# HMM Volatility Forecasting with Uncertainty-Aware Covariates

Duncan Cameron-Steinke

April 15 2026

## Contents

# 1 Hidden Markov Models

A **Hidden Markov Model (HMM)** as explained by Zucchini et al. [2017], consists of two stochastic processes, a state process $\mathbf{S} = [S_0, S_1, \ldots, S_n]$, and a state-dependent observation process $\mathbf{Z} = [Z_0, Z_1, \ldots, Z_n]$. The state process represents the "hidden" random variable to be inferred, while the observation process is a directly measurable random variable. For example, a financial analyst can observe the weekly price changes of the Toronto Stock Exchange Composite Index, hereafter referred to simply as TSX, and directly measure those price changes as the random variables ($[Z_0 = \text{TSX}_0, Z_1 = \text{TSX}_1, \ldots, Z_n = \text{TSX}_n]$). From the observed sequence an unobserved hidden sequence representing market regime can be inferred despite that sequence never being directly measured.

HMMs are characterized by the following dependence assumptions:

1. The hidden state sequence $\mathbf{S}$ is composed of discrete probability vectors over the state space $K$ and follows a Markov Process, which means that the state vector at time $n+1$ is only dependent on the state vector at time $n$

$$\Pr\big(S_{n+1} \mid S_n, S_{n-1}, \ldots, S_0\big) \;=\; \Pr\big(S_{n+1} \mid S_n\big).$$

2. The hidden state vector $(S_n)$ evolves based on the transition probability matrix $P$

$$S_{n+1} = \begin{pmatrix} s_1 \\ s_2 \\ \vdots \\ s_K \end{pmatrix}_{n+1} = \begin{pmatrix} p_{11} & p_{12} & \cdots & p_{1K} \\ p_{21} & p_{22} & \cdots & p_{2K} \\ \vdots & \vdots & \ddots & \vdots \\ p_{K1} & p_{K2} & \cdots & p_{KK} \end{pmatrix} \begin{pmatrix} s_1 \\ s_2 \\ \vdots \\ s_K \end{pmatrix}_{n} = PS_n$$

   where $s_i := \Pr(S_n = i)$ is the probability of hidden state $i$, and $p_{ij}$ is the probability of transitioning from state $i$ to $j$ over a single time step.

3. The observation $Z_n$ is a random variable which also follows a Markov process conditioned on the current hidden state

$$f\big(Z_n \mid Z_{n-1}, \ldots, Z_0, S_n, \ldots, S_0\big) \;=\; f\big(Z_n \mid S_n\big).$$

While the function $f(Z_n \mid S_n)$ could, in principle, be any probability density function, in practice it typically follows a parametric model (such as a Gaussian, Poisson, Binomial) with parameters set by the hidden states. In the case of a normal distribution, the conditional observation density function is

$$f(Z_n \mid S_n = i) \sim \mathcal{N}(\mu_i, \sigma_i^2), \qquad i \in K.$$

The marginal density function for a given hidden state vector (i.e the full weighted density) is a weighted sum over all the hidden states

$$f(Z_n \mid S_n) = \sum_{i=1}^{K} \Pr(S_n = i)\; f(Z_n \mid S_n = i).$$

## 1.1 hmmTMB

The software package hmmTMB presented in Michelot [2025] extends the usefulness of HMMs by allowing for covariate effects within both the observation distribution parameters and the hidden state transition probabilities. Let $\boldsymbol{X}_{[n,m]}$ be a design matrix of the covariates having $n$ rows, corresponding to each observation, and $m$ columns denoting the number of covariates. The linear predictor for the fixed effect covariates is

$$\eta = \boldsymbol{X}\boldsymbol{\alpha}$$

where $\alpha$ is a parameter vector of length $m$.
Additionally, hmmTMB supports random effects, which capture group specific differences while assuming that the groups are normally distributed overall. Combing fixed and random effects yields the mixed effects linear model

$$\eta = \boldsymbol{X}\boldsymbol{\alpha} + \boldsymbol{R}\boldsymbol{\beta}, \qquad \boldsymbol{\beta} \sim N(0, \underline{\ }\mathrm{Q})$$

where $\underline{\mathrm{R}}_{[n,k]}$ is a matrix with $k$ columns corresponding to the number of groups to be modelled and $\underline{\mathrm{R}}_{[i,j]} = 1$ if observation $i$ belongs to group $j$ and 0 otherwise.
Splines, including cubic and penalized splines, allows for non-parametric modelling between the covariates and the response variable $\eta$. They are incorporated within the mixed effect model as

$$\eta = \boldsymbol{X}\boldsymbol{\alpha} + \boldsymbol{R}\boldsymbol{\beta} + \boldsymbol{R'}\boldsymbol{\beta'}$$

where columns of $\boldsymbol{R'}_{[n,q]}$ are the spline basis functions evaluated at a covariate and $\boldsymbol{\beta'}$ are the function weights.

The mixed effect model produces a real valued predictor $\eta \in \mathbb{R}$. However many HMM parameters have specific domains requiring the use of a link function to map between the real valued linear model and the HMM parameter $\theta$,

$$\eta = h(\theta).$$

Table 1: HMM link functions

| Component | Parameter | Link Function | Conversion Formula |
|---|---|---|---|
| Observation Distributions | Normal (mean) | Identity | $\eta = \mu$ |
| | Normal (variance) | Identity | $\eta = \sigma^2$ |
| | Poisson (scale) | Log | $\exp(\eta) = \lambda$ |
| Hidden State | Transition rate | Logit | $\frac{\exp(\eta_{ij})}{\sum_{k=1}^{K}\exp(\eta_{ik})} = p_{ij}$ |

## 1.2 Forecasting Extension

hmmTMB does not directly support forecasting, so I developed an extension of hmmTMB which allows for forecasts while accounting for the flexible impact of covariates[1]. Forecasting

[1]See: `https://github.com/DuncanCSt/hmmTMB/blob/forecast_dev/R/forecast.R`

with covariates assumes that the covariates must be known into the forecast period. Let $X^{(t)}$ and $R^{(t)}$ be the design matrix for a forecast period $t$. The model parameters are computed as

$$\theta^{(t)} = h^{-1}(\eta^{(t)}) = h^{-1}(X^{(t)}\hat{\boldsymbol{\alpha}} + R^{(t)}\hat{\boldsymbol{\beta}})$$

where $\theta^{(t)}$ is the forecasted parameter of interest, and $h^{-1}(\cdot)$ is the inverse link function for that parameter. So given a forecasted design matrix we can compute the transition rates $p_{ij}^{(t)} = h^{-1}(\eta^{(t)}$ and write the full probability matrix simply as $P^{(t)}$.

For a last known hidden state vector $S_n$ the forecasted vector $S_t$ is computed via recursion as

$$\begin{aligned} S_{n+1} &= P^{(1)}S_n \\ S_{n+2} &= P^{(2)}S_{n+1} = P^{(2)}P^{(1)}S_n \\ &\cdots \\ S_t &= \left(\prod_{i=1}^{t} P^{(i)}\right) S_n. \end{aligned}$$

Once the hidden state vector is known the observation distribution is computed in the usual way, while noting that the observation parameters may themselves be functions of forecasted covariates.

$$Z_t \sim \sum f(\theta_i^{(t)}) \Pr(S_t = i)$$

# 2 Modeling Financial Timeseries

Hidden Markov Models are frequently used to model heteroscedasticity in financial timeseries, meaning modeling techniques that capture a non constant variance. HMMs are also easy to interpret, unlike 'black-box' forecasting methods, and they are naturally auto-regressive which further lends itself to the study of financial timeseries. Financial timeseries are also useful to forecast so it is a natural choice for testing forecasting methods. However, most financial timeseries, including the TSX index, don't naturally follow parametric distributions so can't be fit directly to an HMM. A useful solution is to model the timeseries using the drift volatility model which assumes the asset follows a random walk with a constant drift term. After applying the drift volatility model, differencing and log transforming, the result is Gaussian distribution representing log returns over a single time step. For more in depth analysis of the drift volatility model applied to the TSX refer to appendix A.

We can see from figure 1 that the HMM states generally correspond to distinctive trends of the observed TSX price. To understand these states further, and to apply a physical intuition to them, we examine the observation parameters broken down by state.

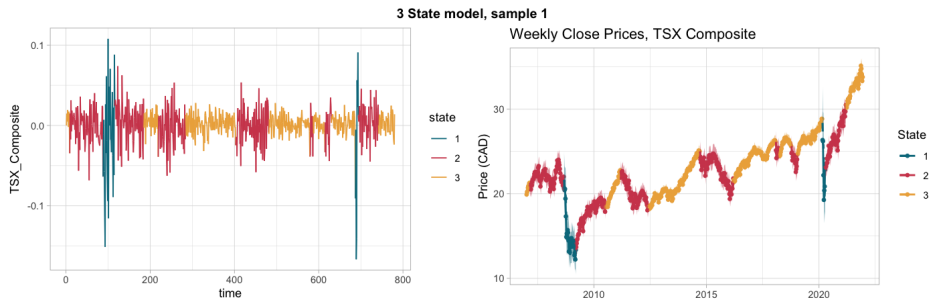


Figure 1: Left: A 3-state HMM fit to TSX log returns based on the Friday weekly close price. Right: The TSX close price plotted on a linear scale with points coloured by the HMM.

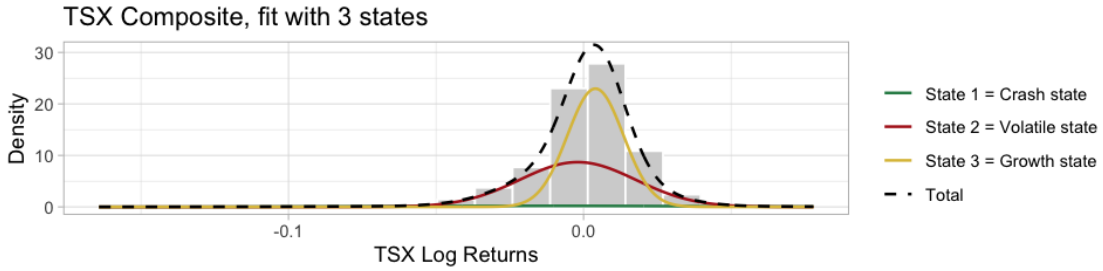


Figure 2: Plot of Observation distributions broken down by hidden states shows that the 3 state Gaussian model fits the data well.

Table 2: Observation Parameters for each hidden state of the 3-state model.

| | State 1 | State 2 | State 3 |
|---|---|---|---|
| TSX_Composite Mean | -0.004 | -0.003 | 0.005 |
| TSX_Composite SD | 0.044 | 0.019 | 0.009 |

By examining the observation parameters in table 2 we can see that state 3 is the only one with a positive expected growth so we can call it the growth state. State 2 and State 3 are both volatile with negative expected growth, but from the plot in figure 1 we can see that State 1 corresponds to significant market crashes while state 2 is volatile without a preferred direction. We can therefore interpret hidden states as corresponding to the distinct market regimes of Crash, Volatile, and Growth.

# 3 HMM Forecasting

The goal of this analysis is not to predict the observation sequence (TSX log returns) directly, but rather to predict the HMM hidden states. This makes sense for two reasons, first because the HMM hidden states correspond to physical market regimes which are intrinsically useful to forecast, and second because the observation sequence is derived from the hidden state so improved hidden state estimates are a prerequisite for improved market predictions.

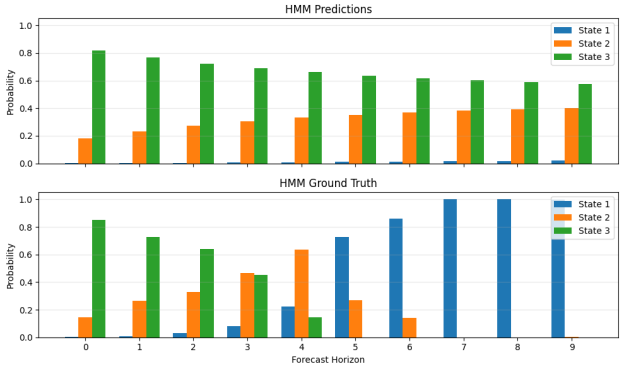


Figure 3: Top: Hmm forecasts over 10 timesteps shows a gradual reversion to the stationary distribution. Bottom: Ground truth TSX state labels which don't match the HMM predictions.

### 3.1 Ground Truth labels

The ground truth labels used to train and evalutate the model are obtained in two steps. First a 3-state HMM is fit on the training set which comprises data from 2005-2017. Second, the fitted HMM model is used to perform state decoding to the full TSX time series. The last 5 observations (i.e the last 5 weeks) of the test period are discarded after decoding to avoid edge effects.

An example of an HMM forecast is shown in figure 3 and highlights the drawback of HMM forecasting. While the forecast does well at predicting the hidden state over short time steps, its inflexibility prevents the model from capturing sudden changes, as seen by state 1 becoming dominant in the ground truth labels, while the HMM forecast continues to predict states 2 and 3.

Applying covariates to the hidden states is a reasonable method that allows the state transition process to become more dynamic and responsive. However including covariates directly requires knowledge of the covariate values throughout the forecast period. While some covariates are naturally known in advance, such as civic holidays, seasonal averages, age of an individual, etc. many other covariates are not. It is therefore useful to develop HMM forecasting methods that can leverage present day covariate information to adjust its future transition probabilities.

## 4 Deep Learning model and feature set

I designed a deep learning model which predicts future HMM hidden states given a historical covariate feature set. This approach doesn't require extrapolating covariates and allows for complex non-linear relationships between the covariates and future TSX market regims. The four covariates selected along with TSX are shown in figure 4.

A 3-stage model was used. The first stage is a Recurrent-Neural-Network (RNN), which models the evolution of the covariates over a the lookback period. Next, a sequence of fully connected dense layers are used to model the complex covariate inter-relationships. Finally an output layer computes the parameters of interest, which will be discussed more in the next

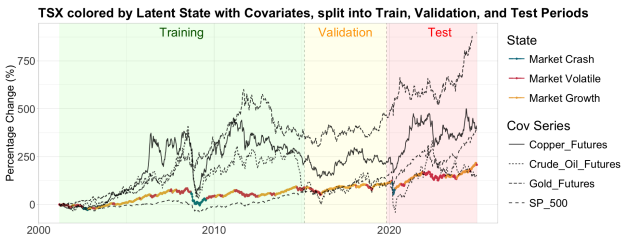


Figure 4: TSX along with covariates for data split into 60% training, 15% validation and 25% testing periods.

section. Dropout and L2 Regularization were employed in both the RNN and the Dense layers to avoid overfitting. Hyper-parameters, such as the number of dense layers, RNN-units, the dropout rate, and the regularization rate were optimized by performing a hyper-parameter search that minimized the average Kullback-Leibler divergence score on the validation set.

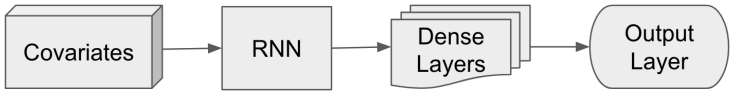


Figure 5: Deep learning model which predicts future market regimes based on historical covariates.

The TSX hidden states are themselves included as a covariate while also being the target ground truth label. The significant difference is that the TSX covariate hidden states represent the best estimate up to the last available timestep, while the ground truth labels also include all future information. So the ground truth label for an index i might be very different from the TSX covariate for that same index. A complete description of the data engineering steps are presented in appendix B.

# 5 Evidential Learning

The seminal work Evidential Deep Learning work by Sensoy et al. [2018] applies the theory of subjective logic Jøsang [2016] towards uncertainty aware classifications using deep learning. The core principle is to use the Dirichlet Distribution as a posterior estimate of the class labels, where the Dirichlet parameters are predicted by the model. This can be visualized by comparing the generic classification model using a softmax output seen in figure 6, with an evidential model using a ReLu output shown in figure 7. It can be seen that the generic classification model only provides a point estimate without any uncertainty measures, meanwhile the evidence vectors of the Dirichlet distributions allows for various levels of uncertainty for the same expected point estimate.

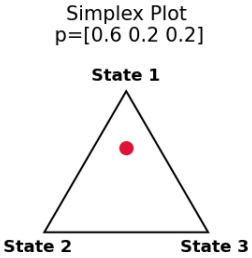


Figure 6: Generic classification output using a softmax output layer and no uncertainty quantification.

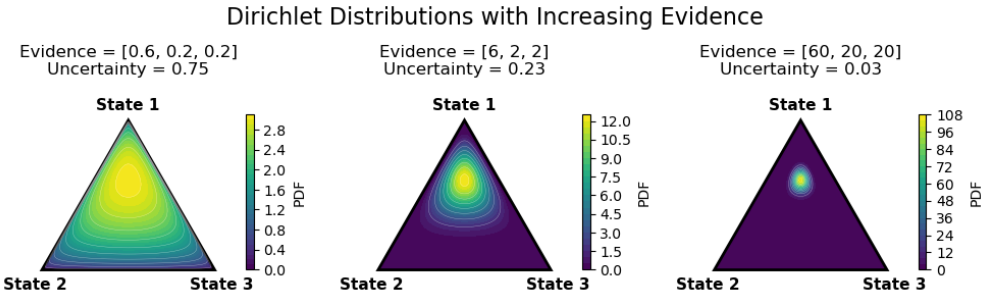


Figure 7: Evidential learning posterior distributions as a function of evidence vector with decreasing uncertainty.

The evidential deep learning model requires a specialized loss function where the goal is to maximize the probability mass over the true value. The loss is computed using the Bayes risk, which is the expected loss over the posterior for a given loss function.
The Cross Entropy Bayes risk given a target $\mathbf{y}$ with a Dirichlet posterior parameterized by $\boldsymbol{\alpha}$ is

$$\mathcal{L}(\Theta) = \mathbb{E}_{\boldsymbol{p}\sim \mathbf{Dir}(\boldsymbol{\alpha})}[\,\mathbf{CE}(\boldsymbol{y},\boldsymbol{p})] = \int \mathbf{CE}(\mathbf{y},\mathbf{p})\,\mathrm{Dir}(\boldsymbol{p},\boldsymbol{\alpha})d\mathbf{p}.$$

Where $\mathbf{y}$ is the true label and $\mathbf{p} \sim \mathrm{Dir}(\alpha)$ is a random realization of the Dirichlet distribution. The cross entropy loss is

$$\mathbf{CE}(\mathbf{y},\mathbf{p}) = -\sum y_i \log(p_i).$$

The expectation can be simplified as

$$\begin{aligned}
\mathbb{E}_{\mathbf{p}\sim \mathrm{Dir}(\alpha)}[\,\mathbf{CE}(\mathbf{y},\mathbf{p})] &= \mathbb{E}_{\mathbf{p}\sim \mathrm{Dir}(\alpha)}\left[-\sum_{i=1}^{K} y_i \log(p_i)\right] \\
&= \sum_{i=1}^{K} y_i\ \mathbb{E}_{\mathbf{p}\sim \mathrm{Dir}(\alpha)}[-\log(p_i)] \\
&= \sum_{i=1}^{K} y_i\,(\psi(S) - \psi(\alpha_i))
\end{aligned}$$

Where $\psi(x)$ is the digamma function and $S$ is the total confidence computed as $S = \sum_{i=1}^{K} \alpha_i$. The expectation is only applied to $p_i$ since $y_i$ is the true observation and not a random variable. $\mathbb{E}_{\mathbf{p}\sim \mathrm{Dir}(\alpha)}[-\log(p_i)] = \psi(S) - \psi(\alpha_i)$ is a known property of the Dirichlet distribution.

Sensoy et al. [2018] further propose the use of a regularization term that adds an additional penalty for predictions that deviate from a uniform distribution. However, in the case of learned distributions over labels, the uniform distribution is a valid target and does not represent the absence of information. So instead I created a novel penalty term based on total confidence that encourages predictions to have lower confidence (a.k.a. high uncertainty). The novel confidence penalty follows the same spirit as the regularization term of Sensoy et al. [2018], they are both an attempt to push the model towards a default state of "I don't know", such that the model only assigns evidence when a true signal is detected. The loss function with a confidence penalty is

$$\mathcal{L}(\Theta) = \mathbb{E}_{\boldsymbol{p}}[\,\mathbf{CE}(\boldsymbol{y}, \boldsymbol{p})] + \lambda \log(S).$$

Where $S$ is the total confidence and $\lambda$ is a hyper-parameter which is tuned on the validation set.

The EDL predictions are taken as the expectation of the Dirichlet distribution, and uncertainty is the reciprocal of confidence.

$$EDL : \boldsymbol{p} = \frac{1}{\sum_{i=1}^{3} \alpha_i} \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{pmatrix} = \begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix}, \qquad u = \frac{1}{\sum_{i=1}^{3} \alpha_i}. \tag{1}$$

The EDL model was trained on the training set using the custom loss function and a plot of confidence vs cross entropy loss of the validation set is shown in figure 8. From the plot we can see a strong correlation between uncertainty and accuracy where low uncertainty predictions (a.k.a high confidence) tend to produce lower loss (a.k.a high accuracy). The choice of $\lambda$ is an important hyper parameter, lower values encourage the model to prioritize overall accuracy, while a larger value yields a stronger correlation between accuracy and uncertainty.

## 5.1 Hybrid Model

I developed a hybrid HMM-EDL forecasting model that leverages EDL when EDL confidence is high and falls back to the HMM otherwise. This approach makes rapid hidden state transitions possible whenever a strong covariate signal is present, while still preserving the key statistical advantages of a Hidden Markov Model. These advantages include reversion to the long-term average over long horizons, dependence between successive predictions, and straightforward interpret-ability.

In the hybrid model hidden state transition rates are computed as a function of both the EDL predictions and the native HMM transition rates using

$$p^*_{ij} = \boldsymbol{p}(j) \cdot \beta(u) + p_{ij}(1 - \beta(u)).$$

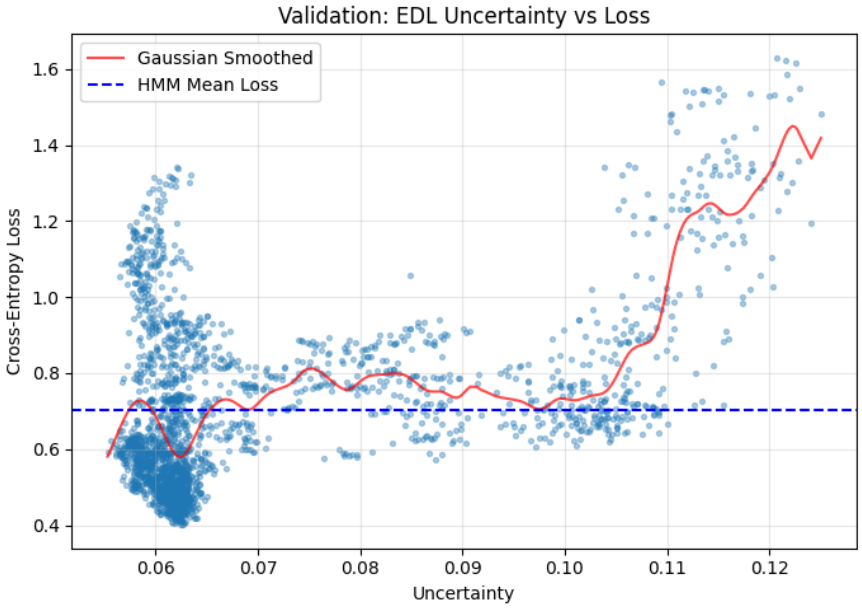


Figure 8: EDL uncertainty vs cross-entropy loss with confidence penalty $\lambda = 0.2$ shows a strong correlation between uncertainty and accuracy. The red line is a Gaussian Kernel Regression help visualize the relationship. Average loss using a covariate free HMM forecast is plotted as a dashed blue line as a reference.

Where $\boldsymbol{p}(j)$ is the EDL prediction of state $j$, $p_{ij}$ is the native HMM transition rate from $i$ to $j$ and $\beta(u) \in [0, 1]$ is a mixing parameter.
The mixing parameter incorporates EDL uncertainty and only assigns weight to EDL when confidence is high. For example, in the case of complete uncertainty $\beta(u) = 0$ and the hybrid model reverts to native HMM, conversely when confidence is high $\beta(u) = 1$ the hybrid model relies exclusively on EDL. A truncated linear model was used for the mixing function and optimized using the the validation set

$$\beta(u) = \begin{cases} \beta_0 - \beta_1 u & \text{if } 0 \leq \beta(u) \leq 1, \\ 1 & \text{if } \beta(u) > 1, \\ 0 & \text{if } \beta(u) < 0. \end{cases}$$

# 6 Results

The HMM, EDL and hybrid HMM-EDL models were all evaluated on the test period. The results for each model over all 10 forecast horizons are plotted in figure 9. The results show that the Hybrid model is the only model that is superior for both minimum and maximum loss metrics. It is expected that hybrid does best at the extreme cases because those are the conditions when the uncertainty quantification is most meaningful. The hybrid model also does very well with mean loss, where it is close to, and occasionally best. What is surprising is that the median loss is worse compared to the HMM, it's not clear from the other results why this would be the case. A full table of results is provided in appendix C.

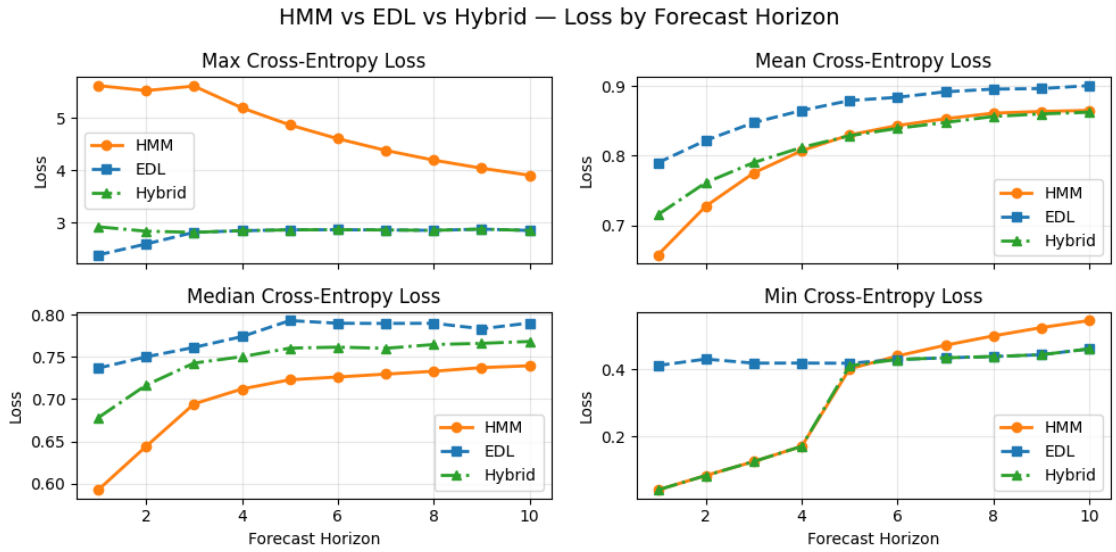


Figure 9: Maximum, mean, median, and minimum loss by forecast horizon for the HMM, Hybrid, and RNN models.

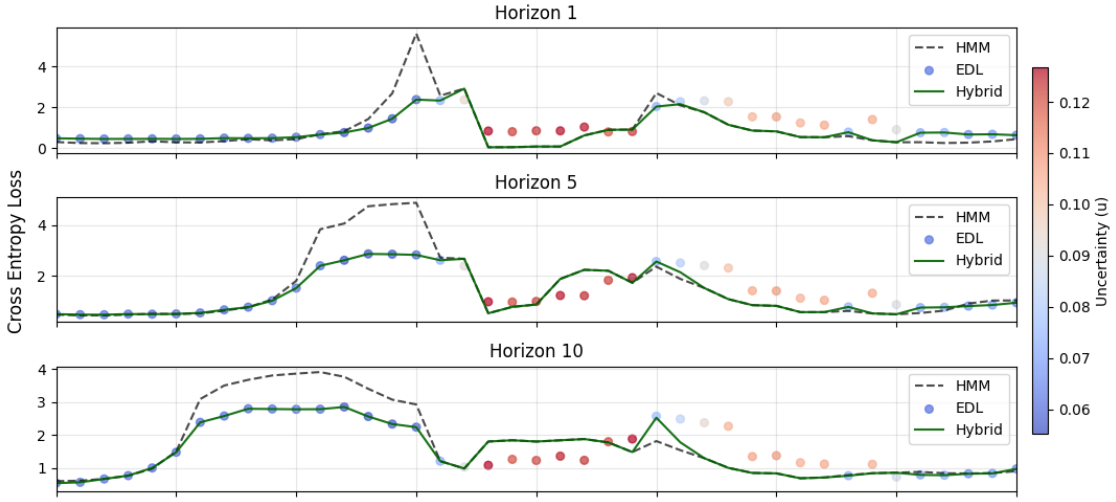


Figure 10: Loss by forecast period for three different forecast horizons. Demonstrate that the Hybrid model is correctly utilizing EDL when uncertainty is low (blue points), and using HMM when uncertainty is high (red points).

To understand if the Hybrid model was exhibiting the desired behavior, loss for each forecast step for three different forecast horizons is plotted in figure 10. The timeseries clearly show that the hybrid model benefits from EDL when uncertainty is low and uses HMM otherwise. While there are some examples where the EDL model predicts high uncertainty, despite having a lower loss then HMM, in general the uncertainty quantifications appear accurate and lead to improvements in the Hybrid Model.

# 7 Conclusion

HMMs are an essential modeling technique that thrive at capturing regime switching patterns in timeseries data. hmmTMB is an R framework which extends the usefulness of HMMs by allowing for covariate effects on both the observation and hidden state parameters through the use of mixed effect linear models. I extended the R package by producing forecasts from fitted HMMs. However, HMM forecasts with covariates are limited by the fact the covariates must be known within the future periods. Evidential Deep Learning offers a solution by allowing future hidden states to be predicted directly using historical covariates while also providing uncertainty estimates of its own predictions. I extended EDL by introducing a custom loss function which works better on distributions over labels. I proposed a novel Hybrid HMM-EDL model which leverages EDL when uncertainty is low and uses HMM otherwise. Finally I applied the hybrid model to a real world task of forecasting the TSX market regime. I found that the Hybrid model does better in extreme cases achieving the best minimum and maximum loss scores, but does not always do better then direct HMM forecasts on average.

# References


Audun Jøsang. *Subjective Logic*. Artificial Intelligence: Foundations, Theory, and Algorithms. Springer International Publishing, Cham, 2016. ISBN 978-3-319-42335-7 978-3-319-42337-1. doi: 10.1007/978-3-319-42337-1. URL http://link.springer.com/10.1007/978-3-319-42337-1.

Théo Michelot. hmmTMB: Hidden Markov Models with Flexible Covariate Effects in R. *Journal of Statistical Software*, 114:1–45, September 2025. ISSN 1548-7660. doi: 10.18637/jss.v114.i05. URL https://doi.org/10.18637/jss.v114.i05.

Murat Sensoy, Lance Kaplan, and Melih Kandemir. Evidential Deep Learning to Quantify Classification Uncertainty. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/hash/a981f2b708044d6fb4a71a1463242520-Abstract.html.

Walter Zucchini, Iain L. MacDonald, and Roland Langrock. *Hidden Markov Models for Time Series: An Introduction Using R, Second Edition*. Chapman and Hall/CRC, Boca Raton, 2 edition, December 2017. ISBN 978-1-315-37248-8. doi: 10.1201/b20790.


# A Drift-Volatility model With Geometric Brownian motion

The drift difusion model of GBM is used as a standard method of modelling a stock. Which is represented by the stochastic differential equation as

$$dS_t = \mu S_t + \sigma S_t W_t \tag{2}$$

for a stock $S_t$ and $W_t$ is a weinner process modeling a random walk. The stochasitc differential equation as the closed form solution

$$S(t) = S_0 \exp\left(\left(\mu - \frac{\sigma^2}{2}\right) t + \sigma W_t\right). \tag{3}$$

Where the term $\mu$ is the drift, which is corrected by a factor of $\sigma^2/2$ to account for the positive bias of volatility on an exponenial function and $\sigma W_t$ is the variance which is characterized as

$$\sigma W_t = \sigma\sqrt{\Delta t} Z_t, \quad Z_t \sim N(0,1). \tag{4}$$

Next equation (3) is transformed using natural logarithm as

$$\ln\left(\frac{S(t)}{S_0}\right) = \left(\mu - \frac{\sigma^2}{2}\right) t + \sigma\sqrt{t} Z_t \tag{5}$$

Converting to discrete time and assuming that $\Delta t = n = 1$ we have

$$\ln\left(\frac{S[n]}{S[n-1]}\right) = \mu[n] - \frac{\sigma[n]^2}{2} + \sigma[n]Z_n \tag{6}$$

$$\sim N\left(\mu[n] - \frac{\sigma[n]^2}{2}, \sigma[n]\right) \tag{7}$$

We can therefore log transform the TSX returns and model weekly difference as a normal distribution.

### A.1 Converting between Working Scale and Natural Scale

In this analysis, the working scale is defined as the log transformed data:

$$Y[n] = \ln\left(\frac{S[n]}{S[n-1]}\right). \tag{8}$$

The conversion from the working scale back to the natural scale is given by

$$S[n] = S[n-1]\exp(Y[n]). \tag{9}$$

For the cumulative transformation up to step $N$, this extends to

$$S[N] = S[0]\exp\left(\sum_{n=1}^{N} Y[n]\right). \tag{10}$$

Note that if the $Y[n]$ represents a continuous probability distribution (rather than fixed values), the probability density function of their sum is obtained via the convolution of the individual distributions. For example, for two independent random variables $Z_1$ and $Z_2$ with PDFs $f_{Z_1}(z)$ and $f_{Z_2}(z)$, the PDF of $Z = Z_1 + Z_2$ is:

$$f_Z(z) = (f_{Z_1} * f_{Z_2})(z) = \int_{-\infty}^{\infty} f_{Z_1}(z-u) f_{Z_2}(u)\, du. \tag{11}$$

This extends to multiple variables through successive convolutions.

## B Creating the feature set

In addition to using the raw log covariate as a feature, it was found that fitting the covariate to its own HMM model and using the HMM hidden state vector also as a feature achieved superior prediction performance. The process of generating covariate features is similar but not identical to the ground truth labels. First, an HMM is fit to the covariate using the training period only. Next, for a given index i, a lookback period of 50 weeks, is used to decode the covariate states up to index i. The first 40 rows are discarded and the last 10, corresponding to the covariate hidden state over the last 10 weeks, is kept.

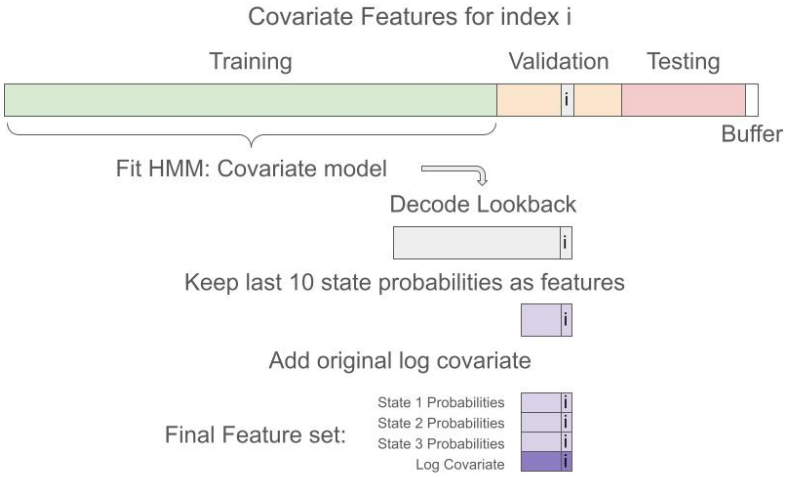


Figure 11: Process for obtaining feature matrix for index i.

It's important to highlight that the covariates (including TSX) suffer from edge effects and so the state sequence produced from the lookback up to index i may differ from the ground truth state sequence obtained when fitting beyond the i's index. This is impossible to avoid and stems from the models lack of future knowledge.

The process is repeated for the 5 covariates used, TSX, Oil Futures, Gold Futures, Copper Futures, and SP 500. The input shape is 10x20 for a lookback of 10 and 20 covariate columns, the output shape is 10 x 3 for a 10 state forecast over 3 TSX hidden states.

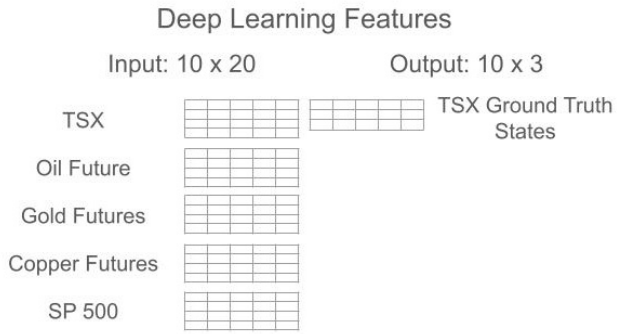


Figure 12: The Deep learning input shape is 10x20 for the 20 covariate columns each with a lookback horizon of 10. The output predictions are the ground truth TSX states.

# C Results table

Table 3: Cross Entropy Log Loss Comparison of HMM, Hybrid, and RNN Models Across Forecast Horizons

| | | **Forecast Horizon** | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Metric** | **Model** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
| | HMM | 5.625 | 5.530 | 5.616 | 5.199 | 4.871 | 4.608 | 4.384 | 4.197 | 4.042 | 3.908 |
| max | Hybrid | 2.919 | 2.835 | **2.813** | **2.844** | **2.861** | **2.864** | **2.859** | **2.851** | **2.875** | **2.850** |
| | EDL | **2.380** | **2.590** | 2.813 | 2.844 | 2.861 | 2.864 | 2.859 | 2.851 | 2.875 | 2.850 |
| | HMM | **0.658** | **0.728** | **0.775** | **0.807** | 0.830 | 0.843 | 0.853 | 0.861 | 0.863 | 0.865 |
| mean | Hybrid | 0.715 | 0.761 | 0.790 | 0.812 | **0.829** | **0.839** | **0.848** | **0.856** | **0.860** | **0.862** |
| | EDL | 0.790 | 0.822 | 0.847 | 0.865 | 0.879 | 0.884 | 0.892 | 0.896 | 0.896 | 0.901 |
| | HMM | **0.592** | **0.644** | **0.694** | **0.712** | **0.723** | **0.726** | **0.730** | **0.733** | **0.737** | **0.740** |
| median | Hybrid | 0.678 | 0.717 | 0.743 | 0.750 | 0.761 | 0.762 | 0.760 | 0.765 | 0.766 | 0.768 |
| | EDL | 0.737 | 0.750 | 0.761 | 0.774 | 0.793 | 0.790 | 0.790 | 0.790 | 0.784 | 0.790 |
| | HMM | 0.040 | 0.083 | 0.125 | 0.170 | **0.402** | 0.441 | 0.473 | 0.501 | 0.525 | 0.546 |
| min | Hybrid | **0.040** | **0.083** | **0.125** | **0.170** | 0.411 | **0.430** | **0.435** | **0.439** | **0.444** | **0.462** |
| | EDL | 0.412 | 0.431 | 0.419 | 0.419 | 0.419 | 0.430 | 0.435 | 0.439 | 0.444 | 0.462 |