Hidden Markov Models for Modeling Market States

Ben Hull https://bluehood.github.io/

Abstract

This study investigates the efficacy of a two-state Hidden Markov Model (HMM) in identifying 'risk-on' and 'risk-off' market regimes to enhance portfolio allocation. Utilising a longitudinal dataset of daily S&P 500 returns spanning from 1930 to the present, we characterise two distinct latent states: a low-volatility state associated with higher equity returns and a high-volatility state marked by diminished performance. Our analysis identifies Gold as a robust hedging instrument, maintaining consistent performance across both regimes. By integrating these insights into a Walk Forward Backtest, we constructed dynamic portfolios that significantly outperformed a traditional 'buy-and-hold' strategy on a risk-adjusted basis and in overall returns. We also apply tests to check that the superior performance is statistically significant and not just due to random noise. While the HMM-based strategy trailed equity benchmarks during sustained low-volatility periods, it demonstrated superior capital preservation during significant market downturns. These results suggest that while HMMs effectively capture shifting risk profiles, future performance could be further optimised by incorporating other volatility indicators, such as the VIX, provided that model complexity is managed to mitigate overfitting.

Keywords: Hidden Markov Models, Financial Modelling, Machine Learning, Data Science

1 Hidden Markov Models

In the field of statistical modelling, few frameworks are as versatile as the Hidden Markov Model (HMM). An HMM is a double-stochastic process used to model systems that evolve over time where the underlying mechanism is not directly observable. Instead, we only see a sequence of outputs – or 'emissions' – from which we can infer the probability of the system being in one of these hidden states.

The model assumes the system is a Markov process – that is, the probability of transitioning to the next state depends only on the current states and not any previous states. Mathematically, we can describe the probability of being in state S_t depends only on S_{t-1}:

$$P(S_t|S_{t-1}, S_{t-2}, ..., S_1) = P(S_t|S_{t-1})$$

We then introduce a hidden layer which represents the actual state of the system, but these states cannot be directly observed. The observation layer emits a set of observables from which we can infer the probability of being in a given hidden state. Each hidden state has a specific probability distribution over the possible observables it can produce.

1.1 An Example

A classic example is the weather. Lets assume we live in a different location can cannot actually observe the weather at another location. However, we can look at the clothes that our room mate, who travels to this location for work, is wearing and infer the state of the weather from this information. At any given time the weather can be in a given state that we cannot observe directly. However, we can observe the our room mates choice of clothes. From these observables we can infer which hidden state the weather is currently in based of these observables, e.g. Sunny, Rainy or Cloudy for instance.

In this model, the weather is the hidden variable. For modelling purposes we will classify the weather into three discrete states:

1. State 1: Sunny
2. State 2: Rainy
3. State 3: Cloudy

HMMs have transition probabilities, which are the probabilities of transitioning from one state to another (or staying in the same state). These transition probabilities are normally represented as a matrix. For example, if it is Sunny today, there might be an 80% chance it stays Sunny tomorrow and a 15% chance it becomes Rainy and a 5% chance of becoming Cloudy.

The emission probabilities are inferred from observable evidence. Each hidden weather state has a probability of 'emitting' a specific observation. For example, if it is Rainy then there is a high probabilities that our room mate leaves with a coat or umbrella. If it Sunny then there is a higher probability of our room mate wearing sunglasses.

1.2 Learning and the Baum-Welch Algorithm

HMMs typically use the Baum-Welch algorithm to fit observable data to a HMM model.

Baum-Welch is an Expectation-Maximization (EM) algorithm which are given a sequence of observations (like our roommate's umbrellas and sunglasses) and work backward to find the transition and emission probabilities that most likely produced that data.

When fitting data to a HMM we have a dilemma:

1. If we knew the hidden states (the actual weather), we could easily calculate the probabilities of each observable (Rain, Shine etc).
2. If we knew the probabilities, we could easily estimate the hidden states.

Since we know neither, Baum-Welch starts with a blind guess and refines it through two repeating steps.

The Expectation Step

In this phase, we act as if our current (initial or previous) model parameters are correct. We use the Forward-Backward Algorithm to calculate the probability of being in a specific state at a specific time.

1. **Forward Pass:** Calculates the probability of seeing the partial observation sequence up to time t and ending in state i.

2. **Backward Pass:** Calculates the probability of seeing the remaining observation sequence from time $t+1$ to the end, given we are in state i at time t.

By combining these, we get an 'expected count' of how many times a certain transition (e.g., Sunny $\rightarrow$ Rainy) or emission (e.g., Rainy $\rightarrow$ Umbrella) occurred in our data.

The Maximization Step

Now that we have our 'expected counts' from the Expectation Step, we update the model. We treat these expectations as the new 'truth' and re-calculate the probabilities to maximise the likelihood of the observations.

1. **Update Transitions:** If the Expectation step suggests that Sunny usually follows Sunny in our data, we increase the transition matrix value for $Sunny \rightarrow Sunny$.

2. **Update Emissions**: If 'Umbrella' was frequently associated with the "Rainy" state in our expectations, we increase the emission matrix value for $Rainy \rightarrow Umbrella$.

The algorithm repeats these two steps iteratively. With each cycle, the model's parameters (θ) are adjusted so that the probability of the observed sequence $P(O|\theta)$ increases.

2 Financial Modelling

We can apply HMMs in the context of the stock market. We can model the market as having a given internal hidden state with specific observables emitted for each hidden state. In this way we can construct a model that learns from the observables and state transitions to model the market hidden states.

Hidden Markov models tend to perform better with smaller observable spaces – that is, adding observable data into the training process increases the complexity of the HMM and can degrade the training process and can lead to over-fitting. (The complexity is roughly $O(N^2 \cdot T)$, where N is the number of hidden states and T is the length of the sequence.)

For this work we will be using a single observable – the percentage change in a financial assets price from the previous day. This is a simple observable which will allow us to classify market states by risk and volatility.

2.1 Training on S&P500 Data

The S&P500 is market share capitalised measure of the top 500 companies in the US. This captures the broad market in the US. We will be training our HMM model on S&P500 data from 1930 – 2025. This will allow us the classify the overall market into a set of states. We can then construct an optimal portfolio for each hidden state to outperform the market.

As a starting point we will look at a two state system. We use the closing price percentage change from the previous day as a stationary observable feature. Training on the full data range is shown in Fig. 2.1.

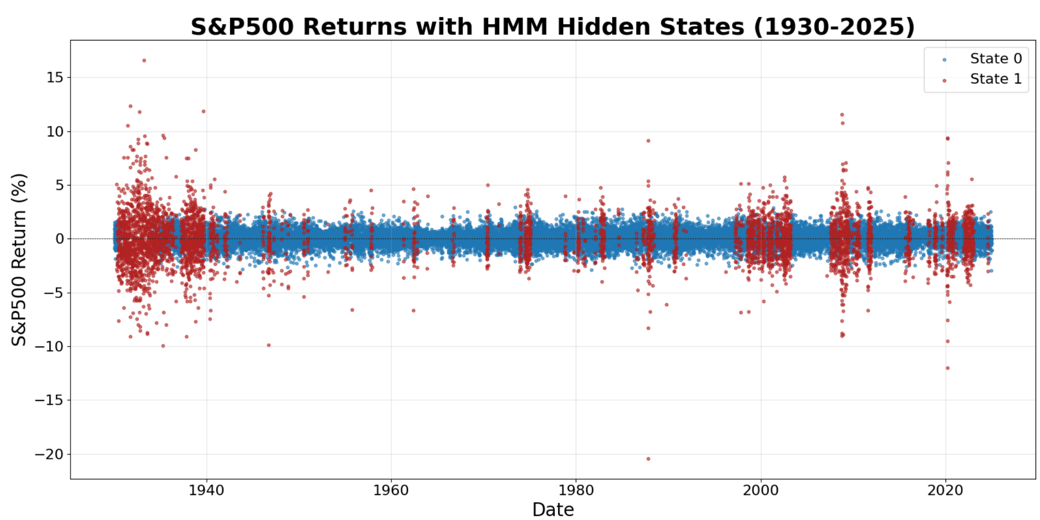

Figure 1: The figure shows the percentage change in the closing price of the S&P500 from 1930 to 2025. Each data point is assigned a HMM state corresponding to state 0 of state 1. State 0 (shown in blue) shows lower risk and volatility, which can be interpreted as a risk off market regime. State 1 shows higher risk and volatility, which can be interpreted as a risk on market regime. Based on this information we could size our portfolios appropriately based on the market regime we are in.

The results clearly show a low and high risk (or risk-off and risk-on) market regimes. The S&P500 has a higher volatility in state 1 compared with state 0. Based on this information, we could size portfolios of assets appropriately depending on the market state we are in.

Currently we could not use this model as we've trained it on the entire range of data. We would need to train on a training set of data and construct a walk forward analysis to determine if a strategy works.

We can look at the performance of two asset classes: Stocks (or the S&P500) and gold in each market regime:

Table 1: Summary Statistics for S&P 500 and Gold Returns by State

Metric	**State 0**	**State 1**
Total Observations	17,102	4,248
S&P 500 Returns		
Mean	0.0554%	-0.0800%
Std. Deviation	0.7090%	2.2375%
Gold Returns		
Mean	0.0488%	0.0447%
Std. Deviation	1.0647%	1.4356%
Observations	2,562	1,043

We can see the equities perform well in State 0 and poorly in State 1, reinforcing our notion of risk on and risk off. In state 0 the standard deviation is much lower than in state 1, again showcasing state 1 as a higher risk when holding stocks. Gold returns do not differ considerably between the states suggesting this asset is a good way of hedging equity risk (particularly in state 1). There is slightly more volatility when holding Gold in state 1, however, this is less pronounced than with Stocks.

Ensure Data Stationarity

A common problem in financial machine learning is that the data is not stationary – meaning that the data's statistical properties, such as mean and variance, change over time. If the data is non stationary then the model may not be able to learn concrete representations as the goal post keep moving. This is a problem for a variety of financial features, such as asset price, whose statistical properties change over time (as a simple example, the mean of a stock price may change as the stock appreciates in value).

We need to make sure that our percentage change dataset is stationary – that way the model can learn concrete and useful insights from the data. This should create a stationary dataset as we differencing neighbouring data points.

The Augmented Dickey-Fuller (ADF) test is the standard statistical test used to determine if a time series is non-stationary due to a unit root. Essentially, it tests whether a change in the data is a result of a predictable trend or if it is a random walk that will not return to its mean. If the test is below some critical p-value then the data is assumed to be stationary.

We applied the ADF test to the distribution of daily returns for the S&P500 and Gold and found that they were stationary at the 1% confidence interval. Therefore, daily percentage returns could offer a good baseline for our HMM model.

2.2 Walk-Forward Portfolio

A Walk-Forward Backtest (WFB) is a method of model validation that aims to prevent overfitting to training data. While a traditional backtest evaluates a strategy over a single, static historical period (say between 2015-2025) a walk-forward approach simulates the real-world process of periodically re-optimizing a model as new data becomes available. In this way, we train the model on previous historical data, and then backtest the model for a period of time after this, say one year. We then move forward and re-train and optimise the model and then backtest the model on the next batch of unseen data. In this way we have two phases:

1. **In Sample Phase:** A 'training' window where the model's parameters are optimised to maximise a specific objective function, like expected returns or the Sharpe ratio (risk adjusted returns).

2. **Out Sample Phase:** A 'testing' window where the optimised parameters are applied to unseen, forward-looking data.

This approach prevents overfiting by ensuring that the optimal coefficients for our model generalise well to unseen data. That is, if a researcher tests parameter combinations on the some dataset, they will eventually find a 'perfect' configuration by pure chance. The WFB addresses this by:

1. **Ensuring Temporal Integrity:** It strictly respects the forward passing of time; the model never looks ahead at future data during the optimisation phase.

2. **Measuring Parameter Stability:** If a strategy requires drastically different parameters in every walk-forward window to remain profitable, it suggests the model is capturing noise rather than a structural market alpha.

We construct our WFB through the 2005-2026. We will train the model on data from 1980 to the present backtest windows. We then backtest over the next year before shifting forward one year and performing the same backtest. In this way we can measure the performance of the model over the full timescale.

The question still remains of how we size our positions based on the market state. For this we will use the Kelly Criterion to size our positions.

The Kelly Criterion

The Kelly Criterion is a mathematical formula used to determine the optimal size of a series of bets to maximize the long-term growth of capital. Developed by John Kelly in 1956, it balances the trade-off between risk and reward by calculating the exact percentage of a portfolio that should be committed to a specific favourable opportunity. The formula is expressed as $f^* = \frac{bp-q}{b}$, where f^* is the fraction of the current bankroll to wager, b is the odds received on the wager (decimal odds - 1), p is the probability of winning, and q is the probability of losing.

The primary advantage of the Kelly Criterion is that it theoretically prevents ruin—since you only ever bet a percentage of your remaining capital—while ensuring that your wealth grows at the maximum possible exponential rate. However, in practical investing, many practitioners use a "Fractional Kelly" approach (such as betting only half the recommended amount). This is because the standard formula is highly sensitive to the accuracy of your probability estimates; if you overstate your edge, the Kelly Criterion can lead to extreme volatility and significant drawdowns.

For the S&P500 and Gold respectively, we treat returns as a mixture over HMM states with weights equal to the posterior (the probability of being in a specific hidden state at a specific time) at the current time. So for each asset we calculate the mean μ and standard deviation σ from the posterior.

We then calculate the single-asset Kelly fraction as:

$$f^* = \frac{\mu}{\sigma^2}.$$

So the S&P500 get f^*_{sp} and Gold f^*_{gold} from their posterior weights. We then size these fractions by half Kelly (multiply by 0.5) and clip to between the maximum leverage. In this work we will not be using any leverage, so f^* is clipped between $[-1, 1]$. We finally normalise these weights by the total of $f^*_{sp} + f^*_{gold}$ to ensure the final weights sum to one.

Our final portfolio weights are between 0 and 1 meaning we only have long positions (no short selling).

WFB Results

Figure 2.2 shows the the performance of our WFB strategy vs. the S&P500 buy and hold strategy over 2005-2026, with a 0.1% trading fee when resising the portfolio[1]. The portfolio shows strong performance over this period and, in particular, preforms well in market downturns. We observe considerably fewer downturns compared with the S&P500, suggesting our portfolio holds less risk.

The empirical results presented in Table 2 reveal a significant performance divergence between the proposed Stitched Strategy and the S&P 500 benchmark over the 5,292 trading-day sample period. The strategy achieved a total return of 4500.69%, translating to an annualized return of 19.29%, nearly doubling the 10.27% return generated by the benchmark. These improved returns are noteworthy given the strategy's simultaneous reduction in risk across multiple dimensions. Notably the strategies beta is 0.2 suggesting we are able to make returns relatively independently of the market.

The primary strength of the model lies in its risk-adjusted efficiency, as evidenced by the Sharpe Ratio and Sortino Ratio:

1. This is a relatively naive and simple approach to estimating the trading fee incurred. The actual cost of trading will depend on the capital under management and the liquidity available in the market. If there is not enough liquidity to make the trade the cost of trading will be much greater. For simplicity, we will assume we are manageing a small portfolio and maintain the 0.1% trading fee.

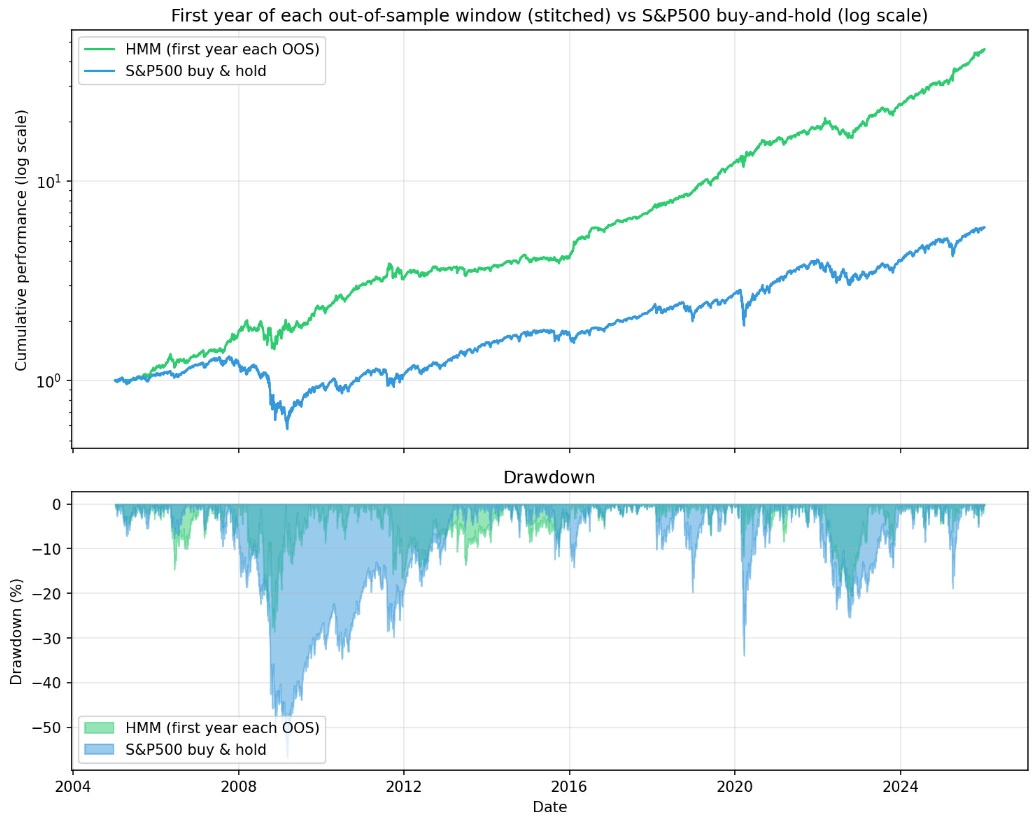

Figure 2: A figure showing logarithmic returns of our WFB vs. the S&P500 and the maximum drawdown suffered by the portfolios. The WFB shows consistent returns over the 2005-2026 periods with lower downturns compared with the S&P500. Notably, the portfolio performs well in market downturns.

- **Reward-to-Variability:** The strategy's Sharpe Ratio of 1.33, compared to the benchmark's 0.54, indicates that the model produces substantially more return per unit of total risk.
- **Downside Protection:** The Sortino Ratio of 1.74 (vs. 0.65 for the S&P 500) suggests that the strategy is effective at minimising downward volatility – those returns falling below the risk – free rate – while capturing upside momentum.

The model's resilience during market stress is quantified by its tail-risk metrics and drawdown profile. The Max Drawdown of -28.58% is approximately half that of the S&P 500 (-56.78%), implying that the strategy likely incorporates a regime-switching mechanism or dynamic hedging that mitigates catastrophic losses during bear markets.

Furthermore, the strategy exhibits a Beta of 0.20, signifying a low correlation with the broader market. This low market sensitivity, combined with improved Value at Risk (VaR) and Conditional Value at Risk (CVaR) figures, suggests the model successfully identifies alpha rather than merely tracking systematic market beta.

Table 2: Comparative Performance Statistics

Metric	**WFB**	**S&P 500**
Total Return (%)	4500.69	488.28
Ann. Return (%)	19.29	10.27
Ann. Volatility (%)	14.52	19.13
Sharpe Ratio	1.33	0.54
Sortino Ratio	1.74	0.65
VaR (5%)	-1.32	-1.79
CVaR (5%)	-2.12	-2.96
Beta	0.20	—
Max Drawdown (%)	-28.58	-56.78
Trading Days	5292	5292

The Diebold-Mariano Test

In time-series forecasting, the Diebold-Mariano (DM) Test analyses the predictive accuracy of two competing forecasts. This test essentially asks 'does Model A have better predictive accuract than Model B, or is the difference in their performance just down to random nosie.

DM focuses in the forecast residuals of two models. If both models are equally good, the difference in their residuals should be about zero on average. We first define some loss function L, which is usually the Mean Squared Errror (MSE) or the Mean Absolute Error. For each time in the series, we calculate the loss differential d_t:

$$d_t = L(e_{1_t}) - L(e_{2_t}),$$

where $e_{1,t}$ is the error of Model 1 at time t and similarity for e_{2_t}.

DM checks if the null hypothesis (H_0) that the mean of the differentials is zero:

- H_0: $E[d_t] = 0$ then both models have the same predictive accuracy.
- H_0: $E[d_t] \neq 0$ the one model performs better.

The test statistic is calculated as:

$$DM = \frac{\bar{d}}{\sqrt{\hat{V}(\bar{d})}},$$

where $\bar{d}$ is the mean of the loss differentials and $\hat{V}(\bar{d})$ is the variance of the mean. As forecast errors are often autocorrelated (especially in multi-step-ahead forecasts), the DM test uses a Long-Run Variance estimator (like Newey-West) to ensure the results are not biased.

The DM statistic follows a standard normal distribution under the null hypothesis:

- **High Positive DM:** Model 2 is significantly better (lower loss) than Model 1.
- **Low Negative DM:** Model 1 is significantly better (lower loss) than Model 2.
- **Near Zero / Low p-value**: No significant difference in performance.

Statistical Significance of Results

We will use the DM model to test the statistical significance. We will compare the simple benchmark of the S&P500 model (Model 1) vs. our HMM Model (Model 2). We define the loss function to be the negative daily return at a given timestep:

$$L_t = -r_t,$$

where L is the loss function and r_t is the return at the time step. In this way, a lower loss indicates the model performed well on that day (as the return would be positive) and vice versa. So the loss function is effectively negative daily return; we're testing whether our HMM strategy has a statistically significant higher average return than the benchmark over the walkforward period.

Over the WFB the mean loss differential was 0.0358% which gave a DM statistic of 2.36 with a p-value of 0.0184, suggesting the two models do not have the same predictive accuracy at the 5% confidence interval. This suggests that our HMM model has statically significant better performance than buying and holding the index.

3 Conclusions

We applied a basic 'risk-on' 'risk-off' HMM model to daily percentage changes in the S&P500 data from 1930 through to the present time. This dataset effectively captures the underlying volatility profile of the market – one state had lower volatility and comparatively higher equity returns and one state had higher volatility with negative equity returns. Gold was an effective hedge in these scenarios, offering similar levels of performance across each of the states. This allowed us to construct 'risk-on' and 'risk-off' portfolios as part of a Walk Forward Backtest, which offered good risk adjusted returns compared to a buy and hold strategy. This portfolio performed particularly well during market downturns but sometimes struggled to keep pace with stock performance during periods of lower volatility. We also saw that the superior performance was statistically significant and not due to random noise in the dataset.

Overall, this work shows that HMM can offer a simplistic model for risk profiles, and can be used to hedge positions during market downturns. Over a large timespan these approaches will likely outperform a buy and hold strategy, with better risk adjusted returns. Further research could look to expand to other features, particularly volatility features like the VIX levels, to improve on the modelling done in this work. However, to avoid overfitting, it would be best to limit the number of features used in such modelling. Additionally, the HMM could be used to split data into two different volatility profiles. This could then be used to train further downstream models on price prediction or portfolio allocation to further improve performance.