Mean (win rate)	Time Used (s)
Maximum	1.0000	0.5172	0.7253	11.6109
Minimum	0.6129	0.4505	0.5285	3.7076
Mean	0.7939	0.4712	0.5995	7.0060
Std Dev	0.1087	0.0161	0.0461	2.1330

Table 4: WPT mean win rate statistics

Ticker	**PWVD Mean**	**WPT Mean**
AAPL	0.5938	0.6
MSFT	0.7261	0.7155
JPM	0.599	0.5809
GOOGL	0.6237	0.6208
AMZN	0.6371	0.6397
NVDA	0.6172	0.5447
UNH	0.5737	0.579
HD	0.6282	0.5902
PLTR	0.6141	0.5724
RIVN	0.5966	0.5937
UBER	0.6397	0.6426
SNOW	0.5758	0.5845
COIN	0.5769	0.5498
ROKU	0.6062	0.5736
AFRM	0.694	0.6579
DKNG	0.6147	0.57
VTI	0.5965	0.5863
IWM	0.6016	0.6009
XLK	0.6472	0.6262
ARKK	0.6031	0.6068
SMH	0.6269	0.6219
XLF	0.5926	0.5641
XLV	0.598	0.5636
SPY	0.605	0.598
QQQ	0.6291	0.6184
DIA	0.5783	0.5846

Table 5: Comparison of mean win rates from PWVD and WPT methods

Ticker	PWVD Max	WPT Max
AAPL	0.8444	0.8667
MSFT	1.0000	1.0000
JPM	0.9286	0.9286
GOOGL	0.9630	0.9643
AMZN	0.8929	0.8929
NVDA	0.9286	0.6897
UNH	0.8571	0.8571
HD	0.8571	0.9333
PLTR	0.9655	0.6889
RIVN	0.7528	0.7333
UBER	1.0000	1.0000
SNOW	0.7500	0.7069
COIN	1.0000	0.7857
ROKU	0.8276	0.8148
AFRM	1.0000	1.0000
DKNG	0.9500	0.7857
VTI	0.7857	0.8571
IWM	0.9286	0.8571
XLK	0.8444	0.8222
ARKK	0.8571	0.8571
SMH	0.8444	0.8136
XLF	0.8571	0.9286
XLV	0.9500	0.8571
SPY	1.0000	0.7941
QQQ	0.8478	0.8478
DIA	0.7857	0.7619

Table 6: Comparison of max win rates from PWVD and WPT methods