Economic regimes identification using machine learning technics

by

Mamed Caki

A thesis submitted in conformity with the requirements
for the MSc in Economics, Finance and Computer Science

University of Huelva & International University of Andalusia

September/November 2018

Economic regime identification using machine learning technics

Mamed Caki

Máster en Economía, Finanzas y Computación

Dr. José Manuel Bravo Caro
Universidad de Huelva y Universidad Internacional de Andalucía

2018

Abstract

Economic conditions over long time periods can be distinguished by *regimes*. Regime identification has been object of numerous investigations in economics and financial modeling for years.

Recently, new machine learning technics such as decision trees, support vector machines and neural networks, among others, followed by alternative datasets and cheap computational processing power became available, allowing for alternative ways to model complex economic relationships.

In the present work, we develop a supervised machine learning classifier using *Random Forest* technic to identify economic regimes using the S&P 500 stock market index series.

Key words: economic regimes, machine learning, random forest, Hidden Markov Model.

Resumen

Las condiciones económicas durante largos períodos de tiempo pueden distinguirse por regímenes. La identificación del régimen ha sido objeto de numerosas investigaciones en economía y modelos financieros durante años.

Recientemente, se pusieron a disposición nuevas técnicas de aprendizaje automático, como árboles de decisión, máquinas de suporte vectorial y redes neuronales, entre otras, seguidas de conjuntos de datos alternativos y una capacidad de procesamiento computacional barata, que permite formas alternativas de modelar relaciones económicas complejas.

En el presente trabajo, desarrollamos un clasificador de aprendizaje automático supervisado utilizando la técnica de *Random Forest* para identificar regímenes económicos utilizando la serie del índices de mercado S&P 500.

Palabras clave: regímenes económicos, aprendizaje automático, *random forest*, Cadenas de Markov.

Table of Contents

1 Introduction

Economic conditions over long time periods can be distinguished by *regimes*. Regime identification has been object of numerous investigations in economics and financial modeling for years.

Conventional statistical and econometric technics have been used to model economic fluctuations. Hamilton's (1989) seminal work proposed the use of hidden Markov models (HMM) to characterize business cycles in a probabilistic framework. This approach was followed by a number of studies in areas as macroeconomic shifts, stock prices, foreign exchange, interest rates, asset allocation and portfolio and risk management – see e.g. Sheikh et al. (2012), Ang et al. (2011), Bulla (2011), Duprey et al. (2017), Guidolin et al. (2007), Nystrup et al. (2015), Mulvey et al. (2014).

Recently, new machine learning technics such as decision trees, support vector machines and neural networks, among others, followed by alternative datasets and cheap computational processing power became available, allowing for alternative ways to model complex economic relationships (Varian, 2014).

In the present work, we develop a supervised machine learning classifier using *Random Forest* technic to identify economic regimes using the S&P 500 stock market index series. Regime characterizations are derived in three ways: (a) two-state positive vs. negative returns, (b) two-state positive vs. negative volatility trend and (c) three-state regime produced by the combination of (a) and (b).

Distinct features are built using a number of terms – from two weeks to three years. The produced models are selected using cross-validation tests, and detailed results are presented for the best ones.

Lastly, we perform out-of-sample dynamic investment simulations using the produced models contrasting it's results to the HMM approach.

2 Random forest

2.1 Decision trees classifiers

Decision trees are algorithms that recursively segment the predictor space into several sub-spaces, and then fit a simple model in each of the produced regions (Friedman et al., 2001). It may be used as a supervised learning method both to regression, where outputs are real numbers, and classification problems, where outputs are categorical. In the present study, we develop a tree classifier, having regime labels, or economy states, as output.

Starting from the total predictor space, one predictor variable and one split-point in its domain is selected to generate a binary partition. For each one of the resulting partitions, an *impurity* measure is evaluated. This process is repeated for every split-point of every predictor variable. The optimal decision of which partition to be made in which predictor variable is based on the best results, i.e., the one that minimizes the weighted *impurity* mean of both produced regions. This process then is recursively repeated at each partition until some stopping rule is applied. This model can be represented by the binary tree at Figure 2.1, where splits of predictor variables X_1 and X_2 produced regions R_1 to R_5.

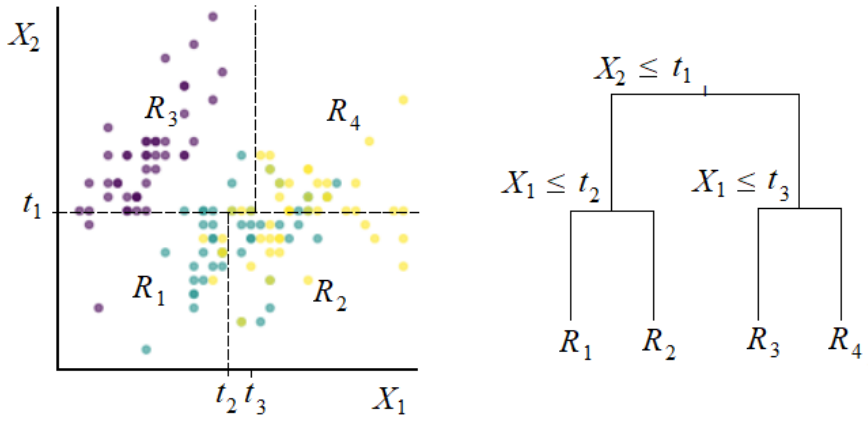


Figure 2.1. *Three classes binary tree representation (right) of the predictor space* (X_1, X_2) *(left).*

If the classification outcome takes the values $k = 0, 1, \ldots, K-1$, for the region R_m with N_m observations, we define the proportion of class k observations as:

$$p_{mk} = \frac{1}{N_m} \sum_{x_i \, \epsilon \, R_m} I(y_i = k)$$

Then we classify the observations in R_m as $k(m) = arg\, max_k\, p_{mk}$, i.e., the majority class in this region. One first simple *impurity* metric may be the mean misclassification error, computed as the proportion of observation that does not match the majority class $k(m)$:

$$Misclassification\ error_{R_m} = \frac{1}{N_m} \sum_{i \, \epsilon \, R_m} I(y_i \neq k(m)) = 1 - p_{mk}(m)$$

Another *impurity* method is the cross-entropy, defined as:

$$Cross - entropy_{R_m} = -\sum_{\mathrm{k}} p_{mk} \log(p_{mk})$$

Figure 2.2 compare both metrics as a function of the proportion of class "2" observations in a two-class problem. Besides being differentiable, cross-entropy is more sensitive to changes in the probabilities than the misclassification error (its derivate is bigger). Therefore, we use the cross-entropy to fit the tree.

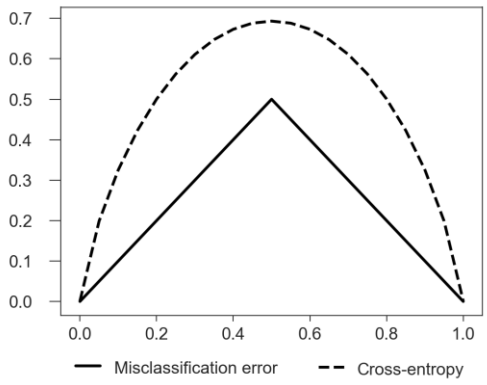


Figure 2.2. *Misclassification error and cross-entropy comparison for a two-class classification example as a function of the proportion p in class "2" observations (adapted from Friedman et al., 2001).*

As a stopping rule for the tree growth we use tree size, i.e., its maximum depth. This parameter determines the complexity of the model and must be tuned using out of sample data in order to avoid overfitting (see Section 8). Mode details on other decision trees parameters may be found at Friedman et al., 2001.

2.2 Random forests

Decision trees are a low bias and high variance estimator. The hierarchical nature of the process makes top nodes splits errors propagate down to bottom nodes. One way to minimize this problem and improve the prediction accuracy is to produce a set of different trees from different training sets and averaging the results. This procedure is known as *bootstrap aggregation*, or *bagging*.

In the *Random Forest* algorithm, bagged trees are produced and averaged. Furthermore, another layer of randomness is added: at each split step for each bagged tree, only $m < p$ predictors are considered to compute the best split. This procedure *de-correlates* the trees, turning the averaged prediction result less variable.

In the present work, we adopt the number of bootstrapped sets of one thousand ($n_{bt} = 1000$) and the number of randomly selected predictors at each split of one ($m = 1$), meaning that we average 1000 single trees, all of them grown with only one randomly selected variable at each node split. This configuration permits a small correlation between pairs of trees, reducing the likelihood of having similar trees, as they do not use the same set of splitting variables.

3 Data

In the context of efficient markets, financial asset prices instantly reflect agents economic outlook. The S&P 500 is a stock market index based on the capitalization of 500 American large companies, and widely followed by worldwide market participants.

It is the leading economic indicator with the best track record at identifying recessionary troughs before they occur (Renshaw, 2002). We use S&P 500 daily index time series from August 1992 to August 2018 containing 5040 observations to build the model, and from August 2013 to August 2018 to compare it with the HMM model (*backtesting*). Figure 3.1 shows the level and daily return.

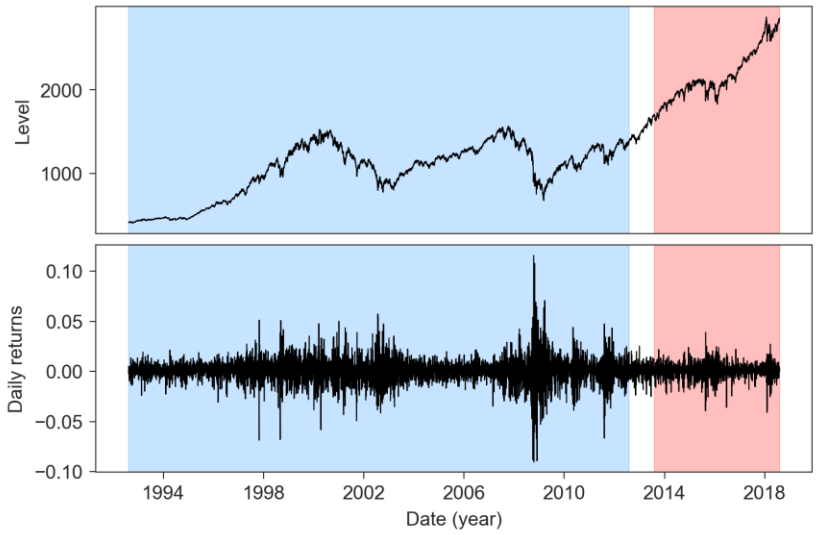


Figure 3.1. *S&P 500 level (top) and daily return (bottom) from 1992-08-08 to 2018-08-08. The blue shade represents the period used to tune the model, while the red shade represents the period used to backtest it.*

4 Regime characterization

Three distinct economic regimes characterizations derived from the S&P 500 data will be used in the *Random Forest* classifier. Each regime, also referred from now as *state*, *outcomes* or *labels*, will be treated as categorical response variables in the context of the supervised learning approach.

4.1 Two-state return regimes

In the *two-state return regimes* representation, the outcome is represented by (i) positive or (ii) negative price returns. For each day t, it is computed over the future price information in a pre-defined timespan T:

$$futReturn_{t,T} = \frac{Price_{t+T}}{Price_t} - 1$$

$$retL_{t,T} = \begin{cases} State\ 1, & if\ futReturn_{t,T} \geq 0 \\ State\ 2, & if\ futReturn_{t,T} < 0 \end{cases}$$

where $Price_t$ is the S&P 500 Index level at time t, $futReturn_{t,T}$ is the price return between t and $t + T$, and $retL_{t,T}$ is the regime label at time t with timespan T. The timespan T is defined in trading days from the periods of two weeks to one year, according to Table 4.1.

Timespan	Symbol	Trading days
two weeks	*WW*	10
month	*M*	21
quarter	*Q*	63
semester	*S*	126
year	*Y*	252

Table 4.1. *Timespan T from two weeks to one year, represented in trading days.*

$retL_{t,T}$ is computed daily. Figure 4.1 shows $futReturn_{t,T}$ for every T with its respective distribution. The shaded areas represent the $State\ 2$ label. Table 4.2 brings the basic statistics.

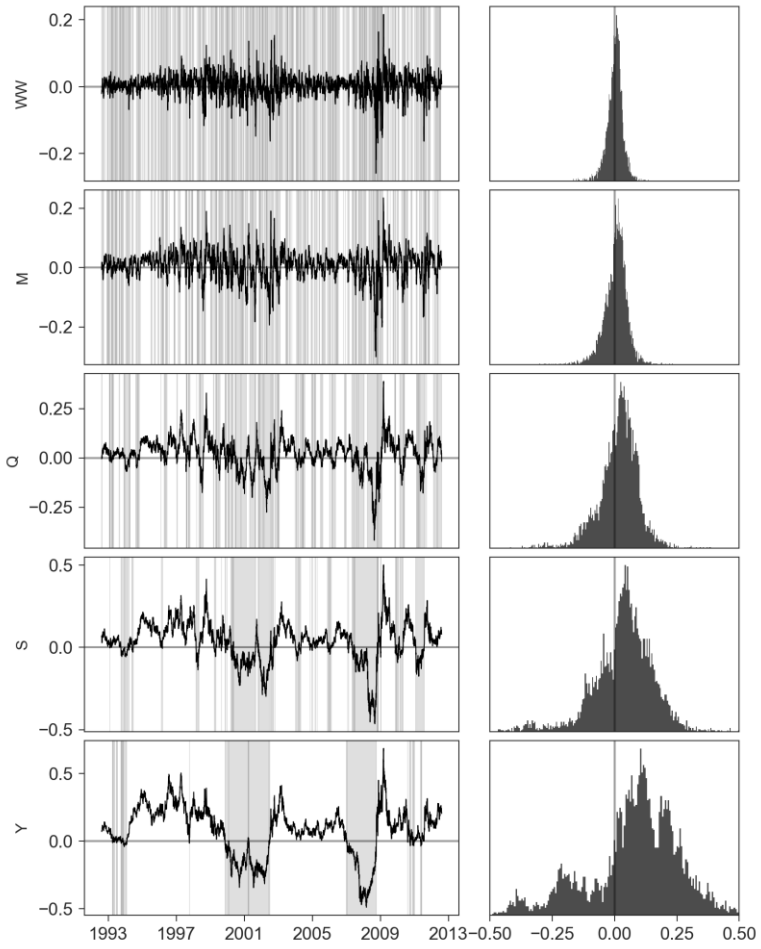


Figure 4.1. *Return outcomes* $futReturn_{t,T}$ *(left, solid line) with respective histograms (right). Shaded areas of the left-hand side plot represent the negative return labels of* $retL_{t,T}$*, while white areas represent the positive return label.*

Time span	μ	σ	N. obs. $State\ 1$	N. obs. $State\ 2$
WW	0.0030	0.0328	2948 (58%)	2092 (42%)
M	0.0062	0.0467	3135 (62%)	1905 (38%)
Q	0.0188	0.0795	3304 (66%)	1736 (34%)
S	0.0384	0.1190	3525 (70%)	1515 (30%)
Y	0.0815	0.1812	3826 (76%)	1214 (24%)

Table 4.2. $Return_{t,T}$ *mean and standard deviation for each timespan* T*, and number of observations of* $State\ 1$ *and* $State\ 2$*.*

4.2 Two-state volatility trend regimes

In the *two-state volatility trend regimes* representation, the outcome is represented by (i) positive or (ii) negative volatility trends. For each day t, it is computed over a pre-defined time span T:

$$volTrend_{t,T} = \frac{DailyStd_{\{t+1,t+T\}}}{DailyStd_{\{t-T,t\}}} - 1$$

$$volTrendL_{t,T} = \begin{cases} State\ 1, & volTrend_{t,T} \geq 0 \\ State\ 2, & volTrend_{t,T} < 0 \end{cases}$$

where $DailyStd_{\{i,j\}}$ is the S&P 500 Index daily returns standard deviation between times i and j, and $volTrendL_{t,T}$ is the regime state at time t for the timespan T defined in Table 4.1. $volTrendL_{t,T}$ is computed daily. Figure 4.2 shows $volTrend_{t,T}$ for every T with its respective distribution. The shaded areas represent the $State\ 2$ label. Table 4.3 brings the basic statistics.

Time span	μ	σ	N. obs. $State\ 1$	N. obs. $State\ 2$
WW	0.0842	0.4682	2473 (49%)	2567 (51%)
M	0.0681	0.4120	2466 (49%)	2574 (51%)
Q	0.0708	0.4215	2405 (48%)	2635 (52%)
S	0.0783	0.4208	2292 (45%)	2748 (55%)
Y	0.0920	0.4262	2625 (52%)	2415 (48%)

Table 4.2. *$Return_{t,T}$ mean and standard deviation for each timespan T, and number of observations of $State\ 1$ and $State\ 2$.*

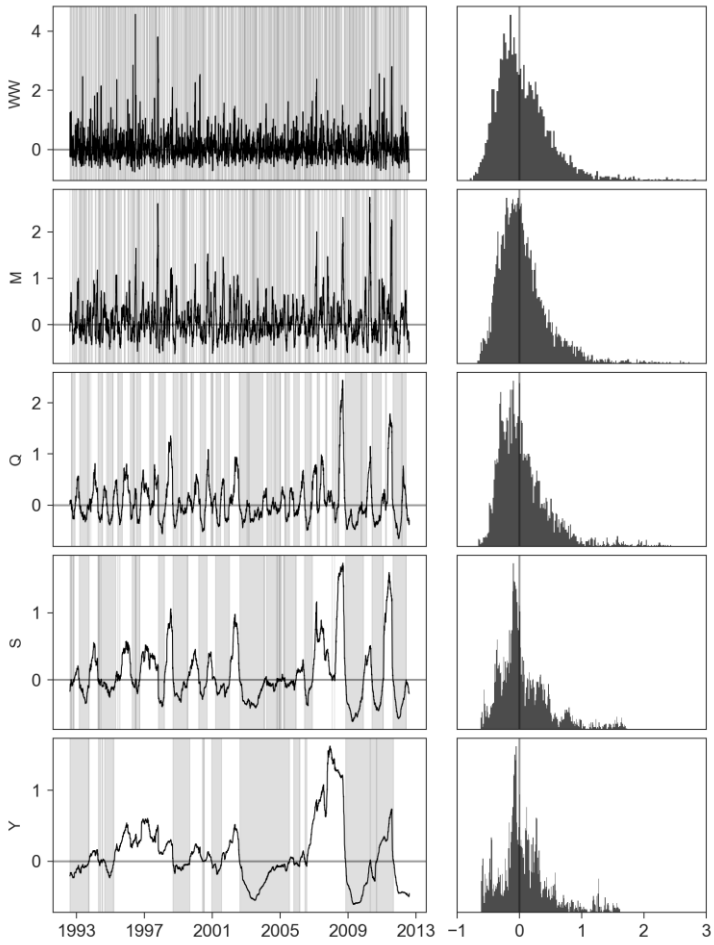


Figure 4.2. *Volatility trend outcomes* $volTrend_{t,T}$ *(left, solid line) with respective histograms (right). Shaded areas of the left-hand side plot represent the negative volatility trend labels of* $volTrendL_{t,T}$*, while white areas represent the positive volatility trend label.*

4.3 Three-state return and volatility trend regimes

In the *three-state return and volatility trend regimes* representation, the outcome is represented by the combination of the previous characterizations: (i) negative return and positive volatility trend; (ii) positive return and positive volatility trend, or negative return and negative volatility trend; or (iii) positive return and negative volatility trend. For each day t, it is computed over a pre-defined time span T:

$$retVolL_{t,T} = \begin{cases} State\ 1, & futReturn_{t,T} < 0\ and\ volTrend_{t,T} \geq 0 \\ State\ 2, & futReturn_{t,T} \geq 0\ and\ volTrend_{t,T} \geq 0\ or \\ & futReturn_{t,T} < 0\ and\ volTrend_{t,T} < 0 \\ State\ 3, & futReturn_{t,T} \geq 0\ and\ volTrend_{t,T} < 0 \end{cases}$$

where $retVolL_{t,T}$ is the regime state at time t with time span T. Figure 4.3 shows $futReturn_{t,T}$, $volTrend_{t,T}$ and $retVolL_{t,T}$ for every T with its respective distribution, while Table 4.3 brings the regime states frequencies.

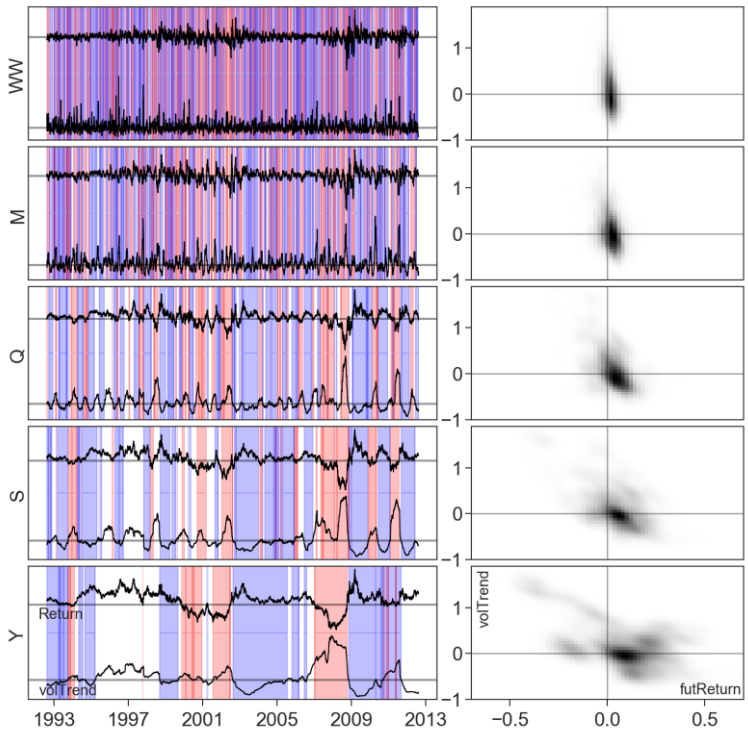


Figure 4.3. *$futReturn_{t,T}$and $volTrend_{t,T}$ are plotted (left, solid lines) with respective joint frequency histogram (right). Red shaded areas of the left-hand side plot represent $State\ 1$, white areas represent $State\ 2$, and blue shaded areas $State\ 3$.*

Time span	N. obs. $State\ 1$	N. obs. $State\ 2$	N. obs. $State\ 3$
WW	1279 (25%)	2007 (40%)	1754 (35%)
M	1207 (24%)	1957 (39%)	1876 (37%)
Q	1192 (24%)	1757 (35%)	2091 (41%)
S	1095 (22%)	1617 (32%)	2328 (46%)
Y	1042 (21%)	1755 (35%)	2243 (45%)

Table 4.3. $retVolL_{t,T}$ states frequencies *for each timespan* T.

5 Predictor variables

Once we have characterized the three outcomes $retL_{t,T}$, $volTrendL_{t,T}$ and $retVolL_{t,T}$, now we will define the predictor variables, also referred as *features*. Differently of the outcomes, which are function of future information, the features are based only in past information.

5.1 Return feature

The *return feature* is computed based on the past price information over a pre-defined timespan T. For each day t, it is computed as:

$$Return_{t,T} = \frac{Price_t}{Price_{t-T}} - 1$$

where $Price_t$ is the S&P 500 Index at time i. Differently from the timespans defined for the outcomes, where it ranges from two weeks (*WW*) to one year (*Y*), the return feature has three more periods added (one day, "*D"*, two years, "YY*"*, and three years, "YYY*"*), as shown in Table 5.1.

Timespan	Symbol	Trading days
day	*D*	1
two weeks	*WW*	10
month	*M*	21
quarter	*Q*	63
semester	*S*	126
year	*Y*	252
two years	*YY*	504
three years	*YYY*	756

Table 5.1. *Timespan T from one day to three years, represented in trading days.*

5.2 Volatility feature

The *volatility feature* is the past standard deviation over the timespan T, with T ranging from two weeks (*WW*) to three years (*YYY*). For each day t, it is computed as:

$$Vol_{t,T} = Std\{Price_{t-T}, \dots, Price_t\}$$

5.3 Price trend feature

The *price trend feature* is defined as the ratio between average prices in two different past periods:

$$priceTrend_{t,T_1,T_2} = \frac{mean\{Price_{t-T_1}, \dots, Price_t\}}{mean\{Price_{t-T_1-T_2}, \dots, Price_{t-T_1}\}} - 1$$

with T_1 and T_2 defined for $(T_1, T_2) = \{T_2 > T_1 \mid T_1, T_2 \in \{D, WW, M, Q, S, Y, YY, YYY\}\}$, as shown Table 5.2. Figure 5.1 illustrates the *price trend feature* for $(T_1, T_2) = \{(D, M), (M, Y), (S, YYY)\}$.

T_1 \ T_2	*WW*	*M*	*Q*	*S*	*Y*	*YY*	*YYY*
D	x	x	x	x	x	x	x
WW		x	x	x	x	x	x
M			x	x	x	x	x
Q				x	x	x	x
S					x	x	x
Y						x	x
YY							x

Table 5.2. T_1 and T_2 for the $priceTrend_{t,T_1,T_2}$ *feature.*

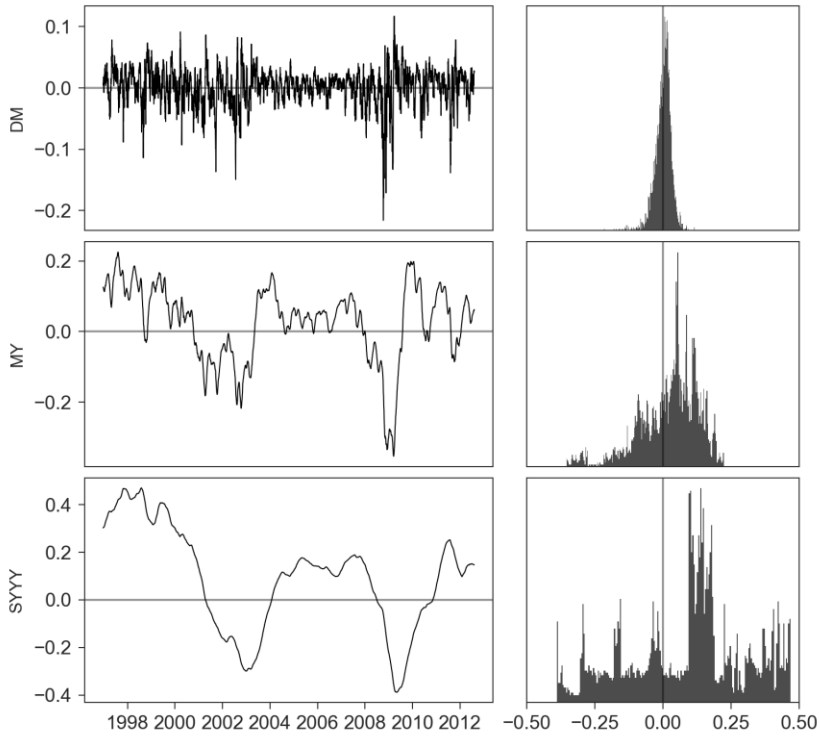


Figure 5.1. $priceTrend_{t,T_1,T_2}$ *for* $(T_1, T_2) = \{(D, M), (M, Y), (S, YYY)\}$ *(left) and respective histograms (right).*

5.4 Volatility trend feature

For each day t, the *volatility trend feature* is defined by the ratio between daily returns standard deviations in two different past periods:

$$volTrend_{t,T_1,T_2} = \frac{Std\{Price_{t-T_1}, \dots, Price_t\}}{Std\{Price_{t-T_1-T_2}, \dots, Price_{t-T_1}\}} - 1$$

with T_1 and T_2 defined for $(T_1, T_2) = \{T_2 > T_1 \mid T_1, T_2 \epsilon \{WW, M, Q, S, Y, YY, YYY\}\}$, as shown Table 5.3. Figure 5.2 illustrates the *volatility trend feature* for $(T_1, T_2) = \{(WW, S), (S, Y), (Y, YYY)\}$.

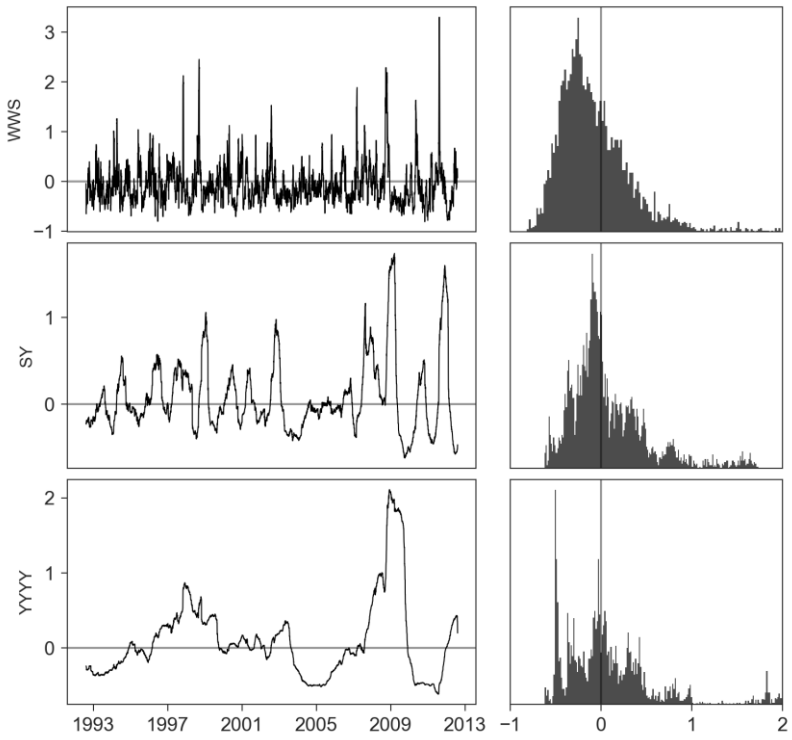


Figure 5.2. $volTrend_{t,T_1,T_2}$ *for* $(T_1, T_2) = \{(WW, S), (S, Y), (Y, YYY)\}$ *(left) and respective histograms (right).*

T_1 \ T_2	*M*	*Q*	*S*	*Y*	*YY*	*YYY*
WW	x	x	x	x	x	x
M		x	x	x	x	x
Q			x	x	x	x
S				x	x	x
Y					x	x
YY						x

Table 5.3. T_1 and T_2 for the $volTrend_{t,T_1,T_2}$ *feature.*

6 Concurrent outcomes

In Section 4, we defined three distinct regime characterizations: $retL_{t,T}$, $volTrendL_{t,T}$ and $retVolL_{t,T}$. Each one of them is a function of the S&P 500 Index returns in some time interval.

Let's assume y_i as some label generated as function of prices over the interval $[t_{i,0}, t_{i,1}]$. When $i < j$ and $t_{i,1} < t_{j,0}$, y_i and y_j will depend on common information. That is, the series $\{y_i\}_{i=1,\ldots,N}$ is not independent and identically distributed (IID) whenever there is an overlap between any two outcomes (Prado, 2018). Figure 6.1 illustrates this mechanism for two consecutive outcomes.

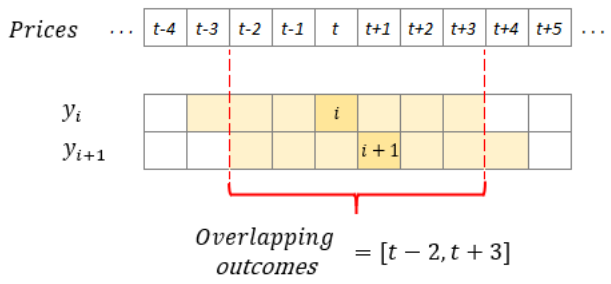


Figure 6.1. *Two consecutive outcomes with generated from concurrent price information.*

6.1 Concurrent outcomes and average uniqueness

When two outcomes are functions of the same price information $p(t)$, they are said to be concurrent at t (Prado, 2018). For a given label y_i, $i = 1,\ldots,I$, function of prices $p(t)$ in the interval $[t_{i,0}, t_{i,1}]$, we may compute the number of concurrent outcomes at each $t = 1,\ldots,T$ by:

$$c_t = \sum_{i=1}^{I} 1_{t,i}$$

where

$$1_{t,i} = \begin{cases} 1, & if \quad t \in [t_{i,0}, t_{i,1}] \\ 0, & if \quad t \notin [t_{i,0}, t_{i,1}] \end{cases}$$

Inversely, we may also compute the uniqueness of outcome i at each t as $u_{t,i} = 1_{t,i} c_t^{-1}$. And finally, the average uniqueness of outcome i is the average $u_{t,i}$ over the entire outcome lifespan, $\bar{u}_i = \left(\sum_{t=1}^{T} u_{t,i}\right)\left(\sum_{t=1}^{T} 1_{t,i}\right)^{-1}$.

For each computed label in Section 4, we can calculate the mean average uniqueness ($avgU$) at each term T. Table 6.1 has the results:

Term	$retL_{t,T}$	$volTrendL_{t,T}$	$retVolL_{t,T}$
WW	0.0911	0.0478	0.0478
M	0.0456	0.0234	0.0234
Q	0.0158	0.0081	0.0081
S	0.0081	0.0042	0.0042
Y	0.0042	0.0022	0.0022

Table 6.1. *Average uniqueness ($avgU$) for each label and timespan T.*

Note that the longer the timespan, the smaller is the $avgU$. This result is expected, once more outcomes share the same price information. This behavior is also observed between $retL_{t,T}$ and $volTrendL_{t,T}$ along all terms, once the latter outcome covers a broader timespan than the former.

$avgU$ can be interpreted as the mean uniqueness of all outcomes y_i, $i = 1,\ldots,I$. E.g., $retL_{t,WW}$ has, in average, 9.11% of non-concurrent information. In other words, its outcomes along t share 90.89% of the information, in average. $avgU = 1$ indicates no overlapping information, while $avgU = 0$ indicates perfect overlapping.

6.1.1 Random forests estimator with overlapping outcomes

As we saw in the last subsection, all the outcomes have a high rate of overlapping information. It means that very similar information is going to feed the Random Forest estimator, producing correlated individual begged trees and, by the end, a weak classifier.

Two precautions may be adopted to reduce this problem: (i) use a smaller number of bootstrapped elements in each bagged tree, and (ii) use sample weights for the node split computation. We will detail both approaches next.

6.1.2 Reduced bootstrap sets

As we saw in Section 2.2, the Random Forest is built as an average of single trees. Each single tree is supplied with a bootstrapped (BT) set. Usually, in machine learning methods, each BT set is constructed through random selection of N elements with replacement, where N is the total number of elements available in the train data.

Nevertheless, once the labels have a high degree of common information, the BT set will also inherit this feature. One way to reduce this effect is to select only a fraction of the total number of elements at each BT set. This fraction could be the average uniqueness or a multiple of it.

Figure 6.2 shows an example for the $retL_{t,Y}$ label, which has a mean average uniqueness of 0.42%. We generated three random BT sets from it using $avgU$ multiples respectively of 10, 20 and 50. That means, the length of each set is $multiple$ x 0.42% times the original number of observations. As a result, we can compute the average uniqueness of each generated set and compares it to the original one.

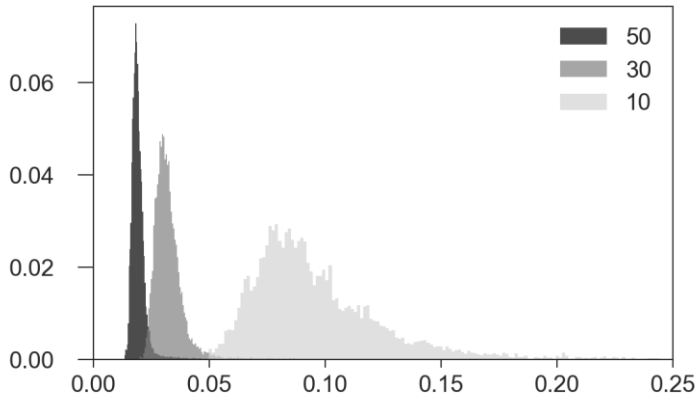


Figure 6.2. *Average uniqueness histograms of bootstrapped sets using multiples of 10, 20 and 50.* The greater the multiple, the smaller the average uniqueness, and the greater the similarity to the original set.

The greater the multiple, the greater the similarity to the original set. For $multiple = 10$, we have $mean(avgU_{10}) = 9.9\%$, or 23.9x the original one. For $multiple = 30$, we have $mean(avgU_{30}) = 3.3\%$, or 8.0x the original one. And lastly, for $multiple = 50$, we have $mean(avgU_{50}) = 2.0\%$, or 4.8x the original one.

6.1.3 Sample weights

We may make use of sample weights to emphasize low overlapping outcomes importance. As we saw earlier, the node impurity is computed based its class probabilities (cross-entropy impurity method). The sample weights are applied in the impurity computation. Using the

average uniqueness of each observation as weight, we enforce more importance on "more unique" samples.

Additionally, we may compose the sample weight with value attributions of the outcome. I.e., high values of return and volatility trend ($futReturn_{t,T}$ and $volTrend_{t,T}$, variables that determine the labels) will have more importance in the nodes splits as well. Therefore, we define the sample weights as (Prado, 2018):

$$\widetilde{w}_i = |attrib_i| \times u_i$$

$$w_i = \widetilde{w}_i I \left(\sum_{j=1}^{I} \widetilde{w}_i \right)^{-1}$$

where $attrib_i$ is $futReturn_{t,T}$, $volTrend_{t,T}$, or the product of both respectively for the labels $retL_{t,T}$, $volTrendL_{t,T}$ and $retVolL_{t,T}$. Figure 6.3 shows three histograms: average uniqueness, return attribution and the weight, computed as the normalized $|attrib_i| \times u_i$ for the two-week Two-state return outcome.

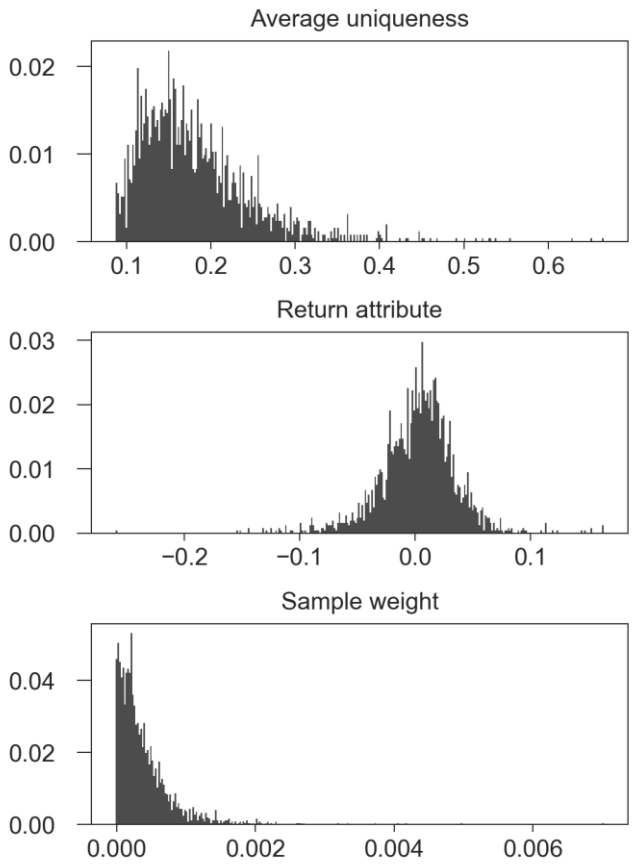


Figure 6.3. *Frequency of average uniqueness (top), return attribute (center) and the composed sample weight (bottom).*

7 Class weights

Imbalanced class distributions may cause biased and poor-quality estimators in classification problems. Let's exam $retVolL_{t,Y}$ as an example: $State\ 3$ has approximately half of the observations, while $State\ 1$ and $State\ 2$ share the other half (see Table 4.3).

Although $State\ 1$ and $State\ 2$ may not be considered rare events, if we do not treat this imbalance, we may have an estimator that favors the majority class in detriment of the overall accuracy. As a solution we apply class weights to the fitting procedure.

For each bootstrapped training set, we evaluate the class composition of the outcome and build weights that are inversely proportional to the relative presence of each class. As the sample

weights presented in Section 6.1.3, the class weights are used to adjust the weights of each node split. Actually, both weights (sample and class) are multiplied to compose the final load to the node splitting computation.

8 Purged cross-validation

As we saw in Section 2.2, the Random Forest is built as an average of multiple decorrelated single trees, each one of them supplied with bootstrapped data set. Usually each BT set is constructed having the same length as the training set. In this configuration, on average, each bagged tree uses approximately two-thirds of the training observations (Gareth et al. 2013). Thus, the remaining one-third of the data are not used to fit the tree. We can call this the *out-of-bag* observations (OOB), and it might be used to evaluate the model generalization power, once the estimator "has not seen it".

Nevertheless, this configuration of out-of-sample testing using OOB samples does not apply to our case. As we have neighbor outcomes containing a high level of overlapping information, this method would cause *information leakage* between the training and testing sets, causing non-detectable model overfitting. Figure 8.1 illustrates this problem and the cause of information leakage.

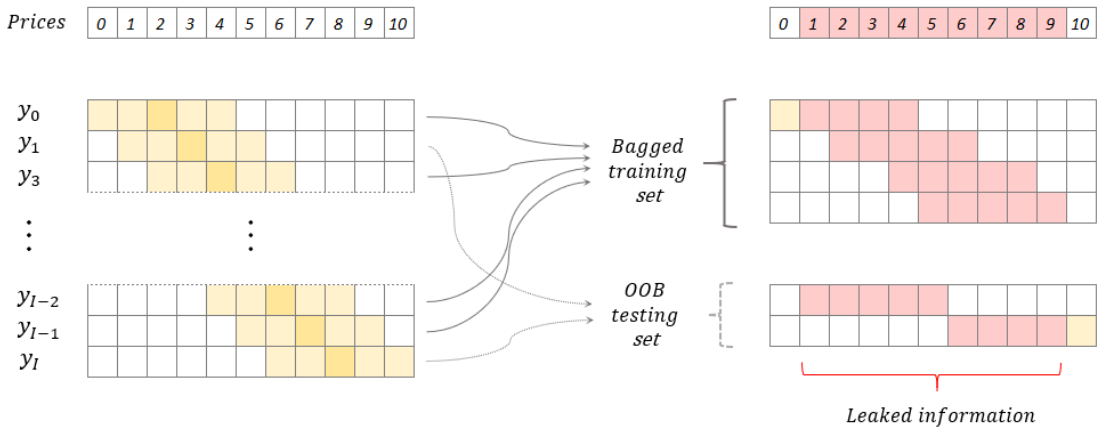


Figure 8.1. *Label y_t produced from price vector (left) is randomly bootstrapped to constitute the training set (upper right). Out-of-bag (OOB) sample that forms the testing set shares information with in-sample data (lower right), generating leakage and distorted generalization scores.*

A second possible approach to measure the model generalization power and largely used by standard machine learning works is the *cross-validation*. In this arrangement, the observations are partitioned into k subsets and, for $i = 1, \ldots, k$, the estimator is trained on all subsets excluding i, and tested on i. This method produces k out of sample performance metrics, that are averaged.

By the same reason mentioned before, k-fold cross-validation also does not avoid information leakage, inducing to wrong model selection conclusions. In our case, this will always occur in the neighbor observations located close to the train/test cross-validation partitions limits, once they are going to share common price information. One possible solution is to use purged cross-validation (Prado, 2018). Figure 8.2 illustrates this method.

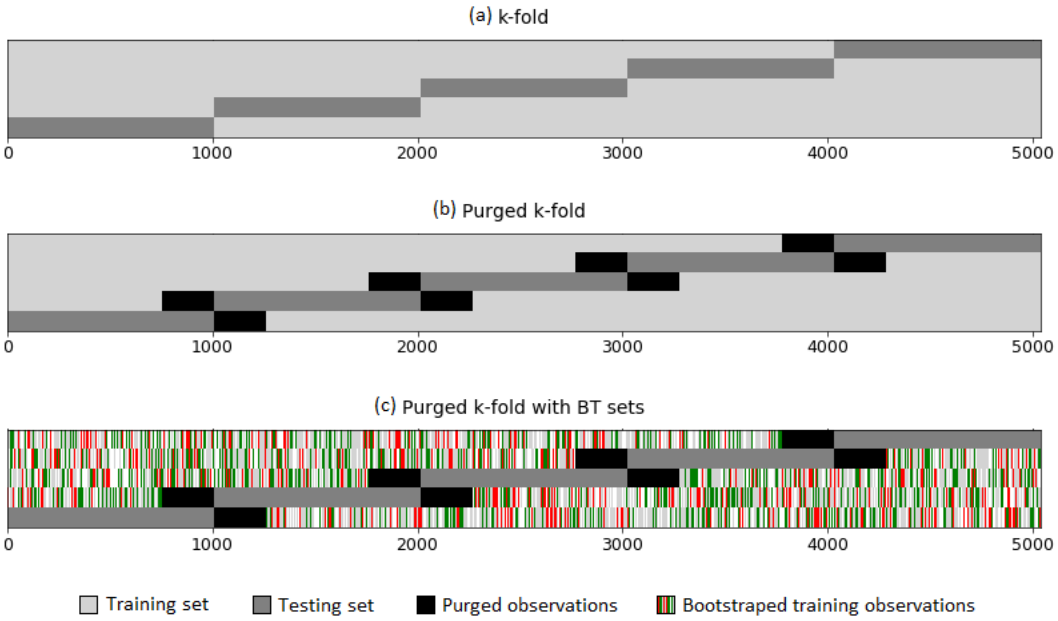


Figure 8.2. *(a) Training and testing sets for the cross-validation structure for* $k = 5$. *(b) overlapping observations are excluded from the training sets, eliminating the information leakage. (c) bootstrapped observations are selected from the training set.*

Figure 8.2 (a) shows the training and testing sets for the cross-validation structure for $k = 5$. On (b), overlapping observations are excluded from the training sets, eliminating the information leakage to testing sets. Lastly, on (c), bootstrapped observations are selected from the training set.

9 Parameter tuning and model evaluation

Using the previous configurations, we perform a grid search for the maximum tree depth parameter in the interval between 1 and 100 for all the 15 regime characterization variables, using the features defined in Section 5.

In the following subsections, we detail the quality metrics used to make this evaluation and present the results.

9.1 Models evaluation

In this section we describe the classification performances for the parameter tuning procedure for all available predictors sets for each one of the outputs.

Figure 9.1 shows the *accuracy* for the 15 distinct models – $retL_{t,T}$, $volTrendL_{t,T}$ and $retVolL_{t,T}$, and $T = \{WW,\ M,\ Q,\ S,\ Y\}$. Each model has the same group of 64 predictor variables ($Return_{t,T}$, $Vol_{t,T}$, $priceTrend_{t,T_1,T_2}$, $volTrend_{t,T_1,T_2}$, respectively from Sections 5.1 to 5.4. Additionally, the horizontal axis contains the tree depth, from 1 to 100.

The top panel shows $retL_{t,T}$ accuracy results for each term along the 1 to 100 tree depth. We can note the accuracy tend to be better for longer terms, and the best result was found for the year term, with accuracy of about 58% for three depth of 19.

The center panel shows $volTrendL_{t,T}$ accuracy results. Differently from the $retL_{t,T}$ models, here the best accuracies are reached by the shorter terms. The *two weeks* term model presents the best result of 63% for three depth of 5. This model has the best accuracy among all.

Finally, the bottom panel shows $retVolL_{t,T}$ results. For this case, the model clearly cannot reach a good level of accuracy.

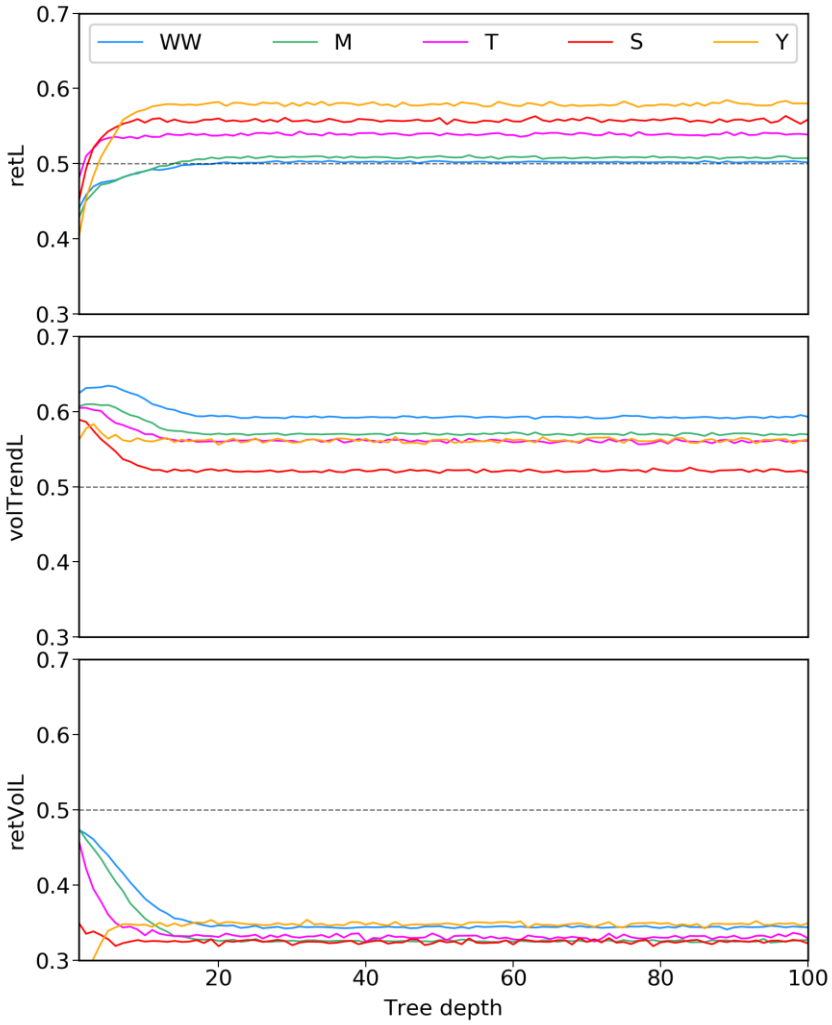


Figure 9.1. *Accuracy for the 15 estimated models. Horizontal axis contains the maximum tree depth from 1 to 100.*

9.1.1 Models evaluations summary

Bellow we presents a summary of the above discussed results.

$retL_{t,T}$ Reasonable capability to predict returns for longer terms as semester and year. Best accuracy of 58% at a maximum tree depth of 19 for the year period;

$volTrendL_{t,T}$ Good capability to predict volatility trends for shorter terms, Best accuracy of 63% at a maximum tree depth of 5 for the two weeks period;

$retVol_{t,T}$ Poor prediction quality for all terms.

10 Hidden Markov model

HHM is a stochastic process constituted of two parts: one undelaying Markov chain that determines the unobservable (hidden) state, and one observable state-dependent process (Zucchini et. al., 2006).

If the Markov chain $\{S_t\}$ has m states, then the bivariate stochastic process $\{(S_t, X_t)\}$ is called an m-state HMM. With $X^{(t)}$ and $S^{(t)}$ representing the values from time 1 to time t, the simplest HMM model can be summarized by (Nystrup, 2014):

$$\mathrm{P}(S_t|\, S^{(t-1)}) \;=\; \mathrm{P}(S_t|\, S_{t-1}), \quad t \;=\; 2,3,\ldots$$

$$\mathrm{P}(X_t|\, X^{(t-1)}, S^{(t)}) \;=\; \mathrm{P}(X_t|\, S_t), \quad t \;\in\; \mathbb{N}$$

Let's consider a simple two-state model with Gaussian conditional distributions:

$$\mathrm{X}_t \;=\; \mu_{S_t} + \varepsilon_{S_t}, \quad \mathrm{X}_t \;=\; \mu_{S_t} + \varepsilon_{S_t} \;\sim\; N(0, \sigma_{S_t}^2)$$

where

$$\mu_{S_t} = \begin{cases} \mu_1, & if \;\; S_t = 1 \\ \mu_2, & if \;\; S_t = 2 \end{cases} \qquad \sigma_{S_t}^2 = \begin{cases} \sigma_1^2, & if \;\; S_t = 1 \\ \sigma_2^2, & if \;\; S_t = 2 \end{cases} \qquad \Gamma = \begin{bmatrix} 1-\gamma_{12} & \gamma_{12} \\ \gamma_{21} & 1-\gamma_{21} \end{bmatrix}$$

and $\Gamma(t) \;=\; \{\gamma_{ij}(t)\}$ is the transition probability matrix with $\gamma_{ij}(t) \;= P(S_{t+1} = j \mid S_t = i)$. We illustrate this process in Figure 10.1, adapted from Zucchini et. al., 2006. More details on the estimation procedures for this kind of model can be found at the same publication.

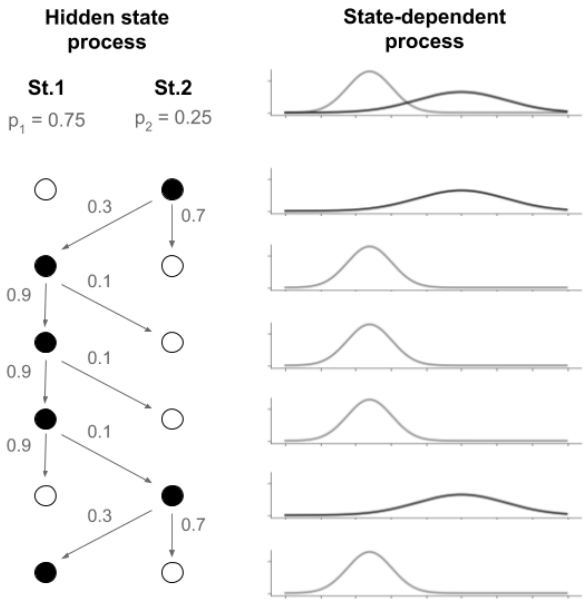


Figure 10.1. *Illustration of a two-state HMM model with Gaussian conditional distributions. The unobservable state (left), and the observable state-dependent process (right).*

In the present work, we estimate HMM models using R *depmixS4* package (Visser et. al, 2010). Figures 10.2, 10.3 and 10.4 illustrate HMM models fitted to S&P 500 daily returns for the years of 2010, 2011 and 2012, using respectively 2, 3 and 4 states and Gaussian distributions.

For the two-state model, we can note that it is able to capture high and low volatility moments, associated respectively with negative (-0.001) and positive (0.001) mean returns. The high volatility state ($State\ 2$), has a standard deviation of 0.018, which is 2.25 higher than the standard deviation of the low volatility state ($State\ 1$). Each state has a high stationarity, with low regimes transition probabilities – 0,6% from $State\ 1$ to $State\ 2$, and 1,9% from $State\ 2$ to $State\ 1$ –, indicating that states shifts tend to be very infrequent. The tables bring the estimated parameters and the transition matrix (Figure 10.2).

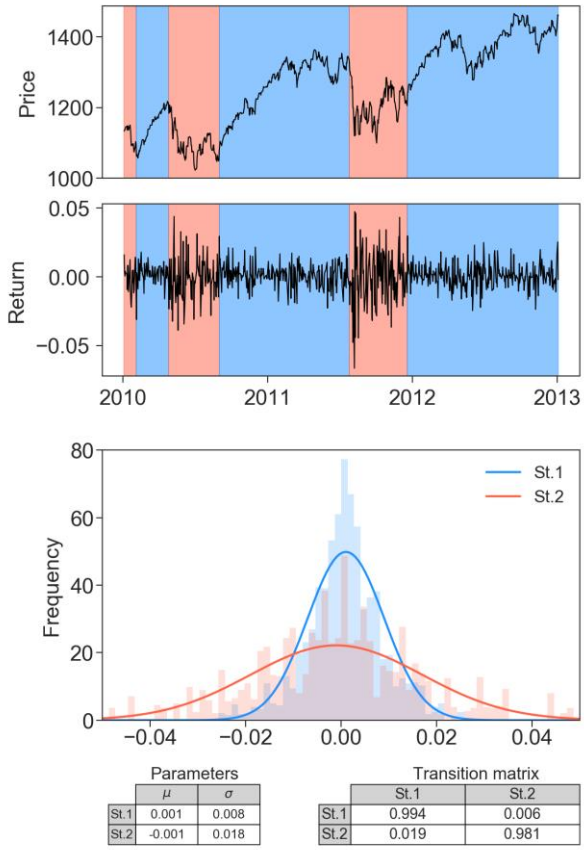


Figure 10.2. *Two-state HMM model fitted in the S&P 500 daily returns for the years of 2010, 2011 and 2012. The blue shaded areas on the top plot represent* $State\ 1$*, while red areas represent* $State\ 2$*. The histograms on the bottom were built with bootstrapped observations for each respective state. The blue and red solid lines represent the respectively estimated gaussian distributions.*

Figure 10.3 shows the results for the three-state model. In this case, we also note that the model can detect distinct volatility moments, with $State\ 1$ being the low volatility state and $State\ 3$ being the high volatility state. $State\ 2$, in this case, works as a transition state from $State\ 1$ to $State\ 3$, although, by the other hand, $State\ 3$ has a direct path to $State\ 1$ (indeed, the only existing path). The discrepancy between low and high volatility states standard deviation is broader than the two-state model, of the order of 5 times, being clear the transition state absorbed great part of the volatility of $State\ 1$.

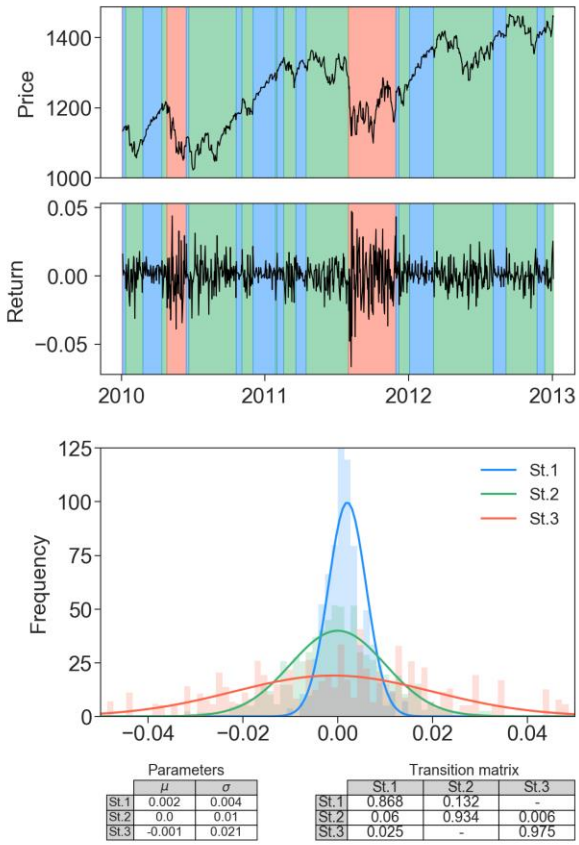
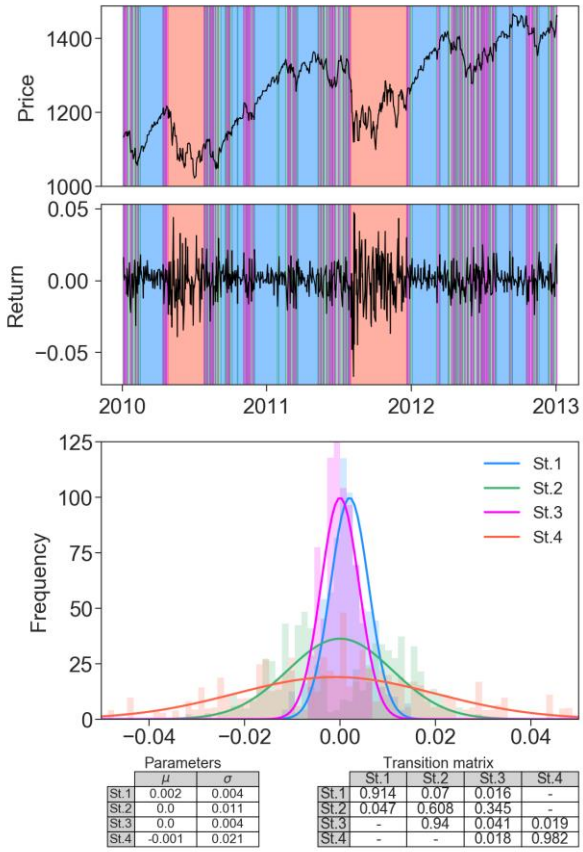


Figure 10.3. *Three-state HMM model fitted in the S&P 500 daily returns for the years of 2010, 2011 and 2012. The blue shaded areas on the top plot represent* $State\ 1$*, green areas represent* $State\ 2$*, and red areas represent* $State\ 3$*. The histograms on the bottom were built with bootstrapped observations for each respective state. The blue, green and red solid lines represent the respectively estimated gaussian distributions.*

Figure 10.4 shows the fitted results for the four-state model. The low volatility state ($State\ 1$) and high volatility state ($State\ 4$) have the same estimated parameters as the three-state case. $State\ 2$ and $State\ 3$ may be regarded as transition states between $State\ 1$ and $State\ 2$.

Nevertheless, it is interesting to note that $State\ 2$ and $State\ 3$ oscillate a lot between each other (see the first two panels). This behavior is based on the transition probabilities between both, with 34.5% chances of transitioning from $State\ 2$ to $State\ 3$ and 94% from $State\ 3$ to $State\ 2$.

As the former case as well, the only path to reach the high volatility state from the low volatility one is through $State\ 3$. But the way back, differently from the three-state case, where it had a direct path, in the present case it has to be also through $State\ 3$.



	μ	σ
St.1	0.002	0.004
St.2	0.0	0.011
St.3	0.0	0.004
St.4	-0.001	0.021

	St.1	St.2	St.3	St.4
St.1	0.914	0.07	0.016	-
St.2	0.047	0.608	0.345	-
St.3	-	0.94	0.041	0.019
St.4	-	-	0.018	0.982

Figure 10.4. *Four-state HMM model fitted in the S&P 500 daily returns for the years of 2010, 2011 and 2012. The blue shaded areas on the top plot represent* $State\ 1$*, green areas represent* $State\ 2$*, purple areas represent* $State\ 3$*, and red areas represent* $State\ 4$*. The histograms on the bottom were built with bootstrapped observations for each respective state. The blue, green, purple and red solid lines represent the respectively estimated gaussian distributions.*

Finally, we can note that the high vs. low volatility states coincides with the negative vs. positive return states respectively. Additionally, and very important, we note that these two regimes also

coincide along the two, three and four states HMM models. I.e., the periods of both regimes overlap independently of the number of states. This can be better seen in Figure 10.5.

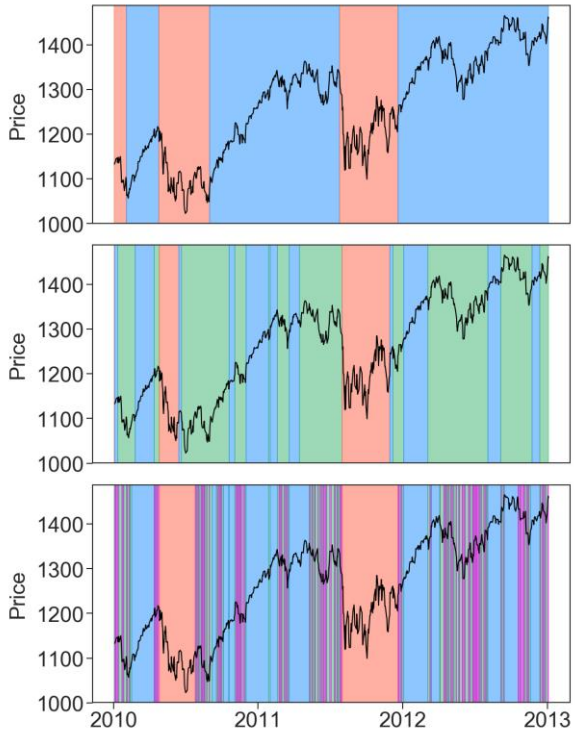


Figure 10.5. *Two, three and four-state HMM model fitted in the S&P 500 daily returns for the years of 2010, 2011 and 2012. High and low volatility states periods overlapping.*

11 Out-of-sample model comparison

In this section we are going to execute dynamic investment strategy simulations based on the three presented models using an OOS period, from Aug 2013 to Aug 2018. The investment strategy may assume two distinct positions, depending on the market regimes predicted by the respective model. If the market is identified as being in a low risk period, then the resources are fully located to the risk asset (S&P 500). By the other hand, if the market is identified as being in

a high-risk period, then the resources are fully located to the risk-free asset, in this case represented by a 3% annual interest rate investment.

The simulations of both strategies are executed as follows:

1. HMM strategy: for each day t in the simulation period, the 2 state gaussian HMM model is fitted to the last 2 years, including t. The estimated state for t (low volatility state or high volatility state) determines the new investment position;

2. Random Forest $retL_{t,\mathrm{Y}}$: for each day t in the simulation period, the 2 state $retL_{t,\mathrm{Y}}$ model is fitted to the last 10 years. The estimated state for t (positive or negative return) determines the new investment position; and

3. Random Forest $volTrendL_{t,\mathrm{WW}}$: for each day t in the simulation period, the 2 state $volTrendL_{t,\mathrm{WW}}$ model is fitted to the last 10 years. The estimated state for t (positive or negative volatility trend) determines the new investment position.

Note that the simulations are executed in the presence of transaction costs of 0.1% of the portfolio value. Figure 11.1 presents the portfolios values trajectory of the investment strategies.

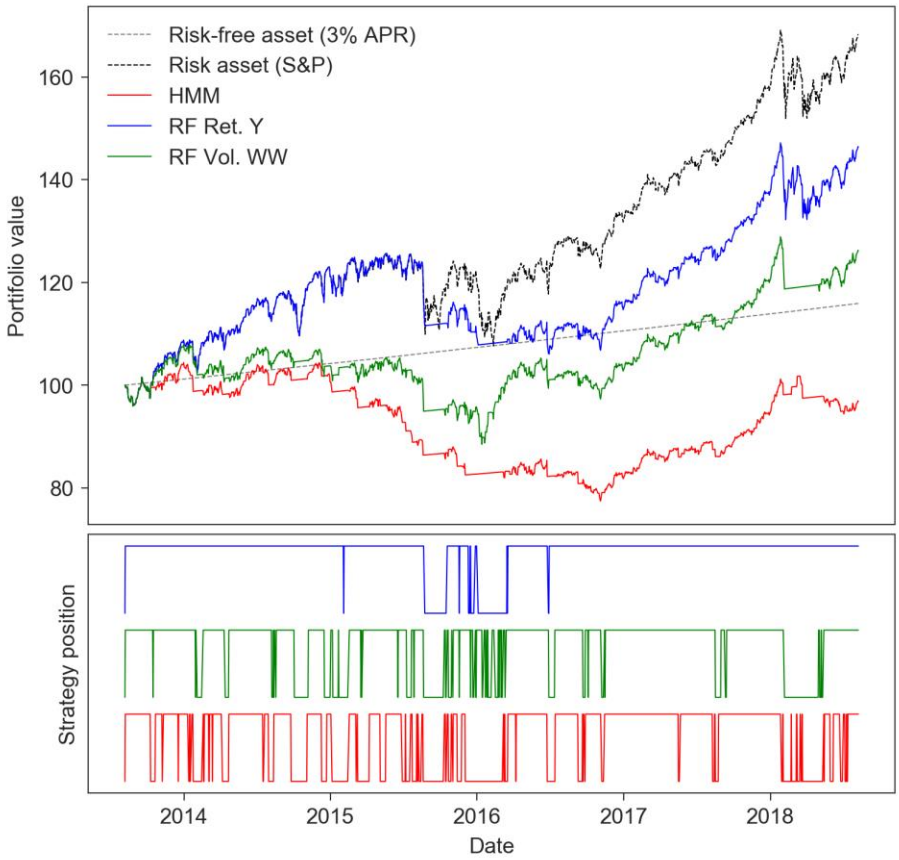


Figure 11.1. *Investment strategies simulation with out-of-sample data. Portfolio values for HMM, RF Ret. Y and RF Vol. WW are shown with S&P and risk-free asset references (top), and investment positions of each strategy (bottom).*

The *Sharpe Ratios* (SR) of each strategy, defined as the ratio between the average excess return (i.e., excess in relation to the risk-free rate) and the returns standard deviation are presented in Table 11.1. We can note that the *RF Ret. Y* strategy has the best performance among the three implemented, with a SR of 2.13. The HMM strategy presented a negative performance.

Portfolio	SR
Risk (S&P)	3.01
RF Ret. Y	2.13
RF Vol. WW	0.88
HMM	-2.28

Table 11.1. *Investment strategies Sharpe Ratios.*

It is also interesting to note that none of the implemented portfolios had a better result than the risky asset buy-and -hold strategy. Despite this is an indication that more investigation is needed to understand the predictor variables determinants to the strategy movements, we have to have in mind that this kind of simulation is biased, once it represents only one possible realization path of the underlying stochastic process.

12 Conclusion

Economic regime identification has been object of numerous investigations in economics and financial modeling using conventional statistical and econometric technics.

In the present work we developed a supervised machine learning classifier using *Random Forest* technic to identify economic regimes using the S&P 500 stock market index series. The regimes were derived in three ways: (a) two-state positive vs. negative returns, (b) two-state positive vs. negative volatility trend and (c) three-state regime produced by the combination of (a) and (b).

We used sampling methods, among other techniques, to de-correlate bagging data sets and avoid the use of highly serial correlated information. Models related to regimes characterized by future price return and volatility trend had the best accuracy results.

We contrasted this approach with the broadly studied Hidden Markov models by executing dynamic investment strategies, showing that the tree-based models had better performance than the HMM.

References

Ang, A., Timmermann, A., 2011. Regime changes and financial markets. Working Paper 17182, National Bureau of Economic Research.

Bulla, J., 2011. Hidden Markov models with t components. Increased persistence and other aspects. Quantitative Finance 11 (3), 459–475.

Duprey, T., Klaus, B., 2017. How to predict financial stress? An assessment of Markov switching models. Working Paper Series 2057, European Central Bank.

Friedman, J., Hastie, T., and Tibshirani, R., 2001. The elements of statistical learning, volume 1. Springer Series in Statistics, New York.

Gareth, J., Witten, D., Hastie, T., Tibshirani, R., 2013. An Introduction to Statistical Learning: With Applications in R. New York: Springer.

Guidolin, M., Timmermann, A., 2007. Asset allocation under multivariate regime switching. J. Economic Dynamics & Control 31 (11), 3503–3544.

Hamilton, J. D., 1989. A new approach to the economic analysis of nonstationary time series and the business cycle. Econometrica, 57 (2), 357–384.

Mulvey, J. M., Bae, G. I., Kim, W. C., 2014. Dynamic asset allocation for varied financial markets under regime switching framework, European Journal of Operational Research, Elsevier, vol. 234 (2), pages 450-458.

Nystrup, P., 2014. Regime-based asset allocation. Do profitable strategies exist? M.Sc. thesis, DTU Compute, Department of Applied Mathematics and Computer Science.

Nystrup, P., B.W. Hansen, H. Madsen, and E. Lindstrom, 2015. "Regime-Based Versus Static Asset Allocation: Letting the Data Speak." The Journal of Portfolio Management, Vol. 42, No. 1, pp. 103-109.

Prado, M. L., 2018. Advances in Financial Machine Learning. John Wiley & Sons, Inc., Hoboken, New Jersey. Print.

Renshaw, E., 2002. The Stock Market, Oil Price Shocks, Economic Recessions and the Business Cycle With An Emphasis on Forecasting, available at https://www.albany.edu/cer/bc/bc_essays.html.

Sheikh, A. Z., Sun, J., 2012. Regime change: Implications of macro shifts on asset class and portfolio performance. J. Investing 21 (3), 36–54.

Varian, Hal R., 2014. Big Data: New Tricks for Econometrics. Journal of Economic Perspectives, 28 (2): 3-28.

Visser, I., Speekenbrink, M., 2010. depmixS4: An R Package for Hidden Markov Models. Journal of statistical Software, Vol. 36, Issue 7.

Zucchini, W., Berzel, A., Bulla, J., 2006. Hidden Markov Models.