Hands on Hidden Markov Models

Giovanni Battista Esposito[a], Giuseppe Martucci[a]

[a]University of Bologna, Bologna, Italy

Abstract

Markov chains has been known for almost 80 years, but it is only in the last decade that it has been applied explicitly to problems as time series forecasting, speech processing and path recognition. The major reasons why models, based on Markov chains, have not been developed until recently was the lack of an optimizing method for the parameters of the Markov model. The purpose of this *hands on* document is to give an introduction to the theory of Hidden Markov models and to present an example of how they can be applied to problems in time series forecasting.

Keywords: Advanced Time Series; *Hidden Markov Models*; *Classification*; *Econometrics.*

1 Introduction

The goal of this document is to show step by step how Hidden Markov Models work, starting from a basic introduction on classification and Markov property, then moving to the definition of Hidden Markov Model and his parameters. It will be illustrated answering at three main questions and consequently three algorithms will be described: Forward-Backward, Viterbi and Baum-Welch.

2 Markov Model

A traditional classification problem is structured as follows: each column is a feature, each row is a set of features, one columns is the class you are going to predict, the prediction is based on the other features in that row. Each row is *independent*, much more different from reality where dependence between rows can help you understanding the problem.

In a markov model it is structured differently because each row is *dependent* on the previous, and there are no features, but just the class we want to predict based on the previous one.

The Markov Property is a mathematical modeling constraint that says that in the model we assume even though a row is dependent on the previous rows it is only dependent on the previous row and nothing else comes before.

We can make an example of a discrete Markov model, called discrete because we can have discrete number of states. If we assume 5 states, consequently we have a 5x5 transition probability matrix representing the probability of the transition from a state to another, included the probability of remaining in the same state.

Example 1:

Assuming that q_t are the states of the system at time t.

$$q_1 = S_3 \; q_2 = S_4 \; q_3 = S_5 \; q_4 = S_4 \; q_5 = S_1 \; q_6 = S_1$$

$$P(q_t = S_z |\; q_{t-1} = S_i, \; \ldots, \; q_2 = S_j, \; q_1 = S_1)$$

This is the full probabilistic description of a given system that would require knowledge of the current state and all previous states, but for the Markov Property explained above we have:

$$P(q_t = S_z |\; q_{t-1} = S_i)$$

Keeping things simple, if we assume that our transition probabilities don't vary over time then:

$$\alpha = P(q_t = S_z |\; q_{t-1} = S_i) \;\; 1 \leq i, j \leq N$$

$$\alpha_{i,j} \geq 0$$

$$\sum_{j=1}^{N} \alpha_{i,j} = 1$$

2.1 *Example of an Observable Markov Model*

Let us imagine that each day at night you check if the daily return of your portfolio composed by 30 assets was positive or negative.

$S_1 = positive\ daily\ return$

$S_2 = negative\ daily\ return$

From day to day the return of your portfolio is in one of those states. Given these states there is a transition matrix A with shape 2x2, because 2 are the states.

$$A = \alpha_{ij} = \begin{bmatrix} 0.3 & 0.7 \\ 0.6 & 0.4 \end{bmatrix}$$

3 Hidden Markov Models

In some situations, the states of a Markov Model could not be observable, but their effect is still measurable. These situation can be analyzed through the Hidden Markov Model.

3.1 *Definition of an Hidden Markov Model*

By definition, an Hidden Markov Model is a Markov Model that assumes that a process of observed states $\{O_t\}_{t=1,2,...T}$ is directly influenced by another process $\{q_t = S_i\}_{t=1,2,...T}$, whose states $S_i \in S = \{S_1, S_2, ...S_N\}$ are the ones we are interested in, and they are latent. For instance, let us go back to the example of the financial market. Each night, you will observe a known state O_t

$O_t = 1 : positive\ daily\ return$

$O_t = 0 : negative\ daily\ return$

Imagine that you are interested in the detection of the trend, for which the market, at each t, can be in a state $q_t = S_i$, with $S_i \in \{S_1, S_2\}$ and

$S_1 : Buy$

$S_2 : Sell$

This state is intuitively hidden. The goal of the Hidden Markov Model is to analyze the observed process to estimate the most likely path of hidden states that has lead to it.

The main problems that Hidden Markov Models try to solve are described by the following three key questions:

- What is the probability that a model generated a sequence of observations?

$$O = O_1, O_2, ...O_T \qquad \lambda = (A, B, \pi)$$
$$P(O|\lambda)?$$
(Forward-Backward Algorithm)

- What sequence of states best explains a sequence of observations?

$$O = O_1, O_2, ...O_T$$
$$Q = q_1, q_2, ..., q_T?$$
(Viterbi Algorithm)

- Given a set of observations sequences, how do we learn the model probabilities that would generate them?

$$O = O_1, O_2, ...O_T$$
$$\lambda = (A, B, \pi)?$$
(Baum-Welch Algorithm)

3.2 *Parameters of the HMM*

An Hidden Markov Model is defined by:

$$HMM = (O, S, \lambda)$$
$$\lambda = (A, B, \pi)$$

where:

O is the observed sequence $\{O_t\}_{t=1,2,...T}$

S is the set of all possible hidden states $\{S_i\}_{i=1,2,...N}$

A is the *state transition matrix*

B is the *emission probability matrix*

π is the vector of *initial probabilities*

3.2.1 *State transition matrix A*

Given N the number of all possible states S_i, the state transition matrix A is defined as an N x N matrix

$$A = \begin{bmatrix} a_{11} & a_{12} & ... & a_{1N} \\ a_{21} & a_{22} & ... & a_{2N} \\ ... & ... & ... & ... \\ a_{N1} & a_{N2} & ... & a_{NN} \end{bmatrix}$$

in which each element

$$a_{ij} = P(q_t = S_j | q_{t-1} = S_i) \qquad 1 \leq i, j \leq N$$

describes the probability of moving from the state S_i to the state S_j, and it doesn't vary over t.

Given that the i_{th} row shows the probabilities to move from S_i to any state (included S_i itself), it is intuitive to get that

$$\sum_{j=1}^{N} a_{ij} = 1 \ \forall i$$

3.2.2 *Emission probability matrix B*

The emission probability matrix, denoted with the parameter B, consists of all the probability distributions of seeing the observations given that you are in a particular state. In the discrete case, given M the number of all possible different observations, B can be formalized as an N x M matrix:

$$B = \begin{bmatrix} b_1(1) & ... & b_1(M) \\ b_2(1) & ... & b_2(M) \\ ... & ... & ... \\ b_N(1) & ... & b_N(M) \end{bmatrix}$$

$$b_j(k) = P(O_t = v_k | q_t = S_j)$$

$$1 \leq j \leq N$$

$$1 \leq k \leq M$$

K represents the different possible observations. The thing we observed at every t in a Markov Model and so the sum of the probabilities be N over all the possible subscripts 1 to M also has to be equal to 1 because we must observe something at every state transition.

3.3 *Forward-Backward algorithm*

As we mentioned before the Forward-Backward algorithm is used in order to calculate the probability of a set of observations given the set of parameters λ. To do it directly is infeasible:

$$P(O|\lambda) = \sum_{q_1,q_2,...,q_T} \pi_{q1} b_{q1}(O_1)\ a_{q_1 q_2}\ b_{q2}(O_2) a_{q_2 q_3}\ \ldots\ a_{q_{T-1} q_T} b_{qT}(O_T)$$

The Forward-Backward algorithm relies on two particular helper functions: *alpha* and *beta*. The first we describe is the forward part: α.

$$\alpha_t(i) = P(O_1, O_2, ...O_T, q_t = S_i|\lambda)$$

The role of alpha is to help us to reduce the number of calculations that we need to make in order to calculate the probability. The base case is:

$$\alpha_1(i) = \pi_i b_i(O_1) \quad 1 \leq i \leq N$$

It means what is the probability of seeing observation 1 given that we are in state i. The inductive step is:

$$\alpha_{t+1}(j) = [\sum_{i=1}^{N}\ \alpha_t(i)\ a_{i,j}]\ b_j(O_{t+1})$$
$$1 \leq t \leq T-1$$
$$1 \leq j \leq N$$

The final step is:

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i)$$

This elegant formula shows that there are some sequence of observations that led up to that time step and by using the α_t variable we can say that we know the probability of being in state S_1 or S_2 given all the observations that came before.
Alpha captures the probability given all the possible ways of getting there.

The same thing is done reasoning and moving *backward*. Given the observation from the time T forward, in other words what we are going to see in the future, we calculate the probability that we started in state T. Obviously, it is something that you can do once you know what the future looks like. This can be formalized as:

$$\beta_t(i) = P(O_{t+1}, O_{t+2}, ...O_T, q_t = S_i|\ \lambda)$$

The base case is:

$$\beta_T(i) = 1 \quad 1 \leq i \leq N$$

The inductive step is:

$$\beta_t(i) = \sum_{j=1}^{N}\ a_{i,j}\ b_j\ (O_{t+1})\ \beta_{t+1}(j)$$
$$t = T-1, T-2, ..., 1$$
$$1 \leq i \leq N$$

3.4 *Viterbi algorithm*

The Viterbi algorithm answers to the second key question: what sequence of states best explains a sequence of observations?

It is a very common question that is asked about Hidden Markov Models because the nature of the observations is that you can observe them, but the things you are most interested in are the states that generated them. Patterson (Patterson)

The best explanation of a sequences of observations is the most likely state at any given time t. This measure can be represented by γ: the probability that we are in state i at time t given the observations O and the parameters λ.

$$\gamma_t(i) = P(q_t = S_i|O, \lambda);$$

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{P(O|\lambda)} = \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=1}^{N}\alpha_t(j)\beta_t(j)};$$

$$\sum_{i=1}^{N}\gamma_t(i) = 1.$$

Hence, the most likely state at given time t will be the state i with the highest associated γ. In order to compute that we should find the argmax over the function γ, looking over all the states from 1 to N:

$$q_t = argmax_{1 \leq i \leq N}[\gamma_t(i)] \quad 1 \leq t \leq T$$

This formula will maximize the expected number of correct state, but it is not what we are looking for. Our desired result is a path sequence that maximizes a whole sequence of states given the observation that we have seen and the model that we have:

$$P(Q|O, \lambda)$$

Since we are interested in maximizing this sequence, it is equivalent to maximize the probability of a state sequence and an observation sequence given our model. So we are answering to the following question: what is the path with the highest probability that accounts for the first t observations and ends at state S_i?

$$\delta_t(i) = max_{q_1,q_2,...,q_{t-1}} P((q_1, q_2, q_3, ..., q_t = i), (O_1, O_2, O_3, ..., O_t)|\lambda)$$

We can solve this using induction. The basic inductive step is listed here this:

$$\delta_{t+1}(j) = [max_i \ \delta_t(i) \ a_{i,j}] \cdot b_j(O_{t+1})$$

It means that we want to figure out what is the best sequence of state going from t to $t+1$. In other words what is the state that maximized delta at the previous time step i, and all the ways of getting from i to j, that is the current state.
So $\delta_{t+1}(j)$ is going to find the previous state i that maximizes the transition from it to the state we care about now j. It is a kind of search over all the previous states in order to figure out which one is the best and then use his probability multiplied by $a_{i,j}$ (transition probability) and $b_j(O_{t+1})$ (emission probability). Matthew F. Dixon (2020) This is the inductive step!

To sum up the Viterbi Algorithm is structured as follows:

- Initialization:

$$\delta_1(i) = \pi_i \ b_i(O_i)$$

$$\psi_1(i) = 0 \ state \ initialization$$

- Inductive step:

$$\delta_t(j) = [max_{1 \leq i \leq N} \ \delta_{t-1}(i) \ a_{i,j}] \cdot b_j(O_t)$$
$$2 \leq t \leq T$$

$$\psi_t(j) = argmax_{1 \leq i \leq N} \ \delta_{t-1}(i) \ a_{i,j}$$
$$1 \leq j \leq N$$
(where $\psi_t(j)$ keeps track all the state we pass through at each t)

- Termination step:

$$P^* = max_{1 \leq i \leq N} \ [\delta_T(i)]$$

$$q_T^* = argmax_{1 \leq i \leq N} \ [\delta_T(i)]$$

It is curious to see how similar is the structure and the way of calculation between $\delta_t(j)$ and $\alpha_t(j)$ in the Forward-Backward Algorithm.

3.5 *Estimation of the parameters:* the Baum-Welch algorithm

We have seen how the Viterbi algorithm uses the forward and the backward steps to compute the most likely sequence of hidden states, given the observations and the parameters. However, in practical situations, knowing the parameters of the model apriori is a very rare situation. Most of the times, the transition matrix A, the emission probability matrix B and the initial probability vector π need to be inferred in order to fit the model. This is the role of the Baum-Welch Algorithm. Martin (Martin) Rabiner and Juang (1986)

Baum-Welch is an extension to what we have seen before. It is algorithm that belongs to the family of the Expectation-Maximization methods. This means that it recursively runs:

- an estimation step, in which the Forward-Backward algorithm and the Viterbi algorithm are used to compute α, β and γ;

- an expectation maximization step, in which estimates for $\hat{\lambda} = (\hat{A}, \hat{B}, \hat{\pi})$ are updated.

Baum-Welch is a local optimization method. Unfortunately, there is no known way to find globally optimal parameters and, because of that, initialization is a very important step as different initialization can lead to extremely different results.

So, the first step is to set initial values for $\hat{\lambda} = (\hat{A}, \hat{B}, \hat{\pi})$. These values can be found in different ways. One could be to approximate them by analyzing the sequence of observations and use values that seem to be appropriate. A good strategy could be to run Baum-Welch algorithm several times for different initialization parameters, and then optimize them by maximizing a likelihood function or minimizing a loss function. This method has the only disadvantage to be very computationally expensive.

After setting the initial parameters, we need to introduce a new quantity:

$$\xi_t(i, j) = P(q_t = S_i, q_{t+1} = S_j | O, \lambda).$$

$\xi_t(i, j)$ is the joint probability that we are in state S_i at time t and we will be in state S_j at time $t + 1$, given our observations and parameters. Our ξ will be a 3-dimensional matrix of dimension T x N x N, in which we will have, at each time t, a matrix showing the probability to be in a state S_i and move to a state S_j. From a mathematical point of view, this can be formalized by:

$$\xi_t(i, j) = \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{P(O|\lambda)}$$

The denominator is a sort of normalizing factor, in order to get probabilities. We calculate it simpy by iterating for all i and j, and so it can be written as

$$P(O|\lambda) = \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j).$$

And so the final equation becomes:

$$\xi_t(i, j) = \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}.$$

Now we can explore an interesting fact. If we sum up $\xi_t(i, j)$ for all the possible destinations j, we can discover a clear relation between ξ and the γ computed in the Viterbi algorithm. In fact, we've seen that:

- $\gamma_t(i)$ is the probability of being in state S_i at time t. So, if we sum up over all t we obtain a quantity that can be treated as the expected number of times the process passes through S_i;

- $\xi_t(i,j)$ is the probability to move from S_i to S_j at time t. So, if we sum up over all t we obtain a quantity that can be treated as the expected number of times the process moves from S_i to S_j;
- then it's intuituve that if we sum up $\xi_t(i,j)$ over all j, we obtain the expected number of times the process moves from S_i to anywhere, and so the expected number of times the process passes through S_i, that is exactly $\gamma_t(i)$

To formalize it:

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j)$$

So, now we got:

- an existing model with parameters $\lambda = (A, B, \pi)$
- a set of observations O
- a set of tools $\alpha_t(i)$, $\beta_t(i)$, $\gamma_t(i)$, $\xi_t(i,j)$

and we need to use them to improve our model and get $\hat{\lambda}$.

The new estimate for the initial probability vector $\hat{\pi}$ is simply the expected frequencies in S_i at time $(t = 1)$:

$$\hat{\pi} = \gamma_1(i)$$

Then, we can calculate our updated transition matrix $\hat{A}$ as the ratio, for all possible i and j, between the expected number of transitions from S_i to S_j and the expected number of transition from S_i to anywhere, namely

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}$$

Finally, we need to improve our observation probabilities $\hat{B}$, and this will be the matrix containing the ratios between the expected number of times in state j observing some value v_k, over the expected number of times in state j at all. Formalized,

$$\hat{b}_j(k) = \frac{\sum_{t=1}^{T} [\gamma_t(i) \mathbb{1}(O_t = v_k)]}{\sum_{t=1}^{T} \gamma_t(i)}$$

We finally have our estimates and we can fit a new model with them.

To summarize:

- Given $\lambda = (A, B, \pi)$ and O, we can produce $\alpha_t(i), \beta_t(i), \gamma_t(i), \xi_t(i, j)$
- Given $\alpha_t(i), \beta_t(i), \gamma_t(i), \xi_t(i, j)$, we can produce $\hat{\lambda} = (\hat{A}, \hat{B}, \hat{\pi})$ and substitute them to $\lambda = (A, B, \pi)$

Running these two steps recursively, the estimates will converge to a local optimum, that will be the parameters of our final and definitive Hidden Markov Model.

4 Implementation

We try to apply the above described algorithms in order to detect the trend of a portfolio composed by 30 stocks, randomly selected from the basket SP500. Perhaps, we will present the empirical and numerical analysis of the time serie taken into consideration, measures of financial performances such as: Annualized returns, Volatility, Sharpe Ratio, Sortino Ratio and Max Drawdown. And it will be shown a walk-forward backtest in order to compare the only-long strategy and two strategies taking the Buy-Hold and Buy-Sell signal from the Hidden Markov Model. Origin (Origin)

4.1 Data Preparation

The data are downloaded from the Yahoo Finance API. Firstly, we extract the tickers composition of stocks of the index SP500 and we select randomly from that basket 30 companies. Origin (Origin) Then, for each stock we download the close price and we sum all the prices for each day and we obtain the portfolio price from 2001 to today. After that we compute the daily return of the portfolio and we assign a value of 0 if that return is negative and 1 if that portfolio is positive. Next we initialize two hidden state: Buy and Sell. Origin (Origin)

4.2 Algorithms Implementation

The Algorithms described above are implemented from scratch following the theoretical formulas without any change. For further details see the code in the following GitHub page. Origin (Origin)

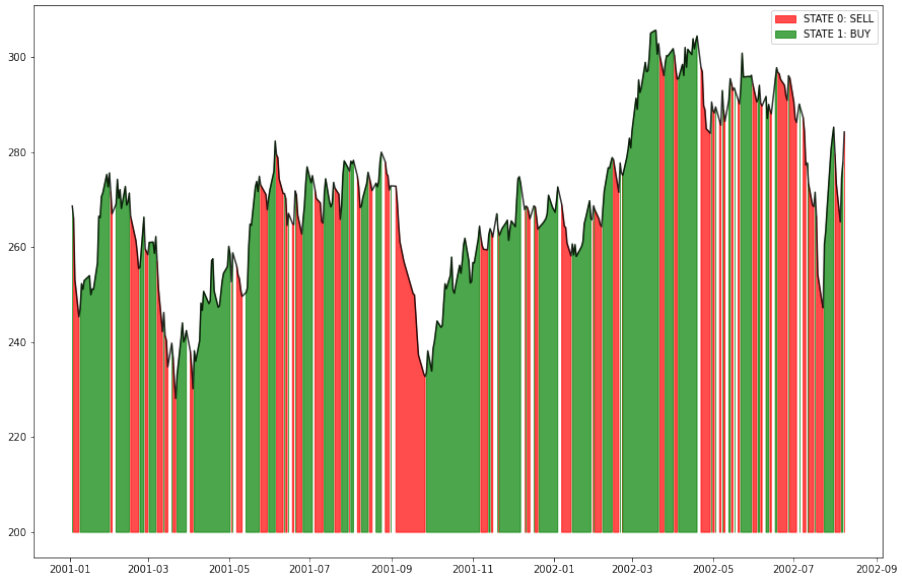


Figure 1: 11th September 2001

4.3 Results

First results we want to show is how good the Hidden Markov model is able to understand periods in which the price is decreasing and associates a State Sell and periods in which the price is increasing and consequently associates a State Buy. Origin (Origin) We will present three significant periods for the financial markets in windows of 300 days:

- 11th September 2001
- Subprime mortgage crisis: 2008-2009
- Covid: 2020 stock market crash.

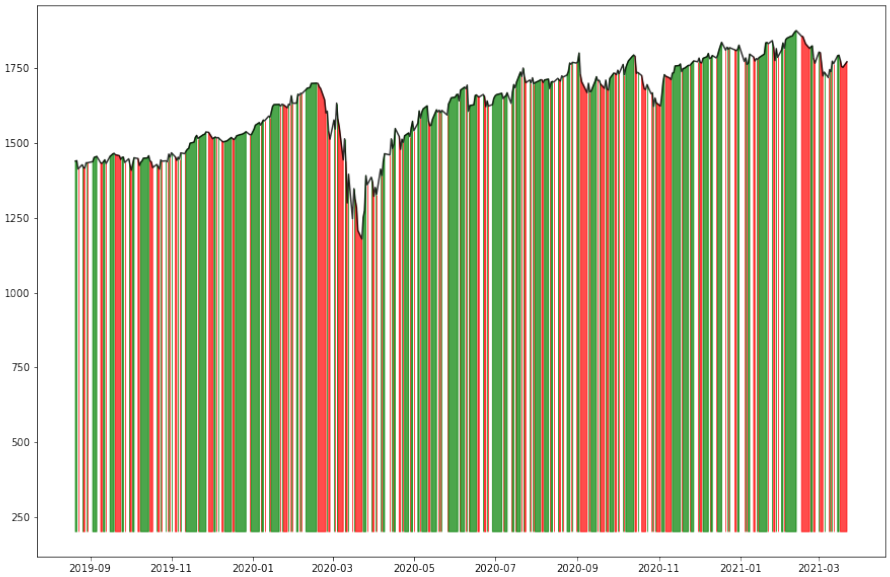


Figure 2: Covid: 2020 stock market crash

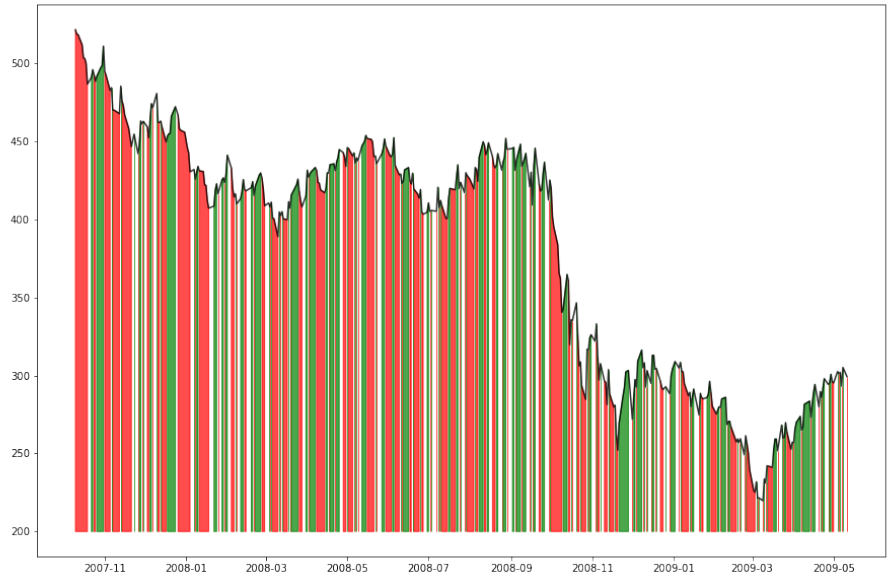
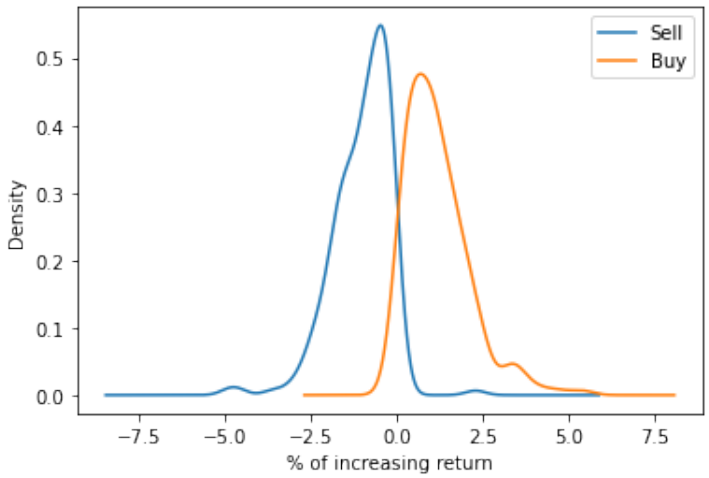


Figure 3: Subprime mortgage Crisis: 2008-2009

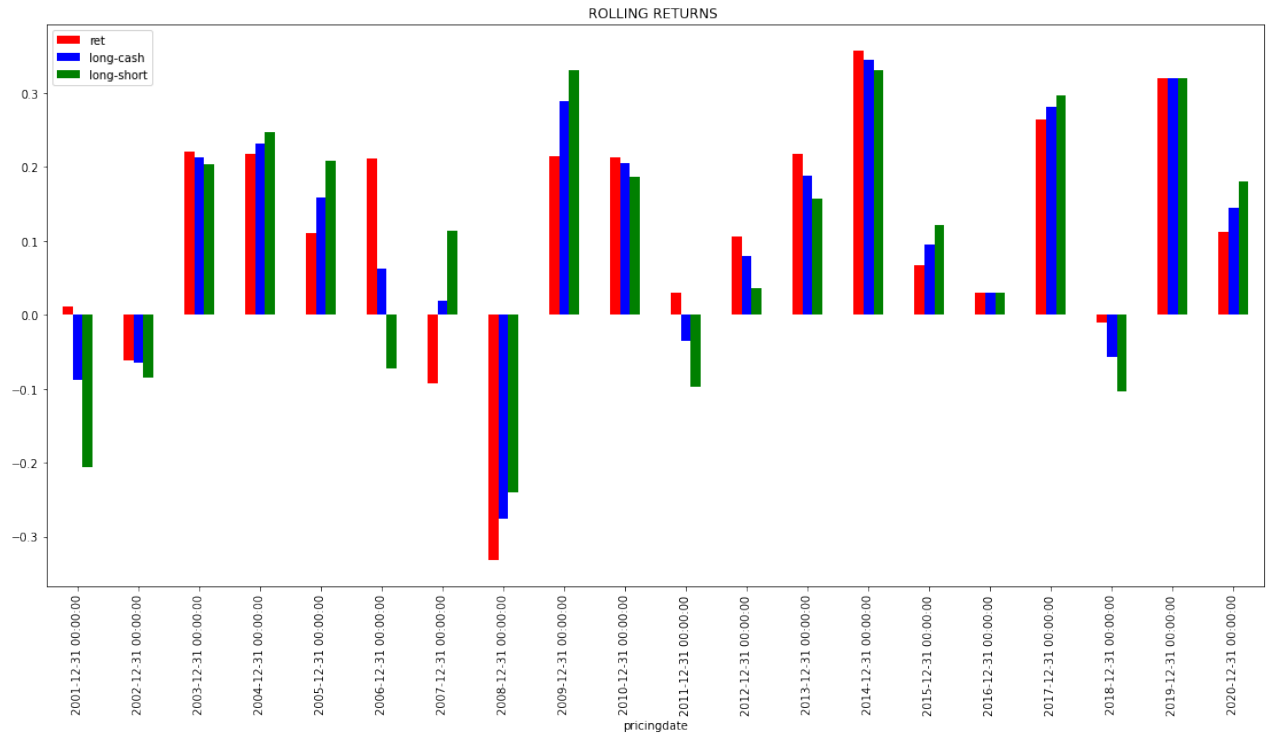
With the density plot we can see that the returns conditioned by each state are distributed approximately Normal. We can observe also that negative returns are associated with a Sell State and positive returns with a Buy State. Zhang et al. (2018) Rogemar S. Mamon (2014)



The figure below shows the compound annual growth rate (CAGR) of the three strategy: Only Long, Long-Cash, Long-Short. The last two are following the prediction 0-1 of the states, 1 means going long for both, 0 means going cash (not be invested) in the long-cash strategy and short (be invested against the market) in the long-short strategy. Origin (Origin)

Following the same colors you can see also the main financial annualized performances and the rolling return bars.

```
Return: 9.48
Volatility: 19.45
Sharpe Ratio: 0.56
Sortino Ratio: 0.8
Calmar Ratio: 0.22
Max Draw-Down: -43.97 Start: 09/14/2007 End: 03/05/2009
Return: 8.24
Volatility: 21.66
Sharpe Ratio: 0.47
Sortino Ratio: 0.67
Calmar Ratio: 0.17
Max Draw-Down: -49.51 Start: 03/13/2001 End: 08/07/2002
Return: 9.74
Volatility: 21.66
Sharpe Ratio: 0.54
Sortino Ratio: 0.77
Calmar Ratio: 0.16
Max Draw-Down: -59.11 Start: 02/20/2007 End: 03/09/2009
```



5 Conclusion

We have seen how the Hidden Markov models work on theory and on practice. The main consideration about the empirical analysis are the following: they are not outperforming the only-long strategy all over the period taken into consideration but they are helping reducing the volatility and the Max Draw-Down, consequently the measure linked to them are improving (Sharpe, Sortino, Calmar ratios).

The major contribution of this model is that we can detect approximately the trend, avoiding periods in which the returns are negative (Long-Cash) or exploiting these periods in which the returns should be negative to gain betting against the market (Long-Short), even if it is always riskier than divest.

For further studies and researches the model can be improved adding more features (using weekly, monthly or quarterly returns), and the trading strategy can be diversified and strengthened with the help of more than just one signal.

References

Martin, D. J. . J. H. Standford tutorial on hidden markov models.

Matthew F. Dixon, Igor Halperin, P. B. (2020). *Machine Learning in Finance: From Theory to Practice*. Springer New York, NY. Purchasable at `https://doi.org/10.1007/978-3-030-41068-1`.

Origin, G. Github project on hidden markov models. `https://github.com/Espo45/hmm_unibo`.

Patterson, P. Hidden markov models 10: motivating the viterbi algorithm. `https://www.youtube.com/watch?v=JRsdt05pMoI&t=856s`.

Rabiner, L. and B. Juang (1986). An introduction to hidden markov models. *IEEE ASSP Magazine 3*(1), 4–16.

Rogemar S. Mamon, R. J. E. (2014). *Hidden Markov Models in Finance*. Springer New York, NY. Purchasable at `https://doi.org/10.1007/978-1-4899-7442-6`.

Zhang, M., X. Jiang, Z. Fang, Y. Zeng., and K. Xu (2018). High-order hidden markov model for trend prediction in financial time series.