Forecasting the direction of high-frequency returns. An Ensemble-Trees Applications.

Rafael Silva Wagner
Departament of Economics and International Relations
Federal University of Santa Catarina

André Alves Portela dos Santos
Departament of Economics and International Relations
Federal University of Santa Catarina

July, 2018

Abstract

We propose the construction of an intra-daily algorithmic trading strategy based on machine learning models, specifically Tree-Based Methods as Gradient Boosting, Random Forests and BART. The models are constructed to capture the dynamic of market micro structure variables and model their impact into the sign of the future returns. We also verify the impact in the statistical and financial performance of time aggregation and minimum returns requirements for class identification. In the end we verify the financial significance of our models through a trading simulation.

Introduction

Forecast the financial markets is of huge interest for different agents. Central Bankers benefits themselves adjusting their forecasts and fine-tuning their monetary policies, additionally they are interested in forecasting international exchanges and interest rates that affects the inner market and are out of their direct control. Fund managers are interested in adjust the shares of their portfolios and provide excess of returns for their clients along with traders that are always searching for profitable trading strategies.

Besides the variety of different interests this is not an easy task, expectation of the asset returns doesn't show any reasonable predictability, but, besides that, some success has been achieved along the years including the forecasts of returns sign. This possibility was first documented by a long time in medium term horizons but only when Christoffersen et al. (2006) showed that predictability of sign, was a consequence of predictability of variance, or higher moments, that the forecast of returns mean gave space to the forecast of returns sign. The general theoretical results will be recovered in section 1.2.

By the same time that the development of models to forecast the sign in medium horizons was emerging, Market Microstructure Theory was trying to explain how trading frictions and inefficiencies could generate forecasts for the mean of high-frequency returns.

Cao, Hansch e Wang (2009) for example verified that at least 22% of the oscillation of prices in intra-day horizons can be explained by the trading dynamic, generating forecast ability of high frequency returns mean, we'll talk more about Market Microstructure on section 1.3.

The availability of a great amount of high-frequency data allowed that the success of the forecasts at lower frequencies spread out to the highest ones. This huge amount of data needs special techniques for being analyzed, this enforced the researchers to get some distance of the traditional econometric techniques and get more closer to data science models, looking for benefits that are made for generate accurate predictions through huge amount of data without the need of strong hypothesis that traditional statistical models usually require.

Some works made by Dixon, Klabjan e Bang (2016), Tsantekidis et al. (2017) and Fletcher e Shawe-Taylor (2013) show that this approach can be well-succeed, all works achieve good forecast performance when trying to forecast the sign of the prices or the spread of assets based on market microstructure variables as: number and volume of orders into bid-ask, number of new/canceled/executed orders, quantity/volume of closed trades, among others variables. We propose to verify the ability of a algorithms family that as far as we know hasn't yet been fully analyzed, to explain the dynamics of the sign we'll use twenty-seven market microstructure variables describing the trading dynamics. Ensemble-trees are a hot topic in the Machine Learning (M.L.) community specially due to recently excellent performance in M.L. competitions.

Breiman et al. (1984) were the responsible to popularize the decision trees, they're characterized by not presuming any data-structure and can be preferable in situations where doesn't exist good prior knowledge for modeling the Data Generating Process (DGP), decision trees will be presented in section 2.2.2. A lot of ensemble tree methods has already been proposed in literature we'll investigate the forecast ability of three of them: Boosting (FREUND; SCHAPIRE, 1997), Random Forests (BREIMAN, 2001) and BART (CHIPMAN et al., 2010). Together we'll make forecasts with the Logistic Regression adding a linear model to compare the other proposals, we'll review all the methodologies in section 2.

We'll use this models to forecast the signal of the asset returns based on market microstructure variables. We'll analyze the major Brazilian asset, the Petrobras Company (PETR4.SA), recently in media due to politics corruption scandal. The series starts in 01/12/2015 and goes to 12/06/2018, part of the troubled period.

1 Theoretical Reference

1.1 Sign of Returns

Let's start defining the terms used by literature, we'll follow the terminology of Christoffersen et al. (2006). Let Ω_t be the set of information available at t, and $Pr[R_t > 0]$ be the probability of an asset be positive in period t, with the respective conditional mean and variance $\mu_{t+1|t} = E[R_{t+1}|\Omega_t]$ and $\sigma^2_{t+1|t} = Var[R_{t+1}|\Omega_t]$.

We'll say that the series of returns presents predictability of conditional mean if $\mu_{t+1|t}$ varies along with Ω_t, the predictability of conditional variance is defined in the same manner. If $Pr[R_t \lesseqgtr 0]$ exhibits conditional dependence, that is, if $Pr[R_{t+1} \lesseqgtr 0|\Omega_t]$ varies along with Ω_t we'll say that the series of returns presents the sign predictable (or that the series has the sign of the change in price foreseeable)

1.1.1 Intraday Returns

Due to recent results of literature to be presented in section 1.3.1, we'll investigate if exists predictability of returns sign in intra-day horizons. In intra-day sampling windows we have a traditional problem, choose the forecasting window t. In contrast to the first impression, that granular data generated by shorter windows, would improve the quality of our dataset, a huge amount of work in the field of realized volatility have extensively demonstrated that the microstructure noise mess up the signals that would be clarified by this 'ultra-high frequency', frequently in this literature we find that the optimal short horizons, aren't so short; for a recent comparison of realized volatility estimators look at the work of Liu, Patton e Sheppard (2015).

In high frequency intervals situations where we don't see any change on prices are quite common. This kind of situation defines a 3 class prediction/classification problem, returns can be positive/negative/null. Null return can be intuitively understood as $R_t = 0$, but we'll define it in a different way. Due to the noise on high frequency signs we'll test different minimums of oscillation in price to classify it as different from null, this was originally proposed by Tsantekidis et al. (2017), this made through the change of definition of null return to $|R_{t+1}| < m$ where m is a real constant.

We believe that this change will help us by two manners, the first was already exposed, mitigate the effect of the noisy signal of high frequency returns on their signs. The second reason for that is to focus on one of the main applications of this kind of forecast, generate high-frequency trading strategies. To transform forecasts that has good statistical properties into profitable signs the great hindrance are the transaction costs. Transaction costs are a list of taxes, brokerage and operational costs that summed up to structural costs as the bid-ask spread, introduces expenses in the negotiation process (HASBROUCK, 2005). To achieve this economic significance we'll set a threshold to classify a return as different from null, an empirical comparison of different possible values for this threshold is presented in section 3.1.

Presented our problem should be clear by now that we have on our hands a problem of classification in three categories, this kind of problem is usually described by multinomial density functions. Following Lehmann e Casella (2006), the multinomial distribution is characterized by a sequence of n *i.i.d.* experiments that can leave us to r different classes with the respective probabilities $p_1, p_2, ..., p_r | \sum_{i=1}^{r} p_i = 1$. If X_i represents the number os experiments that resulted in class i the conjunct distribution of all $X's$ is the multinomial

distribution $M(p_1, ..., p_r; n)$:

$$P(X_1 = n_1, X_2 = n_2, \ldots, X_r = n_r) = \frac{n!}{n_1! n_2! \ldots n_r!} p_1^{n_1} p_2^{n_2} \ldots p_r^{n_r}, \qquad \sum_{i=1}^{r} n_i = n. \tag{1}$$

We can also derive this equation in a intuitive way following Ross (2013), if we notice that the sequence of n results that leaves us to the result i, n_i times for all $i = 1, 2, \ldots, r$ has, for the independence hypothesis, probability of occurrence of $p_1^{n_1} p_2^{n_2} \ldots p_r^{n_r}$, as we have $n!/(n_1! n_2! \ldots n_r!)$ sequences of results, will exist $n!/(n_1! n_2! \ldots n_r!)$ different permutations of n which $n_1, n_2, ...n_i$ of this permutations will be of the same kind. For the rest of this paper we'll assume that our dataset follows this density and in our specific problem $r = 3$.

1.2 Related Work

Christoffersen e Diebold (2006) were the firsts to give a rigorous treatment about the relationship between sign of returns and volatility, in spite of other works had already show the possibility of forecasting the sign. The greatest innovation introduced was the elucidation that dependence of sign doesn't require dependence of conditional mean, that is, the stylized fact that the mean of returns is hard to predict doesn't contagious the predictability of their signs.

The authors demonstrated that once the mean of returns is different from zero $\mu \neq 0$, we can achieve predictability of their sign modeling the variance. Suppose that $R_{t+1}|\Omega_t \sim N(\mu, \sigma^2_{t+1|t})$, $\mu > 0$; we can verify that the probability of a given return be positive is:

$$\begin{aligned} Pr_t(R_{t+1} > 0) &= 1 - Pr_t(R_{t+1} < 0) \\ &= 1 - Pr\left(\frac{R_{t+1} - \mu}{\sigma_{t+1|t}} < \frac{-\mu}{\sigma_{t+1|t}}\right) \\ &= F\left(\frac{\mu}{\sigma_{t+1|t}}\right), \end{aligned} \tag{2}$$

where F is the cumulative density function. In other words, we know that the definition of the probability of an specific return be positive is the difference between one and the probability of him be negative. If we take the difference of the return in the next period by his mean and divide-it by the conditional standard deviation we get to the standardization of the future return. This can also be expressed through the cumulative density function which gives us the probability of an specific value be greater than zero.

We can see that as long as the conditional variance changes, the probability of a return be positive change together, independently of the mean staying constant. The lower the variance, the greater will be the probability of a given return be positive, being otherwise. In addition, when the mean approximates to zero the probability of a return be positive reaches fifty percent, due to the symmetry of the normal distribution. To sum up, in signal prediction the dynamics of the variance interacts with a no zero mean producing a time-varying probability of an asset be different from zero, generating our object, signal predictability.

Later Christoffersen et al. (2006) demonstrated that this result extends to cases where the unconditional mean is zero, since the higher moments of the distribution, specially the skewness and kurtosis be different from zero. To find this result they realize a Gram-Charlier expansion like:

$$\begin{aligned} 1 - F\left(\frac{\mu}{\sigma_{t+1|t}}\right) \approx & 1 - \Phi\left(\frac{\mu}{\sigma_{t+1|t}}\right) \\ & \times \left[-1 + \frac{\gamma_{3,t+1|t}}{3!}\left(\frac{\mu_{t+1|t}^2}{\sigma_{t+1|t}^2} - 1\right) + \frac{\gamma_{4,t+1|t}}{4!}\left(\frac{-\mu_{t+1|t}^3}{\sigma_{t+1|t}^3} + \frac{3\mu_{t+1|t}}{\sigma_{t+1|t}}\right)\right], \end{aligned} \tag{3}$$

where Φ is the normal-standard distribution and γ_3, γ_4 are the conditional asymmetry and kurtosis coefficients of returns, for more details about this expansion see the work of Cramér (2016). Jondeau e Rockinger (2001) argument that this result allows way more flexibility in comparison to the usual normal distribution, the point that we want to call your attention is that even with a zero mean distribution if the asymmetry or kurtosis coefficients be different from zero we can achieve forecast of the sign through the forecast of high-order moments.

Christoffersen e Diebold (2006) argue that this predictability would only be presented in medium-horizon forecasts, despite of the strong memory of the volatility in the short-run the expected returns would be too low, turning the ratio μ/σ too close of zero and the respective probability of the sign be different from zero to similar from 50%. In fact this generates a common phenomenon in the intraday horizon, the price doesn't vary in all periods of time. But as pointed by the authors, in the short-run other variables are relevant for the dynamic of returns, and consequently their signs. In the intraday horizons predictability of mean can be generated extracting patterns and signs from the micro structure dynamic of each specific asset.

1.3 Market Microstructure

As already pointed we'll search for forecast capacity in intraday horizons, we'll get together the variables that were used by Christoffersen et al. (2006), the moments of the returns, with most recent works that has verified the predictive capacity of high-frequency signs using market microstructure variables. Market Microstructue is a research field in finance that relies on the trading mechanisms of markets, the object of research typically is the trading system, verifying how the behavior of the trading system affects the prices, quotations, volumes and others characteristics of the trading dynamics.

As usual, starting the section let's present the microstructure jargons. Traders go to the market trade their assets and when they do that they inform their supply or demand. Despite of the great number of different execution order types most of traders opt for limited orders or market orders. The difference between the two is trivial, in limited orders you are not willing to negotiate at the current price of the asset, so you make an offer to the market specifying the amount of shares that you want to trade in an specific price, every time a new limit order is placed it goes directly to the limit order book which is a list of all active orders, orders that still haven't been canceled or negotiated, the price of the best buying offer is called bid and the price of the best selling offer is called ask, the difference between both is called the bid-ask spread. A market order is the opposite, buyers/sellers are satisfied and willing to trade at the current price in the correspondent

ask/bid, they will send a market order to the exchange and the match engine will make the deal with the order that has the best price and came earlier, the trader don't need to specify the price, he only tells the quantity, the price will be defined by the best limited order.

But how the bid/ask price can change? In the regular market routine only by three ways, if all the orders in a specific price been canceled the bid/ask price will change for the price defined in the next best offer, if all limited orders been executed by market orders the price of the bid/ask again will change for the next best offer. But if we have a whole in the book, a tick inside the spread that doesn't have any order placed, and someone puts a limited order in that price, the price of this order will form the new bid/ask price. So, the current supply and demand of the market only changes in this three ways, you can see that this two prices can change a number of times before we get a trade record.

Said that let's remind Walrasian microeconomics following O'Hara (2003). The walrasian auctioneer in all periods of time receives the current supply and demand, forming the Walrasian equilibrium with the respective price, this price is immediately communicated for all participants sharing the content of information of the trades. If we had the situation of one of the participants with a greater information set, also known as information asymmetry, by the time that he gets into the market and presents his supply/demand, all traders receives his content of information. This theory forms the basis of the most traditional finance models as the *Capital Asset Price Management* (CAPM) and *Arbitrage Price Theory* (APT).

But we could also imagine that could happen what's called a partial informative equilibrium, where frictions and inefficiencies in the trading mechanism prevent the information to be real time shared to all participants, if this equilibrium could exist, there would be information inside the limited order book about an asset that still isn't incorporated into the price, opening a trading opportunity in which we analyze the order flow and discover information that can help explain the future dynamics of price.

1.3.1 Forecasting Returns inside the Ultra-High Frequency

There are many approaches to organize this set of information, model-it and use-it to forecast the high-frequency returns. Breiman (2001) describes the two traditional paths that a statistical analyst usually goes through, the first creates a list of hypothesis of the data generating process (DGP) and use it to build and estimate a predictive model that arises from the hypothesis. The second uses generic algorithms of high versatility without making any hypothesis of the specific process. The high versatility of the second approach allows the easy adaptation to generic DGP's (data generating process) from the available dataset.

Machine learning is inside this second approach, their techniques are designed to capture complex sets of non-linear relationships in high dimensional datasets, without require any prior hypothesis about the true DGP (DIXON, 2017). Recently several works achieved good forecast performance using machine learning models to forecast high-frequency signs of returns, our interest is inside a share of this literature that combines these family of algorithms to the market microstructure variables.

Dixon (2017) proposed to model this relationship with the sign of returns using recurrent neural networks, a machine learning technique specific for ordered data like time-series. His dataset consists of the records of the *E-mini S&P500* future over the entire

month of august/2016. He compared his model with other machine learning structures such as long memory neural networks and more traditional statistical methods as the Kalman filter (KALMAN et al., 1960), he achieved the highest forecast accuracy between all analyzed models with the recurrent neural network.

Also in the neural networks framework Tsantekidis et al. (2017) used convolutionary neural networks, a very used technique in speech and sound recognition, to forecast the mean of the bid-ask spread of 5 finish stocks. As in our work, they defined a threshold to classify the sign of returns as different from null. As regressors variables they limited the analysis to the price and volume of the first 10 levels of the limited order book, bid and ask sides, totalizing forty variables in each period that lagged 10 times formed a regression model of 400 variables. They compare their proposal to multi-layer neural networks and support vector machines and achieved the highest forecast quality with the convolution approach. By end they verify the performance of their models in 3 different forecast horizons and verify that as long the horizon became the strategy become more consistent.

Also with Support Vector Machines, another machine learning method for classification, Fletcher, Hussain e Shawe-Taylor (2010) modeled the relation between the sign of returns and the limited order book of the EUR/USD parity. They used a more restricted set of information consisting only of the volume into the 6 first level prices of the limited order book. They verified that models that used multiple kernels achieved superior predictability than techniques that use only one kernel; in parallel to our proposal, they verify that the combination of multiple models has superior predictive ability than only one. They also verified that the quality of forecasts improve as the sampling window enlarge, the highest horizon they analyzed was 200 seconds. By end they made a financial simulation and verified that all strategies achieve positive results featured by the multiple kernels models.

Later in a sequential work Fletcher e Shawe-Taylor (2013) added 2 sets of variables to the already used in their later work, the first added set consists of information about the traded price including, last price, maximum, minimum, mean and standard deviation of the period; on the second set of variables were estimated parameters generated by the modeling of a wiener/poison process, also including the conditional duration. With this two information sets they build 165 kernels, that were later selected to verify the ones with highest information content, this also helps to decrease the computational cost of the full simulation. They verified that the most important kernels were the ones related to the historic of the traded prices, and that this combination of information generated by multiple kernels creates significant improvements on the forecasting quality specially in bigger sampling windows, again they limited the analysis to 200 seconds, in this horizon of forecast they achieve an accuracy of 55%.

Kercheval e Zhang (2015) were the first ones to extend this approach to NASDAQ stocks. They verified the forecast capacity of the direction of spreads mean, looking for the probability of the future ask being lower than the current bid and vice versa. They also created three separate set of variables, the first one containing only the simple variables as price and volume, the second with variables that were time insensible like spread, last deal, mean of the deals; and the third set with variables that were sensible to time like the acceleration and intensity of trades. They verified that the best set of variables was the first one, with the basic set of information. Again they verified the impact of the trading window and checked that the highest average mean coincide with the lowest standard deviation both in the highest frequencies, the higher frequency they verified was 6 minutes.

Han et al. (2015) extended the same set of variables that was used by Kercheval e

Zhang (2015) now using *S&P 500 - SPY* data, they evaluated the predictive ability of the sign of the spread mean. They verified that the orders closest to the bid and ask prices were the most relevant to make forecast. To model the relationship they proposed the use of random forests, a tree-ensemble methodology for classification, and verified that their performance was superior than those realized by Support Vector Machines, both with linear kernels and with radial basis functions

The lack of one predominant technique to model the relationship between market microstructure estimators and the returns signal turn this an open point in literature. In our work we'll increase one more test to this literature. Ensemble trees have recently returned to popularity due to the good performance of the *XGBoost* system (CHEN; GUESTRIN, 2016). In 2015 from twenty-nine winning solutions in Kaggle's, boosting trees were used is seventeen proposals, and eleven of the solutions were only composed by a well-tuned boosted tree. Due to the variety of problems that have been recently solved by this algorithm we'll test him in our problem and verify if he can achieve a good performance too, boosting trees will be presented in section 2.2.2.1. Despite of this awesome performance other ensemble tree methods are quite famous, as stated before we'll recover the work of (HAN et al., 2015) and also use the random forest model, in addition to that, we will also add one last model, inspired by the recently great performance of Bayesian Additive Trees for forecasting tasks we will also evaluate the model BART (KINDO; WANG; PEÑA, 2016).

2 Methodology

2.1 Data-Set

To construct our dataset we'll use the stock high frequency database from the Brazilian Stock Exchange B3.SA, we are in debt with Perlin e Ramos (2016) for the construction of a system that makes the manipulation of this dataset so simple. Our database starts in 01/12/2015 and goes to 12/06/2018. With this dataset we construct the variables described in table 1.

Table 1 – Variables

Time	Last Price
Period Returns	Sign of last Period
Volatility of Returns**	Number of trades*
Quantity of Trades*	Volume *
Weighted Price by Volume*	Number of new orders*
Number of Updated orders *	Number of Canceled orders*
Maximum Price of Offers *	Minimum Price of Offers*
Weighted Price of Offers*	

* This variables corresponds to two columns in the data frame, one for the buy and other for the sell orders.
** The sum of 10 equally spaced squared returns.

To realize the analysis of best frequency we construct 60 databases with frequencies ranging between 1 and 60 minutes, increasing the frequency of each by one minute. To add the time series structure we'll work with lagged variables and use the time until the trading sections open as covariate. Due to the structure of the Brazilian market a trading day is composite of 7 hours, the exchange opens 10:00 a.m. and closes at 17:00 p.m.. Due to night interruption we choose to work with only 2 lags, so that in the highest frequencies we still have at least 4 forecasting hours, using the three first hours of the day to construct the regressors.

Instead of we sample the data uniformly, we choose to sample then in one minute frequency independently of the variables time aggregation. Therefore in all databases we will forecast in a frequency of one minute but the aggregation of the lagged variables will change $t = 1, 2, 3, ..., 60$. For example in the thirty minutes data base we'll do our first forecast at 11:30, the variables of the current period time like number of trades will describe the number of trades between 11:30 and 11:00, the first set of lagged variables will aggregate the data from 11:00 to 10:30 and the second set of lagged variables will use information from 10:30-10:00. The second forecast will be made at 11:31 using the respective one minute ahead intervals for all other lagged regressors. We will do this procedure with one single intention, increase our database.

We'll restrict the forecast window space of all frequencies to the forecast space of the highest one, 60 minutes. Generating the same amount of forecasts for all the aggregation periods and avoiding create a benefit for the smallest frequencies, if we didn't do this procedure would have many more daily-hours in the small aggregations than in the largest ones.

As final consideration, after we find an optimal frequency we'll apply an approach proposed by Casals, Jerez e Sotoca (2009). After we choose a proper sampling window

to forecast we'll combine variables aggregated in other frequencies. This can lead to improvements in the forecast accuracy providing the notion of acceleration for all the variables, including the variables that aren't originally designed for that. As far as we know no other work have already applied this philosophy to the intra-day signal forecasting problem.

2.2 Machine Learning

We'll make forecasts for the returns sign using three Tree-Ensemble methods, Random Forests, Boosting Trees and BART, additionally we'll estimate the Logistic regression. The interest for the tree algorithms is due to the low volume of works evaluating their performance, with exception of Random Forests that has already been tested by Han et al. (2015). The interest in logistic regression is due to it being one of the most traditional linear regression models, usually when we see a non-linear classification method we ask if the improvement of forecast quality is due to capacity to generate non-linearity in the regressors space, Logit can give us some intuition for that.

2.2.1 Logistic Regression

Despite the existence of two traditional densities being used to define linear classification models, the logistic density (Logit) and the normal density (Probit) we'll focus the presentation in the first one, all main conclusions are extensible for the normal distribution and in fact both are widely used in applied works. We'll make our presentation focused in the multinomial density due to in the high frequency the price can go up, down or simply stay at the same price.

Following Czepiel (2002), let j be the number of classes that a random variable Z can assume, and let's say that this random variable respects the multinomial density presented in section 1.1.1. If M is the number of observations, we'll make subgroups of observations separated by the respective value of Z, in each of this sub-groups we'll count the number of times that Z assumes each of the j classes $n_1, n_2, , \ldots, n_j$ and store this results in a vector $\boldsymbol{n}$, so that $\sum_{j=1}^{N} n_j = M$.

Now let's define the matrix $\boldsymbol{y}$ with N lines and $J-1$ columns, where N is the number of scenarios, a scenario is an specific combination of the covariates. We will have to choose one of the j classes to be the base-line, the category from which we'll compare all others classes. Each element of $\boldsymbol{y}$ will store the number of times that the scenario i generated the response j in the cell y_{ij}, for all the i^{th} possible scenarios with all $J-1$ classes, excepting the basis class. Now let's talk about the $\boldsymbol{\pi}$ matrix with the same dimensions of $\boldsymbol{y}$, this new matrix will store in each of their π_{ij} elements the sample probability to came to the j^{th} value of Z in the i^{th} scenario.

By least let's define our scenarios, the matrix X has N lines with $K+1$ columns containing in the first column only ones, each line correspond to one combination of the regressors, in a manner that none of the lines is repeated. And define the β matrix that in a similar way has $K+1$ lines and $J-1$ columns, where each element in β_{kj} corresponding to the importance of the k^{th} covariate for the j^{th} response value. We can initially define the logits of each reference class as:

$$log\frac{\pi_{ij}}{\pi_{iJ}} = log\frac{\pi_{ij}}{1-\sum_{j=1}^{J-1}\pi_{ij}} = \sum_{k=0}^{k} x_{ik}\beta_{kj} \qquad i = 1, 2, \ldots, N; \quad j = 1, 2, \ldots, J-1 \tag{4}$$

where we define the log-chances of one class j in relation to the basis class J, in an specific i scenario. Solving for π_{ij}, and standardizing the probabilities to sum 1 and be positive we get to:

$$\pi_{ij} = \frac{exp(\sum_{k=0}^{K} x_{ik}\beta kj)}{1 + (exp\sum_{k=0}^{J-1} x_{ik}\beta kj)}, \quad j \leq J, \tag{5}$$

$$\pi_{iJ} = \frac{1}{1 + exp(\sum_{j=0}^{J-1} x_{ik}\beta kj)}. \tag{6}$$

The equation (5) describe the probability of each of the j classes in the i scenario and (6) describes the probability of the basis class. To find the weights β_i we make a gradient search seeking to maximize the likelihood obtained through the joint density (7):

$$f(y|\beta) = \prod_{i=1}^{N} \left(\frac{n_i!}{\prod_{j=1}^{J} y_{ij}!} \prod_{j=1}^{J} \pi_{ij}^{y_{ij}} \right). \tag{7}$$

Usually this search is done by a Newton-Raphson algorithm, which can be seen as an approximation method derived from the taylor expansion, more computational details about how to implement this algorithm can be founded into Czepiel (2002).

2.2.2 Regression Trees

The next three models are all created expanding the concept of regression trees so we'll present these first.

Following Mitchell (1997) we can divide a tree-model into 3 structures: the root, the nodes and the leafs. The root can be understood as our original dataset, we'll take this dataset and divide him in two different no intersecting subsets that together result in the root set. To make this *split* we'll define a decision rule in one of the regressors variables, for classification variables we can make rules like equal/different from b where b is one of the classes, and for continuous regressors we can make a decision rule defining a threshold c that creates two subsets, one with observations that have values bigger than c and other with values less than or equal to c. This procedure is called a *split*, with the resulting subgroups we make again new splits creating new partitions on the regressors space, in the end of the splitting procedure we'll have no intersecting groups of observations that are called leafs, in each of this leafs we have a prediction rule. So when a new observation comes to us we will follow the sequence of tests until we got to a leaf, when we get there we simply follow the defined rule.

An illustrative example of this proceeding can be found in Figure 1, where in the left corner we display this decision scheme. Departing from the root, the point in the middle-top, we initially verify the first rule $x_1 \leq t_1$, if the variable from our observation is bigger then our threshold t_1 the data point flows to the right, if the variable is lesser then or equal she goes to the left. We'll do this proceeding until we get into one of the leafs R_1, R_2, R_3, R_4, R_5, that in our presentation, for economy of notation, predicts that the response value will be R_1, R_2, R_3, R_4, R_5 respectively, the consequence over the regressors space can be seen in the figure at the right corner.

The prediction function in this example could be presented by a function like (8):

$$\hat{f}(X) = \sum_{m=1}^{M} w_q I\{(X1, X2) \in \mathbb{R}_m\}, \tag{8}$$

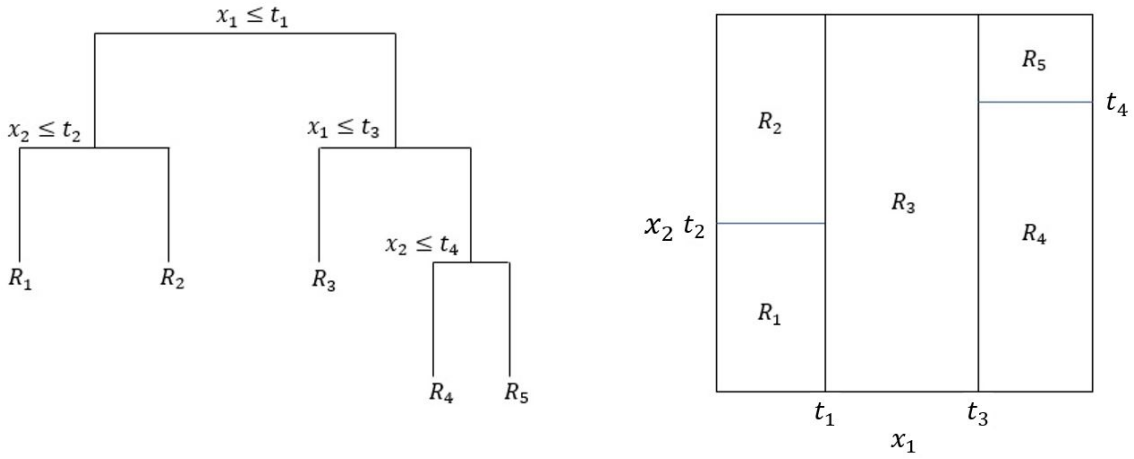


Figure 1 – Example of Decision Tree

where w_q are the constants that will be used for forecasting in each specific region q, and I is an indicator function that show in which region the observation came.

Generalizing this example, the division in one specific point in space is made through an algorithm based on the gradient. Considering all the pairs (x_i, y_i) with $i = 1, 2, \ldots, N$ and $x_i = (x_{i1}, x_{i1}, ..., x_{ip})$ being our regressors, the building tree objective becomes minimize an error measurement. Typically the errors measurements used in classification problems are:

Miss-Classification Error	$1/N_m(\sum_{i \in R_M} I(y_i \neq k(m)) = 1 - \hat{p}_{mk(m)})$
Gini Index	$\sum_{k \neq k'} \hat{p}_{mk}\hat{p}_{mk'} = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$
Cross Entropy	$-\sum_{k=1}^{K} \hat{p}_{mk} log \hat{p}_{mk}$

For effectively find the variable and point to split, consider starting from the root and divide the variable j in point s, the regions that would be generated by this split can be expressed as:

$$R_1(j, s) = \{X | x_j \leq s\} \qquad and \qquad R_2(j, s) = \{X | x_j > s\} \tag{9}$$

The two sets defined in (9) characterize the subdivision in a rule like "bigger/smaller than", in our example if the variable x_j is greater than s it will flow to region 2 and if it is smaller the obs flows to region 1. We can define as our objective minimize one of the errors criteria previously exposed, we'll use as example the cross entropy:

$$min_{j,s}[min_{w1} \sum_{x_i \in R_1(j,s)} \sum_{k=1}^{K} -(\hat{p}_{mk} log \hat{p}_{mk}) + min_{w2} \sum_{x_i \in R_2(j,s)} \sum_{k=1}^{K} (-\hat{p}_{mk} log \hat{p}_{mk})] \tag{10}$$

Where K is the number of classes, the outer minimization in (10) is a reference to which of the j variables will be used to make the split in the s point, the two inner minimizations choose which constant w_1 and w_2 must be allocated to each region for minimize the error measure. This is a general proceeding for finding splits in trees.

2.2.2.1 Boosting Trees

Until Schapire (1991) published his seminal work, the definition of how great should be a tree was treated as a pruning problem. If you intend to do pruning first you let the algorithm make a huge amount of nodes, after that, you start the pruning procedure verifying if the exclusion of a node can improve the forecast quality. Despite of the popularity of the technique it have a disappointing predictive performance in comparison to others classification models like Support Vector Machines and Neural Networks. The ensemble method literature combines a set of different trees to generate one and only one powerful model.

The first propose that we'll revise is boosting that was introduced by Schapire (1991). Following Ferreira e Figueiredo (2012) Boosting is an ensemble methodology that is usually inspired in the idea that a combination of uncorrelated weak learners, models that has predictive ability only a little better than random chance, can generate an strong learner, a model with really small probability of error. Boosting consists in training sequential weak learners and combine them to get an strong learner. The popularity of the technique is partially explained by the fact that computationally is easier to train a sequence of small bad quality models than one robust great model.

To develop intuition define $H_m : \mathcal{X} \to \{-1, +1\}$ as the $m - th$ binary weak learner, with $m = 1, ..., M$, defined into $x \in \mathcal{X}$, the variables used as regressors. Each classifier represents a different way to combine this variables, we'll call this different possibilities of $H_1(x_1), ..., H_M(x_M)$. Combining all this models we obtain only one model. It's a good time to expose a crucial concept of boosting, when we talk that boosting do sequential estimation of models we are saying that the model M depends on the model $M - 1$. In boosting this dependence is achieved by adjusting the target variable by the errors made by all previous models. The rule to make this adjustment change in each application but the philosophy holds.

If each classifier is not correlated with the others we can achieve a good forecasting quality taking the majority vote of them. This decorrelation is achieved though a bootstrap procedure. If our original dataset is composed of M explainable variables we'll use for train each model only a subset of them, usually a real constant $m < M$. So we'll diminish the covariance between models changing their regressors. The combination of all models will always achieve a minor error rate than any model alone (SCHAPIRE; FREUND, 2012). If we give the weight $\alpha_1, ..., \alpha_M$ to each learner, we can represent the model $H : \mathcal{X} \to \{-1, +1\}$ as:

$$H(x) = sinal(\sum_{m=1}^{M} \alpha_m H_m(x))$$

defining our final model as an weighted mean of M forecasts.

Accordingly to Ferreira e Figueiredo (2012) boosting consists in training the same weak learner, in our work a classification tree, in an adaptive algorithm that adjusts the response variable giving more importance to the yet not well good-explained observations. Now let's join the concepts of trees and boosting following the presentation of Chen e Guestrin (2016) the implementation used in this work

Suppose he have a dataset with n observations and m regressors variables x_i $i = 1, \dots m$ and one explained variable y_i, that can be represented as $\mathcal{D} = \{(x_i, y_i)\}$ ($|\mathcal{D}| =$

n_i, $x_i \in \mathbb{R}^m$, $y_i \in \{0, 1, \ldots, k\}$) where k is our number of classes

We've already seen how trees work, they divide the regressors space into hyper-rectangles and assign a real constant w_j to each region so that given a vector $x \in \mathbb{R}_j \rightarrow f(x) = w_j$ we can present the trees as $T(x; \Theta) = \sum_{j=1}^{J} w_j I(x \in \mathbb{R}_J)$, where the parameter $\Theta = \{R_j, w_j\}_1^J$, contains the R_j hyper-rectangles and the w_j constants used to forecast in each of the J regions, usually J is treated as an hyper-parameter previously chosen.

To find this parameters we need to define our objective function, in machine learning typically the function has two parts, an error measurement and a regularization function:

$$obj(\theta) = \mathcal{L}(\theta) + \Omega(\theta). \tag{11}$$

in equation (11) we have the representation of a objective function where the first component $\mathcal{L}(\theta)$ measures the error and $\Omega(\theta)$ controls the over-fitting, this regularization function is the responsible for tree complexity control.

We've already told that boosting is a modeling philosophy that combines different weak-learners, seeking to achieve one combined strong-learner. We can represent our K tree combination as:

$$\hat{y}_i = \phi(\boldsymbol{x}_i) = \sum_{k=1}^{K} f_k(\boldsymbol{x}_i), \;\; f_k \in \mathcal{F}, \tag{12}$$

where $\mathcal{F} = \{f(x) = w_{q(x)}\}$ $(q : \mathbb{R}^m \Rightarrow T, w \in \mathbb{R}^t)$, defining our final model (12) as a sum K trees. To put it simply, for each of our built k trees we'll evaluate the predicted value for the observed regressors x_i in all k trees, the final predicted value $\hat{y}_i$ will be the sum of each predicted value $f_k(x_i)$ in all k trees. In this spirit we can rewrite our objective as:

$$Obj = \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k), \tag{13}$$

equation (13), substitutes the global error function $\mathcal{L}(\theta)$ by the sum of i functions that individually measure the error for each predicted observation.

To find the trees we have to minimize our objective function. According to Friedman, Hastie e Tibshirani (2001) minimize (13) in respect to $f(x)$, can be seen as an numeric optimization procedure in which we seek:

$$\hat{\boldsymbol{f}} = argmin_{\boldsymbol{f}} \, Obj(\boldsymbol{f}),$$

where each individual tree is seen as one parameter and $f \in R^N$ are the approximated functions $f(x_i)$ for our n data-points x_i, $\hat{f} = f(x_1), f(x_2), ..., f(x_n)$.

Analytically this problem is not well-defined and we need a numeric optimization

algorithm to solve this problem, usually, as a sum of trees in which in each step t we have:

$$\begin{aligned}
&\hat{y}_i^{(0)} = 0, \\
&\hat{y}_i^{(1)} = f_1(x_i) = \hat{y}_i^{(0)} + f_1(x_i), \\
&\hat{y}_i^{(2)} = f_1(x_i) + f_2(x_i) = \hat{y}_i^{(1)} + f_2(x_i), \\
&\dots \\
&\hat{y}_i^{(t)} = \sum_{k=1}^{t} f_k(x_i) = \hat{y}_i^{(t-1)} + f_t(x_i),
\end{aligned} \tag{14}$$

where in each iteration the estimated function $f_t(x_i)$ will to be the one that minimizes the following regularized objective function:

$$\mathcal{L}(\phi) = \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k), \tag{15}$$

$$\Omega(f_k) = \gamma T + \frac{1}{2}\lambda |||w||^2, \tag{16}$$

where l is a convex loss function that measures the forecasts quality, in our classification case usually the mlogloss function.

The penalization element (16) helps to smooth the learned functions preventing the over-fitting, when we try to minimize (15) with more trees T or leafs w the proposed addition has to realize a great fall in the error function l to counterbalance the growth of Ω. With this mechanism the algorithm seeks to minimize the objective function with the most simple structure.

We can show that, through a second order taylor expansion this optimization can be rewrote in every step t as

$$\sum_{i=1}^{n} [g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)] + \Omega(f_t), \tag{17}$$

$$g_i = \partial_{\hat{y}_i^{t-1}} l(y_i, \hat{y}_i^{(t-1)}), \tag{18}$$

$$h_i = \partial^2_{\hat{y}_i^{t-1}} l(y_i, \hat{y}_i^{(t-1)}), \tag{19}$$

where (17) became the approximated objective function to the new tree, and (18) and (19) are respectively the Gradient and Hessian of our objective function. Replacing the regularization function by (16) we get to the final equation that define the weights attributed to each leaf and the regularized objective function:

$$w_j^* = -\frac{G_j}{H_j + \lambda} \tag{20}$$

$$Obj* = -\frac{1}{2} \sum_{j=1}^{T} \frac{G_j^2}{H_j + \lambda} + \gamma T \tag{21}$$

$$G_j = \sum_{i \in I_j} g_i, \qquad H_j = \sum_{i \in I_j} h_i. \tag{22}$$

This set of equations gives us a series of informations about the learned trees, equation (20) gives us the attributed values to each of the j leaf-trees that is learned in every step t. Equation (21) is an score function that measures the quality of a tree-structure q. Worth call attention to the fact that both equations (20) and (21) are calculated directly for any loss function with first and second derivatives well-defined.

Accordingly to Chen e Guestrin (2016) usually is impossible to enumerate all q structure. Indeed an algorithm that starts from the root and interactively add new branches is used. Assuming that I_l and I_r are the generated sets after an split where the data-points in the upper father-leaf is $I = I_L \cup I_R$. The split loss function became:

$$\mathcal{L}_{split} = \frac{1}{2}\left[\sum_{j=1}^{T}\frac{(\sum_{i\in I_L} g_i)^2}{H_j+\lambda} + \sum_{j=1}^{T}\frac{(\sum_{i\in I_R} g_i)^2}{H_j+\lambda} - \sum_{j=1}^{T}\frac{(\sum_{i\in I} g_i)^2}{H_j+\lambda}\right] - \gamma \tag{23}$$

Equation (23) is the implemented formula to find the split candidates, the first sum inside the brackets measures the model quality using the observations that flow to the left, the second sum measures the quality in the right, and the third sum measures the error in the parent node before the split is made. You can observe that equation (23) gives us a measure of the contribution that each split has to improve the model quality in reference to the model defined before the split.

2.2.3 Random Forests

Random Forests is another Ensemble-Trees methodology, as this is an extension of Bagging let's first expose it. Bagging is the conjugation of the words Bootstrap and Aggregation, Bootstrap is a famous inference technique for assessing the accuracy of a model, those accustomed with classical statistics know that in there we usually make inference by assuming an error distribution. Bootstrap makes inference by a different way, assuming that our training set has N observations we'll sample with replacement $b < N$ observations in each bootstrap round B. With this B samples we estimate and predict our out-of-sample observations $\hat{y}_i$, the different B samples will generate different $\hat{y}_i$ forecasts, Bootstrap uses the $Var(\hat{y}_i)$ as the estimated forecasting variance, for a review of bootstrap methodology look at Davison e Hinkley (1997).

Bagging is inspired in the bootstrap concept, let's suppose that we want to regress the observations y_i by the m variables $x_i = (x_{1,i}, x_{2,i}, \ldots, x_{m,i})$ with $i = 1, \ldots, N$, and we want to use the model $\hat{f}(x)$ to make this forecast, again we'll select a sample of size b from our dataset B times and fit B models $\hat{f}^{*B}(x)$. Differently from our previously explanation now we are not anymore specially interested in the variance of forecasts, now we are interested in their means. Bagging is a forecast combination methodology that aggregates the B bootstrapping forecasts and uses their mean as our punctual forecast as in 24:

$$\hat{f}_{bag}(x) = \frac{1}{B}\sum_{g=1}^{B}\hat{f}^{*g}(x) \tag{24}$$

This empirical Bagging estimate converges to the true Bagging estimate as $B \to \infty$. The models that usually have the great benefit with this procedure are the ones with low bias and high variance, like bushy Trees! Trees are well-know for the great model variance, the addition or exclusion of one observation can change an entire splitting decision and

the expected value for all observations, but, bushy trees, the ones with lots of branches, usually have low bias, a characteristic that in the earlier stages of tree models caused a lot of problems with the over-fitting phenomenon, but now, we will benefit ourselves this characteristic.

Let's suppose we have B estimated classification-trees $\hat{T}_B(x)$ each one estimated with a sample of size b randomly selected. Let's organize the response vector as a vector with dimension $K-1$ full of zeros with only a one in the k cell, that corresponds to the expected class, if all cells are equal to zero we forecast the K class; our bagged tree $\hat{f}_{bag}(x)$ is defined as a K-vector $[p_1(x), p_2(x), \ldots, p_K(x)]$ where $p_k(x)$ is the proportion of models that forecast the class k, our bagged classifier will forecast the class that has the greatest number of votes from the B trees, $\hat{T}_{bag}(x) = argmax_k \hat{f}_{bag}(x)$. Bagging improve considerably the forecasting performance of trees, Random Forests goes one step further providing even more forecasting capacity.

The variance of the mean $\tilde{\sigma}_B^2$ of B estimates independently and equally distributed decreases accordingly to $\tilde{\sigma}_B^2 = \sigma^2/B$, where σ is the model variance. But if they are not independent this quantity increases proportionally to model correlation ρ, $\tilde{\sigma}^2 = \rho\sigma^2 + [(1-\rho^2)\sigma^2]/B$. In Bagging we are generating B samples that have the same Data-Generating underlying Process, Random Forests adds a new procedure to Bagging trying to decrease the model correlation in the growing tree process. When constructing the classification trees we'll in every split change the underlying DGP by changing the split candidate-variables. In every split we'll select randomly $r < m$ covariates that are eligible for a split and restrict the candidate split variables to this subset, this randomization procedure tries to decorrelate even more the B models. After we grow our B trees $\{T(x; \Theta_g)\}_1^B$ the Random Forests classifier $\hat{T}_{rf}$ is found taking the majority vote, similar to the bootstrap procedure.

$$\hat{f}_{rf}^B(x) = \frac{1}{B}\sum_{b=1}^{B} T(x; \Theta_b) \tag{25}$$

$$\hat{T}_{rf}(x) = argmax_k \hat{f}_{rf}(x) \tag{26}$$

This apparently simple procedure generates a huge improvement into forecasting quality, Random Forests are a modern competitive forecasting method and have numerous applications in different forecasting problems. The number and depth of trees are typically found by cross-validation. One important feature of Random Forests is that usually the addition of new trees doesn't increase the forecasting error, making them robust to over-fitting, when we look to the variance formula we can see that as $B \to \infty$ the variance always decrease, converging to $\rho\sigma^2$, and as we have already pointed the randomization procedure works to decrease ρ. The stopping rule usually is found when the addition of a new tree only increase computer complexity with no decline in estimated variance.

2.2.4 Bayesian Trees and MPBART

Bayesian statistics are a statistical philosophy with high performance in forecasting problems, we will not review the hole philosophy right here, but just present the extension that Chipman, George e McCulloch (1998) made for classification trees and further for Bayesian Additive Regression Trees (BART) (CHIPMAN et al., 2010), recently generalized by Kindo, Wang e Pena (2013) and Kindo, Wang e Peña (2016) to the multinomial case.

We've already talked about trees in section 2.2.2, let's just reformulate all presented information to the Bayesian paradigm. We've already told that a tree structure is compound of two parts, the tree splits also known as the tree-structure that define the hyper-rectangles R and the vector of parameters in the terminal leafs w. Once this two parameters are known the tree is unique and fully specified. So in the usual Bayesian spirit we need to make a prior for our parameters. And will be convenient to decompose our prior as $p(w, R) = p(w|R)p(R)$ implicit stating that the prior for the tree structure is independent of the leafs.

Chipman, George e McCulloch (1998) divide the prior for the tree structure into two parts, the prior for the structure R and for the vector of means w. The prior for the structure can be decomposed in two parts, first the split prior $p_{split}(\eta, R) = \alpha(1 + d_\eta)^{-\beta}$ where d_η is the depth of an specific node in the three with $\alpha \in (0, 1)$ and $\beta > 0$, both are chosen in such a way that they regularize the trees, decreasing the probability of a split as the growing procedure takes place. He also defines the prior split variable in an specific rule $p_{RULE}(\rho|\eta, R)$ using the uniform distribution as default, allocating equal probability to all regressors be chosen before the growing procedure starts, they also designate equal uniform distribution in the space of available split points s for a chosen split variable x_i.

The respective Classification Trees will have as response the vector w_b of dimension $K - 1$. When all values inside w_b are negative the forecast class will be C_K and when there are at least one positive element inside the vector, the forecast class C_k will correspond to the highest element $w_{b,k}$ inside the vector w_b where b are the model leaves. The prior for the b leaves parameters $w_b = (w_{b,1}, \ldots, w_{b,K-1})$ usually follows a normal distribution $w_{b,k}|T \overset{i.i.d}{\sim} N(w, \tau^2)$ for all $k = 1, \ldots, (K - 1)$ classes predicted into the b leaf, with suggested values $\mu = 0$ and $\tau = 3/(r\sqrt{n_T})$, and suggested parameters $r = 2$, $n_T = 200$.

Evaluate the posterior $p(T|X, Y) = \int p(Y|X, T)p(T)$ usually is a hard task and a Metropolis Hasting (MH) Algorithm generates an Ergodic Markov Chain of Stochastic Trees $T^0, T^1, ...$ that after a sufficient burn-in approximates the real-tree. In each step of MH we choose a new candidate tree T^* derived from the transition kernel $q(T^i, T^*)$, and evaluate the acceptation probability:

$$\alpha(T^i, T^*) = min\left\{\frac{q(T^*, T^i)}{q(T^i, T^*)}\frac{p(Y|X, T^*)p(T^*)}{p(Y|X, T^i)p(T^i)}, 1\right\}, \tag{27}$$

if the candidate is accepted we replace $T^{i+1} = T^*$, otherwise $T^{i+1} = T^i$. To complete the model let's talk about $q(T^*, T^i)$. In every step of the iteration we will make one of the respective movements with the correspondents probabilities: **GROW** a terminal node (0.25), **PRUNE** a pair of terminal nodes (0.25), **CHANGE** a non-terminal rule (0.40), **SWAP** a rule between parent and child (0.10).

Now that we presented the necessary modifications into the Tree specifications for the Bayesian philosophy, let's talk about an ensemble of Bayesian Trees, specifically we'll follow the specification proposed by Kindo, Wang e Peña (2016) and we'll need to add a new conception for our problem. We've already told that we can view our classification regression as surging from a response latent vector that now we'll call by z_i with dimensions $K - 1$, when all elements of this vector became negative we'll forecast class K, and other

else if at least one of them is positive we forecast the correspondent class as in (28):

$$y_i(z_i) = \begin{cases} \text{j} & \text{if } \max(z_i) = z_{ij} > 0, \\ \text{K} & \text{if } \max(z_i) < 0. \end{cases} \tag{28}$$

This vector z_i is a function of the regressors variables plus a random noise:

$$z_i = G(X_i; T, M) + \epsilon, \quad \text{for } i = 1, \ldots, n, \tag{29}$$

$$G(X_i; T, M) = (G_1(X_i; T, M), G_2(X_i; T, M), \ldots, G_{K-1}(X_i; T, M))' \tag{30}$$

$$\epsilon_i = (\epsilon_{i,1}, \ldots, \epsilon_{i,K-1})' \sim N(0, \Sigma). \tag{31}$$

Where each element inside (30) is specified as a sum of n_T where $n_T = 1, \ldots, j$ classification trees with respective structures $R_{k,j}$ and leafs parameters $w_{k,j} = w_{k,j,1}, \ldots, w_{k,j,l}$ for all the $l = 1, \ldots, b$ trees terminal nodes :

$$G_k(X_i; T, M) = \sum_{j=1}^{n_T} g(X_i, R_{kj}, w_{kj}) \quad \text{for } k = 1, \ldots, (K-1), \tag{32}$$

The new parameter that we need to specify the prior is Σ. Rewriting this model as a augmented latent model $\tilde{z}_i = G(X_i; T, \tilde{M}) + \tilde{\epsilon}_i$, with $\tilde{\epsilon}_i \sim N(0, \tilde{\Sigma})$ where $\tilde{z}_i = \alpha z_i$, $\tilde{\epsilon}_i = \alpha \epsilon_i$, $\tilde{\Sigma} = \alpha^2 \Sigma$, $\tilde{w}_{k,j} = \{\alpha w_{k,j,l}\}$. Kindo, Wang e Peña (2016) using a constrained inverse Wishart distribution for $\tilde{\Sigma} \sim$Inv-Wish$(v, \alpha_0^2 S)$ and $\alpha^2 | \Sigma \sim \alpha_0^2 tr[S\Sigma^{-1}]/\chi^2_{v(K)}$ place the following prior for Σ:

$$p(\Sigma) = \int p(\Sigma, \alpha^2) p(\alpha^2 | \Sigma) d\alpha^2 \propto |\Sigma|^{-\frac{v+k}{2}} (tr[S\Sigma^{-1}])^{-\frac{v(K-1)}{2}} \tag{33}$$

The posterior is evaluated through a Gibbs Sampler with three steps. First we sample (z, α^2) from the first Gibbs block:

$$(z, \alpha^2) | T, M, \Sigma, y,$$

first obtaining random draws for z_i sequentially accordingly to $z_{i,k} | z_{i,(-k)}, T, \Sigma \sim N(w_{ik}, \psi_{ik})$ where this density has a lower truncation if $y_i = k$ and an upper truncation point if $y_i \neq k$ both at $max\{0, max(z_{i(-k)})\}$ with mean and variance given by (34), (35):

$$m_{ik} = G_k(X_i; T, M) + \sigma_{k(-k)} \Sigma^{-1}_{(-k)(-k)} [z_{i(-k)} - G_{(-k)} G_{(-k)}(X_i; T, M)], \tag{34}$$

$$\psi_{ik} = \sigma_{kk} - \sigma_{k(-k)} \Sigma^{-1}_{(-k)(-k)} \sigma'_{k(-k)}, \tag{35}$$

where $\sigma_{k(-k)}$ is the k column of Σ excluding σ_{kk} and $\Sigma_{(-k)(-k)}$ is the Σ matrix excluding the k column and row. With this sample we find $(\alpha^*)^2 \sim p\{\alpha^2 | \Sigma\}$, and successively $\tilde{z}_i^* = \alpha^* z_i$.

Secondly we sample $(T, \tilde{M}^*)$ from the second block

$$(T, \tilde{M}^*) \sim p\{T, \tilde{M} | (\tilde{z}_i^*)_{i=1,\ldots,n}, \Sigma, (\alpha^*)^2, y\}$$

, through the back-fitting algorithm from Chipman et al. (2010), defining the resulting residuals from all trees excepting the $k-th$ tree, we can define $\tilde{\Sigma}$, $\tilde{z}^*_{i(-k)}$ and $(\alpha^*)^2$ as $z^\dagger_{ik} = \tilde{z}^*_{ik} - \sum_{l\neq j}^{n_T} g(X_i, T_{kl}, \tilde{M}_{kl}) - \tilde{\sigma}_{k(-k)}\tilde{\Sigma}^{-1}_{(-k)(-k)}[\tilde{z}^*_{i(-k)} - G_{(-k)}(X_i; T, \tilde{M})]$, with this residuals the problem of fitting a single tree can be seen as the same problem of estimating a single tree with the previous 4-step procedure proposed by Chipman, George e McCulloch (1998), the only difference is that the response vector will be the residuals from all other trees instead of the original response vector:

$$z^\dagger_{ik} = g(X_i, Tkj, \tilde{M}_{kj}) + \epsilon^\dagger_{ik}, \quad \epsilon^\dagger_{ik} \sim N(0, \tilde{\psi}_{ik}), \tag{36}$$

with $\tilde{\psi}_{ik} = (\alpha^*)^2$,iteratively with this procedure we go updating $M = \tilde{M}^*/\alpha^*$.

Finally we sample Σ, α^2 from the last Gibbs Block using the distribution:

$$\tilde{\Sigma}^* \sim \text{Inv-Wish}(v+n, \tilde{S} + \sum_{i=1}^{n}[\tilde{z}^*_i - G(X_i; T, \tilde{M}^*)][\tilde{z}^*_i - G(X_i; T, \tilde{M}^*)]') \tag{37}$$

followed by an estimation from $\alpha^2 = tr(\tilde{\Sigma}^*)/K$ and the transformation $\Sigma = \tilde{\Sigma}/\alpha^2$.

Since Chipman et al. (2010) first proposed this Bayesian ensemble tree philosophy a lot of different work have verified the competitive ability of this model, and we'll include him in our evaluation.

2.3 Performance Evaluation

All presented models will be estimated with the dataset presented in section 2.1. As previously exposed we'll evaluate the impact of the meta-parameter m in the forecasting performance, as already presented this hyper-parameter is the minimum oscillation required of an asset $|R_{t+1}| < m$ to we consider that we have a return different from null.

We'll use three descriptive statistics to evaluate the models. The first will evaluate the global accuracy of the model and the second and third measures will evaluate the performance of forecasts in an specific class.

Let's define a positive forecasts as a preposition that an specific observation belongs to one class and a negative forecasts as a preposition that an specific observation doesn't belong to an specific class. We can define TP (True Positive) as the number of times that we made a positive forecast and we were right, in a similar way we can define FP (False Negative) as the number of times that we make a negative forecast and we were wrong. Analogously we can define FP(False Positive) and TN (True Negative). With this concepts we can define our errors measurements as:

$$ACC = \frac{TP + TN}{TP + FP + FN + TN}, \tag{38}$$

$$PPV = \frac{TP}{TP + FP}, \tag{39}$$

$$F1 = \frac{2TP}{2TP + FP + FN}. \tag{40}$$

Equation (38) is the definition of Accuracy, that can be understood as a description of the systematic model errors. Also can be understood as the combination of two error measurements: The random error, where we make mistakes by unseen and unpredictable changes in the true DGP, and the systematic errors due to imperfections in the model estimation.

Equation (39) is the precision definition, or true positive rate; it represents the proportion of values correctly predicted as positive. As negotiations are only generated when we make a positive forecast in the extremes classes, never in the null return, we'll verify the quality of the forecasts specifically for this two classes. Doing that we can isolate the mistakes for when we predict that a class is null once they don't insert profits or losses for the trader wallet.

By end the equation (40) presents the definition of F1-Score. She is usually made for binary classes but we'll use her here because she provides a balanced measure between precision and sensitivity. Requiring that good models both make correct positive and negative forecasts, requiring not only that the models be right at the positive forecasts but also in the negatives.

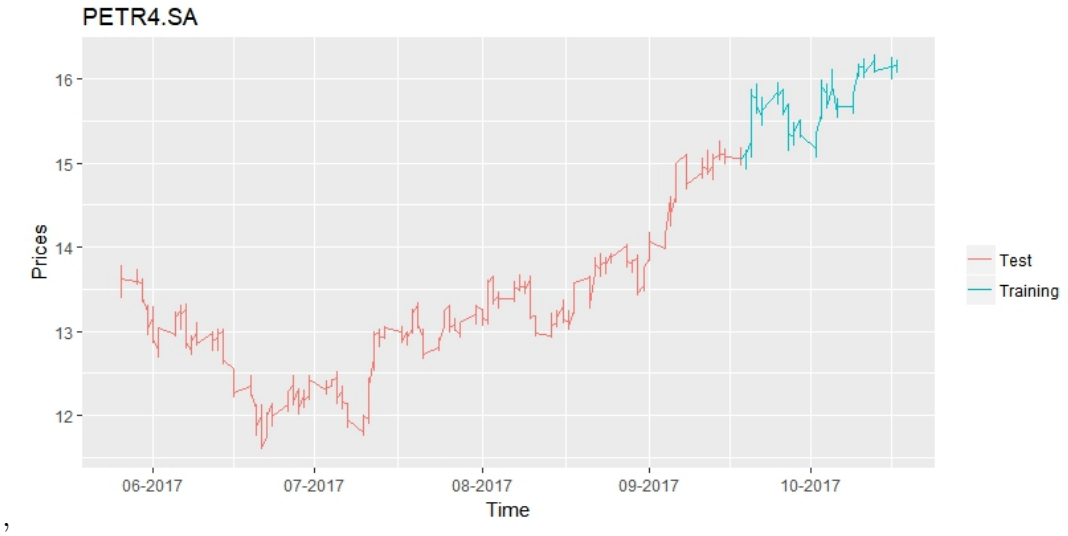
In the end we'll make a financial simulation out-of sample. When the model predicts that the sign of returns will be positive/negative we'll simulate a buy/sell. When we forecasts that the return will be null we will not compute anything.

3 Simulation Results

This section will present a series of results, first we'll focus in the importance of the different frequencies, after that we'll verify the importance of different frequencies combined with 4 different thresholds, after improving our intuition for the results we'll expand our forecasting window and make trading a financial simulation.

3.1 Minimum Threshold and Optimum Frequency

The sample that we'll use in this first part is composed by one-hundred days going from 26/05/2017 to 17/10/2017, Figure 2 displays the intra-day historic price from asset PETR4.SA. Into the figure the red series references the data used for the first estimation and the blue series makes reference to the sample used to validation, as the rolling window goes the series in red moves over the blue.



,

Figure 2 – Price History - PETR4.SA

3.1.1 Frequency Analysis

Were constructed 60 databases with different temporal aggregations, in each successive basis we enlarge the sampling frequency by one minute, the most disaggregate basis is from 1 minute and the most aggregate one is on 60 minutes, in this first part we'll use the minimum threshold 0,01%.

We'll evaluate the forecast quality for all 4 models: Boosting Trees, Random Forests, BART and Logistic regression in all 60 different frequencies. Figure 3 presents the accuracy of the forecasts for the 4 models, in the vertical axis we have the accuracy score and in the horizontal axis we have all different frequencies, as we go to the right we aggregate more our dataset:

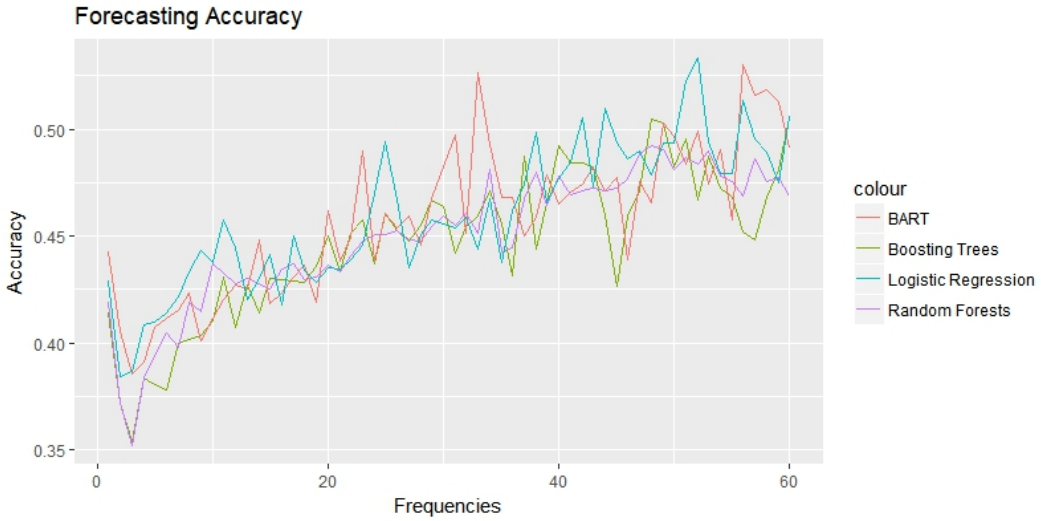


Figure 3 – Forecasting Accuracy

As we can see into 3 as long we aggregate more our data-set the model became more accurate. This behavior is natural, as we came from the lower frequencies to the biggest we slowly get out from a 3 label problem to a 2 label problem, in the 60 minutes database is really rare that the price don't change, but in the 1 minute forecast this is super common. Perhaps a more interesting statistics would be verify the number of times that we predict correctly into one of the extreme class, positive or negative, but the realized value was in the exactly opposite side, figure 4 presents this statistics.

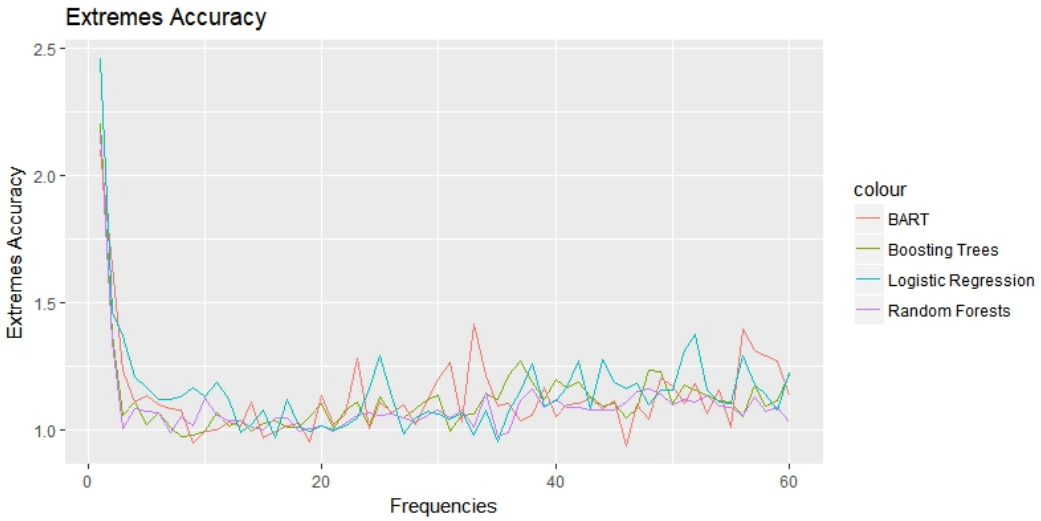


Figure 4 – Forecasting Accuracy

We can see in 4 that this proportion have a huge value in the smallest classes and reaches the minimum a little before the 20 minutes, after that the curves slowly raises as we aggregate more our dataset. We could also verify if the two signs, positive and negative has the same forecasting quality. Again we'll discard the mistakes made to the null class, simple because despite loss from transaction costs, this mistakes doesn't generates *financial*

loss. Figure 5 presents the rate of negative returns that were correctly identified and 6 presents this rate for positive returns.

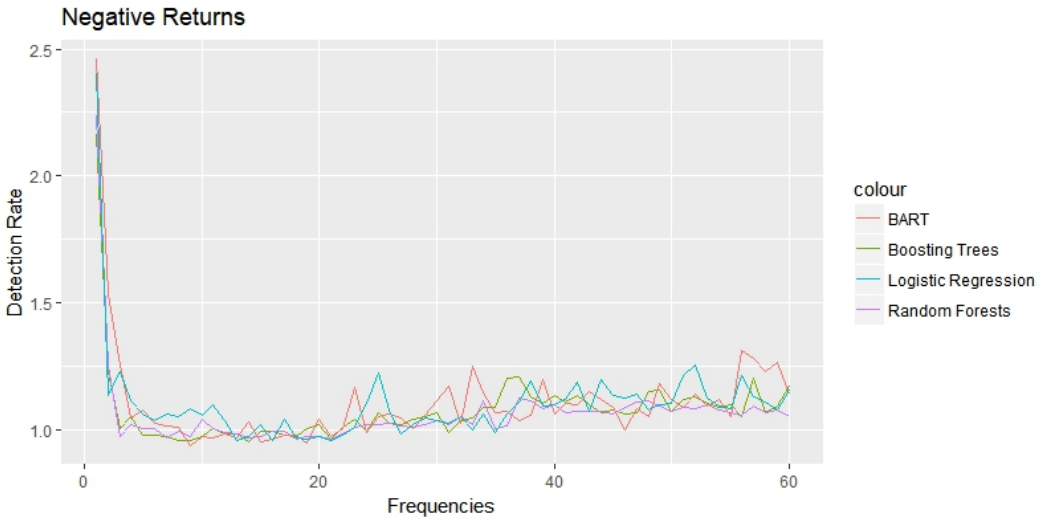


Figure 5 – Forecasting Accuracy on Negative Returns

Can we see that the negative class has a similar detection rate to the both class combined , but a completely different scenario happens in the positive class presented in 6, in there we have most of the model usually getting a detection rate above 2 with some outliers reaching more than 10 times more accuracy.

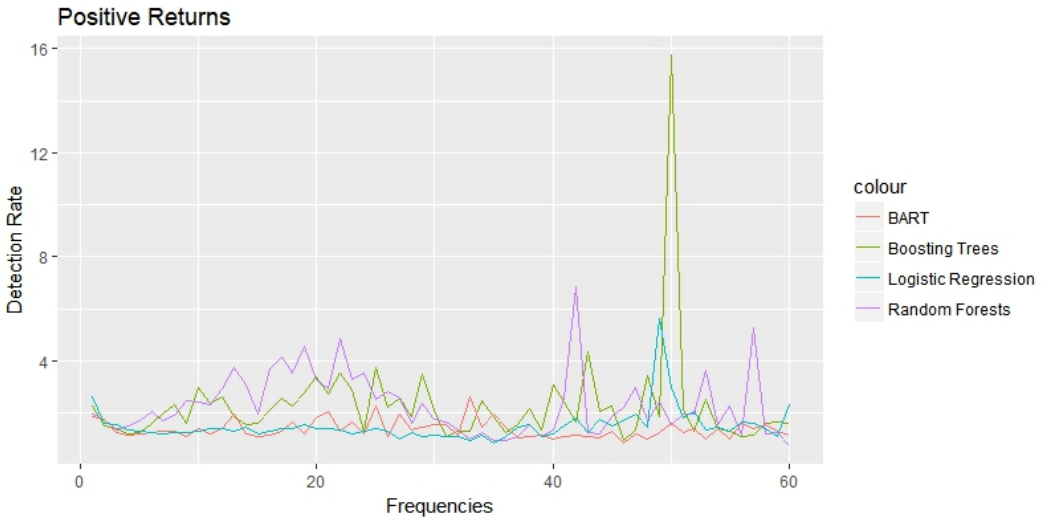


Figure 6 – Forecasting Accuracy

Despite of the great forecasting quality in the positive returns class we have to call the attention of the reader that this class is really low predicted for all our models, despite of the price raises in during the test period, that's why the overall accuracy look more with the performance of negative returns than with positives.

When we do a financial simulation in this 20 final days we don't really get a clear picture. Figure 7 shows the ending balance of a short financial simulation in this 20 days,

when we predict the positive sign we simulate a purchase and when we forecast the negative sign we simulate a sale, when we foresee *"null-return"* we didn't simulate anything. The vertical axis presents the budget in Brazilian Real (R$), we make this simulation with only one stock for getting intuition, the mid price during this 20 days was close to R$15,50.

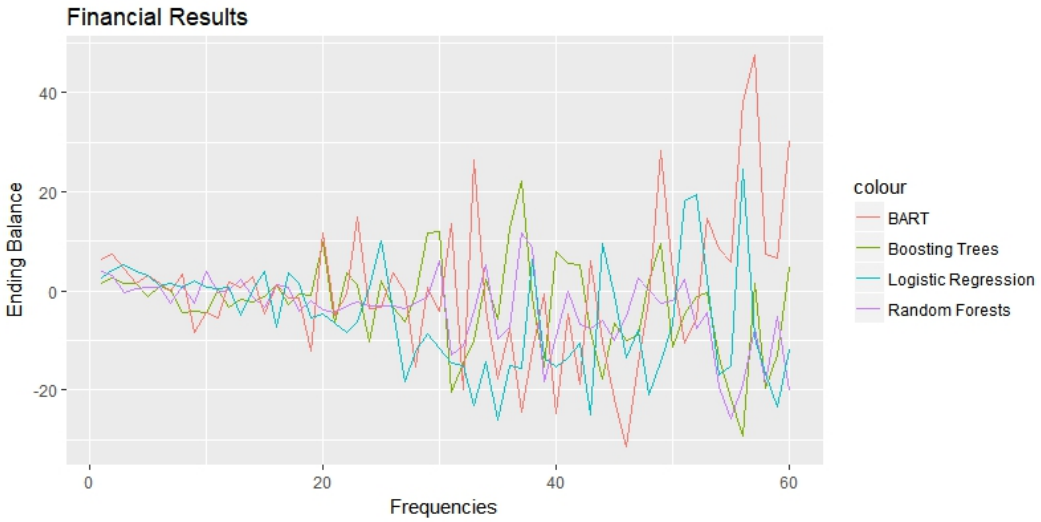


Figure 7 – Financial Simulation

Figure 7 shows to us that the variance of financial results increases dramatically as we start to aggregate more our dataset, starting in the lowest frequencies with strictly positive results and goes to the biggest frequencies with a variety of final results, positives and negatives.

In summary despite some of the highest frequencies presents nice statistical properties that would encourage us to adopt them, the financial results has a biggest variance.

3.1.2 Minimum Threshold

Following this first analysis we selected only two models for make the analysis of the impact of a minimum threshold, we selected the Boosting Trees and the Logistic Regression. This is simply due to this two models be the faster allowing we'll evaluate bigger space of different combinations.

We classified the returns as Positives/Negatives/Null requiring 0,01%, 0,05%, 0,1%, 0,5% of price oscillation. How the price was approximately R$14,00 in our sample this oscillations are approximately correspondent to R$0,01, R$0,06, R$0,14 and R$0,65 cents of Brazilian Real. Figure 8 presents the accuracy of the forecasts for all 60 analyzed frequencies. In the horizontal axis we have again the data bases frequencies, in the vertical axis we have the Accuracy score of both models, Boosting Trees and Logit, for all 4 analyzed thresholds.

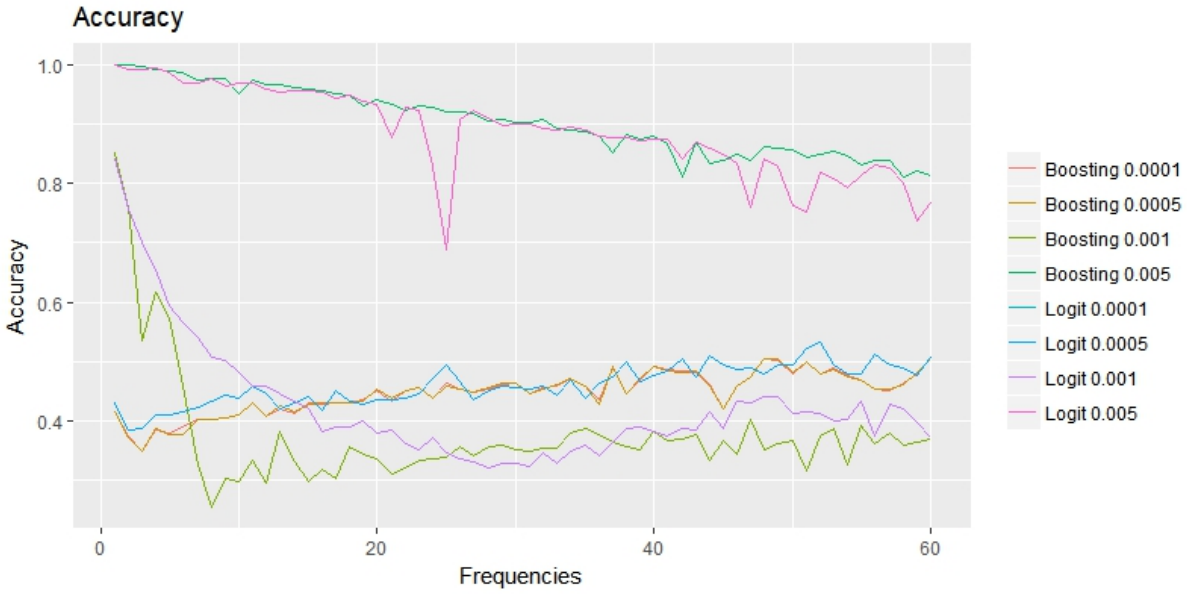


Figure 8 – Forecasting Accuracy

The figure 8 tells us that the highest levels of accuracy occur in the smallest aggregations together with the biggest thresholds. As long we aggregate more our data less accurate became the forecasts for this threshold, we believe that this behavior is due to in the small windows doesn't occur many situation the prices oscillates 0,5%. An interesting behavior also occurs with the second greatest threshold 0,1% he starts the forecasts with a great accuracy but suddenly fall only starting to recover the forecasting quality after the 20 min frequencies. The other two thresholds presents a constantly and improving forecasting behavior during as we aggregate more our dataset, it's nice to remember that the smallest threshold 0,01% is really close to the asset tick.

Figures 9 and 10 present the forecast precision evaluated only at the positive and negative classes. Similar to the last graphic the horizontal axis presents the 60 evaluated frequencies and the vertical axis presents the precision in that specific time aggregation, the lines display the precision for each model with each possible threshold.

We can see at figure 9 that in the class of negative returns the best models were the ones with the smallest minimum thresholds. We can also see that as we aggregate more the data more accurate became the forecasts. Figure 10 presents the same statistics but for the class of positive returns. We can see at figure 10 that the models forecasting with highest thresholds achieved the smallest forecast precision. But now we don't have a consistent improvement in the forecast quality as we aggregate more our dataset.

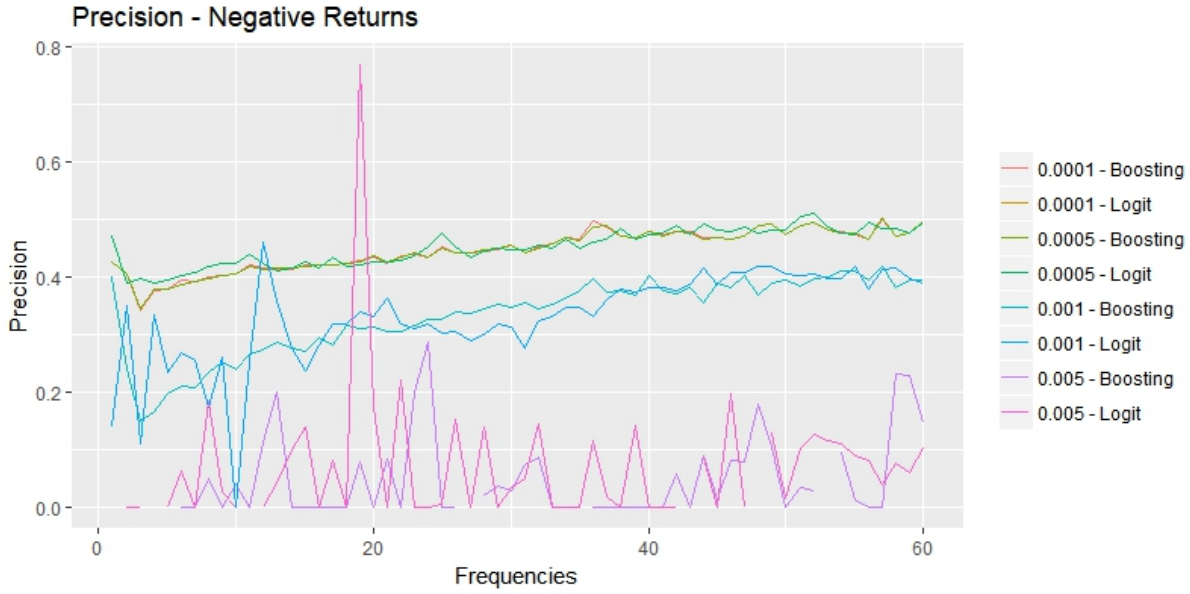


Figure 9 – Negative Returns - Precision

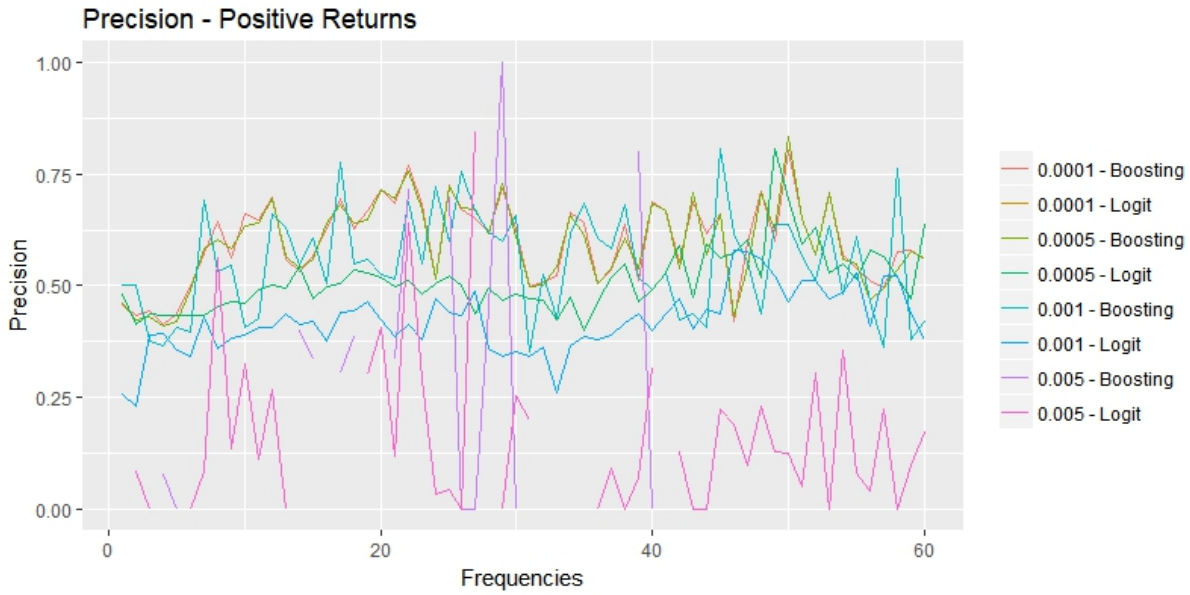


Figure 10 – Positive Returns - Precision

Figures 11 and 12 presents the F1 statistics for this predictions evaluated only for the negative and positive returns. We can see at figure 11 that the F1 score in the class of negative returns have a simylar behaviour to the precision. Again as the we aggregate more our dataset bigget became the F1 score, again we verify that the best statistical forecasts happens with the models forecasting the smallest thresholds 0,01% e 0,05%. Figure 12 presents the same statistics for the class of positive returns. Again we see at figure 12 that in the positive returns the analysis is less clean. Again when we evaluate the F1-Score, there's no benefit in aggregating more our dataset. But there exists a visual benefit of using smallest thresholds.

Figure 11 – Negative Returns - F1-Score

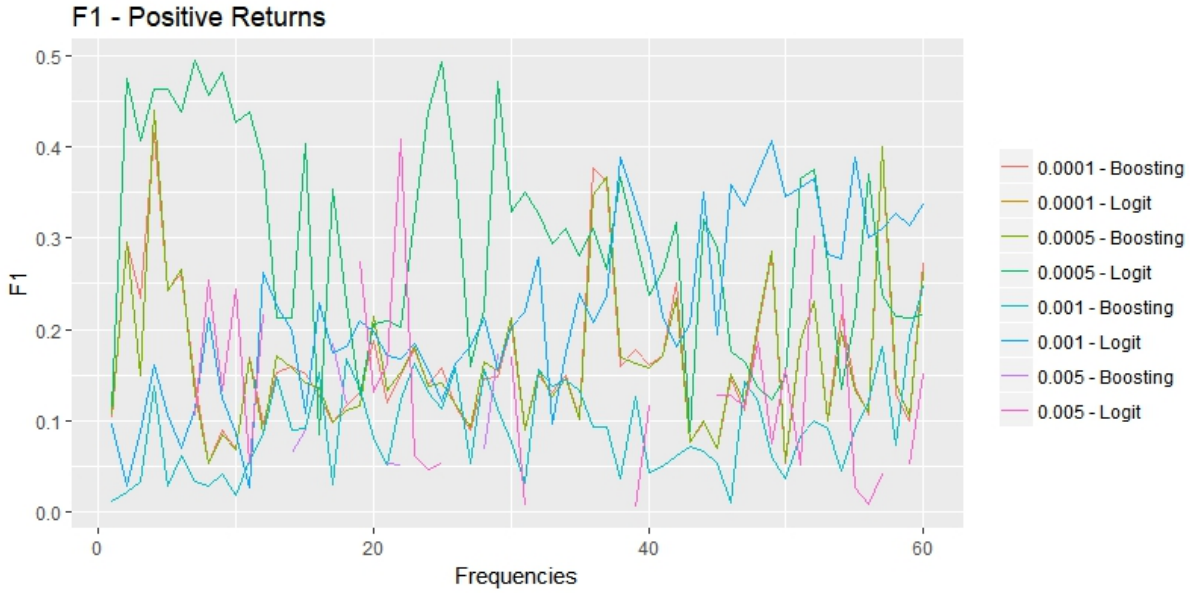


Figure 12 – Positive Returns - F1-Score

The previous analysis allow us verify that the highest accuracy from the models with highest thresholds is probably due to the lower occurence of this great returns in small windows of times. We can also verify that seems to have little difference between the forecasts that uses as thresholds for a minimum classification 0,05% or 0,01%, but the second is more theoretical attractive as this is the real minimum tick of the traded asset.

To refine the analysis we again make a short financial simulation with only 20 days to evaluate the forecasting quality. In all positive returns forecasts we simulated a purchase, and in negative forecasts we simulate a sale, to make the simulation a little different we

change the lot of shares for the minimum required to trade at B3.SA, 100 shares. The final balance of all models with respective threshold is presented at figure 13.

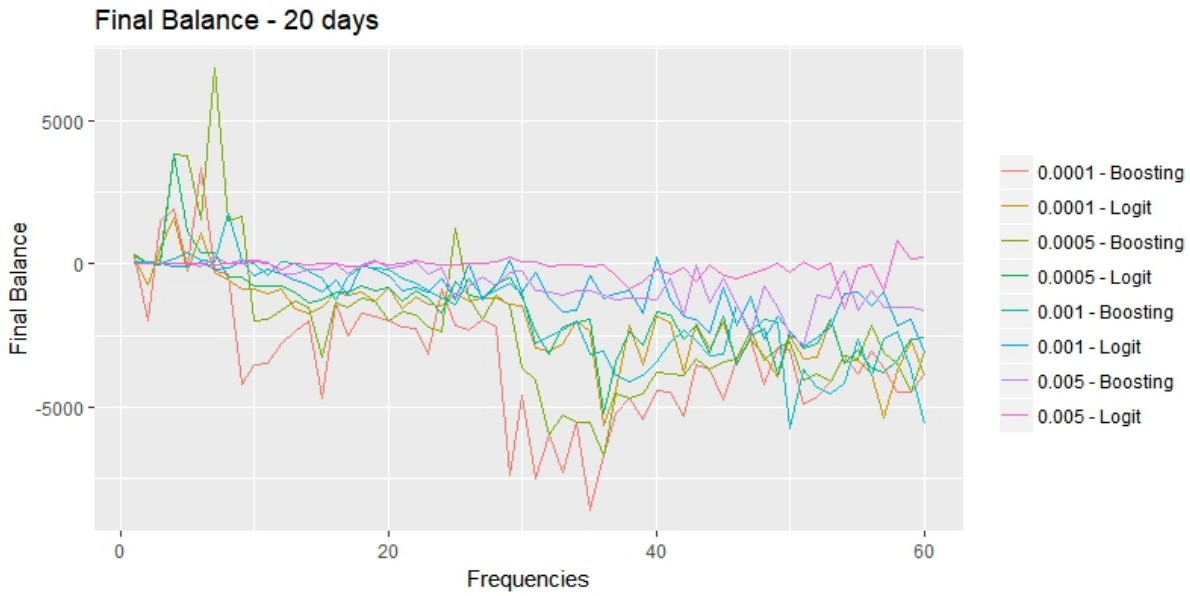


Figure 13 – Simulation Final Balance

Figure 13 shows that the unique strategies that would finalize the simulation with positive final balance would be the ones with smallest frequencies specially going from 4 to 8 minutes. All others frequencies would result in negative balances.

3.2 Financial Simulation

In the last section 3.1 we verified some interesting behaviors, first of all seem to have a great increase in forecasting accuracy as we move to the greatest forecasting windows, but we never see this greater accuracy translate into profit, indeed we verified that only at the small frequencies the models achieve positive final balances. We also didn't see a great benefit in putting a high threshold for the forecasting quality, despite the greatest forecasting performance of the highest threshold in the smallest frequencies, when we look for the performance in the specific classes we understand that the great overall performance in probably due to the small percentage of situations where the prices oscillates great enough to activate the threshold.

Now let's verify the quality of forecasts five minutes ahead requiring only 0.01% as minimum return threshold for classify the return as different from null, different from the previous analysis we will expand our test-set. This simulation will have a significant difference from the previous one, how we verified that great forecasting horizons seems to have good statistical properties, we'll try to insert their quality in small frequencies forecasts, together with the variables sampled in intervals of 5 minutes we will also add variables aggregated by the last 20 and 60 minutes. This will let us add the notion of acceleration to the model and hopefully achieve best statistical results.

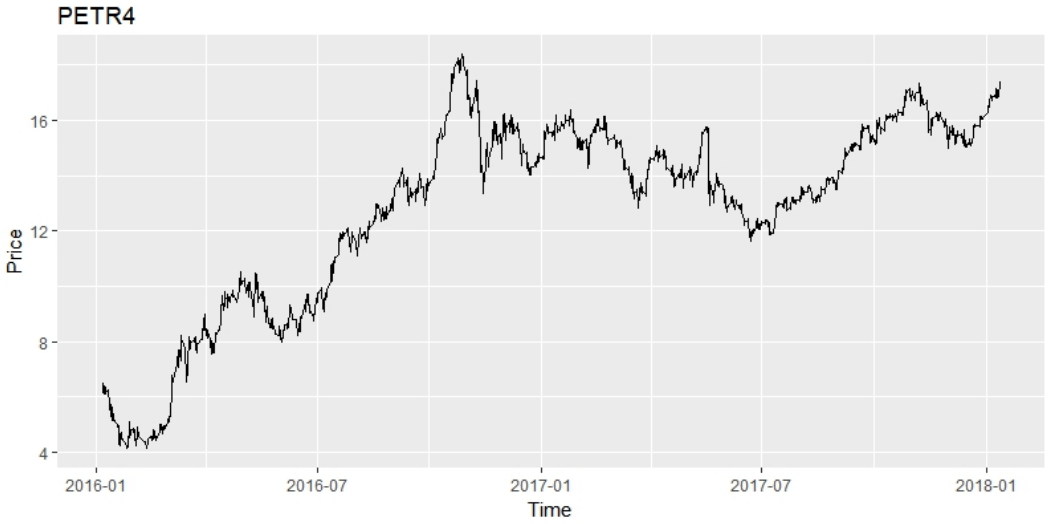


Figure 14 – PETR4.SA

Source – Elaborado pelo Autor

In this simulation our data-base will start at 2015/12/01 and end at 2018/06/12 the historic prices is presented at figure 14. Now we'll make a rolling window simulation with the estimation set having the fixed number of 100 days and test set having the fixed number of 10 days, after each forecasting round we'll reestimate the whole model including the new 10 days and excluding the 10 olds, the regular rolling window methodology. Let's first present some statistical properties of the forecasts. In table 2 we have the forecasting accuracy of our 4 models with respective confidence intervals at 95% of confidence

Table 2 – Forecasting Accuracy - 500 days

Algorithm	Boosting Trees	Random Forests	B.A.R.T.	Logit
Accuracy	0.4113	0.4135	0.4124	0.3988
C.I. Superior	0.4142	0.4164	0.4152	0.4016
C.I. Inferior	0.4085	0.4107	0.4095	0.396

We can see at table 2 that the overall performance of all tree based methods are really similar, with the confidence intervals overlapping, the only model that appears to be significant worst than the others is the logistic regression, in addition to that if we think of ourselves in a 3 category regression we had achieve a performance that is superior to the 33,33% baseline

In table 3 we can verify some statistical forecasting qualities of the models for the relevant classes, the positive and negative returns, we present the F1 statistic and the Positive Predicted Value (PPV) Rate.

Table 3 – Forecasting F1 - 500 days

Algorithm	Boosting Trees	Random Forests	B.A.R.T.	Logit
F1 - Positive	0.4726	0.4729	0.4981	0.4894
F1 - Negative	0.4392	0.4465	0.4101	0.3811
PPV - Positive	0.4214	0.4194	0.4090	0.4006
PPV - Negative	0.4269	0.4221	0.4211	0.4054

We can see by table 3 that we don't have a general better model, BART has a greater F1 in the positive class but Random Forest is better at the negative class, in addition Boosting seems to be the best model when we verify the Positive Predicted Value statistic. When we look to the statistical results we don't really have a direction to say which model seems to be the best.

A good way to differentiate all the models is their financial result, in figure 15 we present the wallet evolution of a trader that trades only one stock in every entry, again the vertical axis is in Brazilian Real (R$), but now in the horizontal axis we present the evolution of the financial balance.

Figure 15 – Financial Simulation

We can verify by figure 15 that all models generates a nice forecasting result with the Boosting Trees model achieving the highest financial result, again logistic regression seems to be the worst choice, worth call the attention that PETR4.SA was being traded by an average of R$12,79 in our sample period so a final profit of R$70 is a really exciting result. In a similar spirit of the sharp index we divided the average results of this models by their standard deviation, the results are at table 4

Table 4 – Forecasting Accuracy - 500 days

Algorithm	Boosting Trees	Random Forests	B.A.R.T.	Logit
Avg. Fin. Ret.	0.001065	0.000987	0.000983	0.000986
Std. Fin. Ret.	0.035269	0.033709	0.033701	0.033724

Table 4 shows that the highest average return provided by the boosting trees model is also benefited by a lower risk return proportion. But shaw we call your attention for the negligible average financial return, as already has been stated before the asset was being traded by an average of R$12.79 in our test window, and our average returns can't beat the traditional market costs we would face, we would need an average cost of less than 0,001% to make this strategy profitable.

4 Conclusions

We reviewed three of the state of art ensemble-tree methodologies in this work: Random Forests, Boosting and BART, together with the logistic regression. And looked for their performance when forecasting high-frequency financial signals. We also investigated the impact of the temporal aggregation of the series and a threshold for classify a return as different from null.

First we verified that as more we aggregate our dataset usually the models start to present better statistical properties, but on the other hand only the smallest frequencies have a positive financial result. In addition we verified that the the forecasting quality gets better and worst in a similar way by all models, indicating that all findings are characteristics of the true DGP.

Secondly we verified that require higher thresholds to classify a return as different from null can generate better statistical properties, but this came with a cost, the lack of predictions in extreme classes, the positive and negative returns, majority forecasting in the null class. In addition to that in the smallest thresholds usually we go improving our forecasting performance as we aggregate more our data, and in the highest thresholds the behavior is the extreme opposite.

By end we verified that all algorithms analyzed achieve a great forecasting performance in our 500 days simulation, all models finalize the period with profits. But we have to call the attention for the Boosting Trees implementation of Chen e Guestrin (2016), they again achieved a great forecasting performance and provide the best risk return relationship. But we shall point that the average returns of all models would probably be corrupted by the transaction costs, a future problem that will have to be analyzed.

In conclusion, we verified that smallest frequencies and smallest thresholds provide good trading strategies independently of the used model, with special attention for Boosting Trees, but none of the models achieve sufficient profits to compensate the transaction costs.

Bibliography

BREIMAN, L. Random forests. *Machine learning*, Springer, v. 45, n. 1, p. 5–32, 2001. Citado 2 vezes nas páginas 2 and 6.

BREIMAN, L. et al. *Classification and regression trees*. [S.l.]: CRC press, 1984. Citado na página 2.

CAO, C.; HANSCH, O.; WANG, X. The information content of an open limit-order book. *Journal of futures markets*, Wiley Online Library, v. 29, n. 1, p. 16–41, 2009. Citado na página 2.

CASALS, J.; JEREZ, M.; SOTOCA, S. Modelling and forecasting time series sampled at different frequencies. *Journal of Forecasting*, Wiley Online Library, v. 28, n. 4, p. 316–342, 2009. Citado na página 9.

CHEN, T.; GUESTRIN, C. Xgboost: A scalable tree boosting system. p. 785–794, 2016. Citado 4 vezes nas páginas 8, 13, 16, and 33.

CHIPMAN, H. A.; GEORGE, E. I.; MCCULLOCH, R. E. Bayesian cart model search. *Journal of the American Statistical Association*, Taylor & Francis Group, v. 93, n. 443, p. 935–948, 1998. Citado 3 vezes nas páginas 17, 18, and 20.

CHIPMAN, H. A. et al. Bart: Bayesian additive regression trees. *The Annals of Applied Statistics*, Institute of Mathematical Statistics, v. 4, n. 1, p. 266–298, 2010. Citado 3 vezes nas páginas 2, 17, and 20.

CHRISTOFFERSEN, P. et al. Direction-of-change forecasts based on conditional variance, skewness and kurtosis dynamics: international evidence. 2006. Citado 3 vezes nas páginas 1, 3, and 5.

CHRISTOFFERSEN, P. F.; DIEBOLD, F. X. Financial asset returns, direction-of-change forecasting, and volatility dynamics. *Management Science*, INFORMS, v. 52, n. 8, p. 1273–1287, 2006. Citado 2 vezes nas páginas 4 and 5.

CRAMÉR, H. *Mathematical Methods of Statistics (PMS-9)*. [S.l.]: Princeton university press, 2016. v. 9. Citado na página 5.

CZEPIEL, S. A. Maximum likelihood estimation of logistic regression models: theory and implementation. *Available at czep. net/stat/mlelr. pdf*, 2002. Citado 2 vezes nas páginas 10 and 11.

DAVISON, A. C.; HINKLEY, D. V. *Bootstrap methods and their application*. [S.l.]: Cambridge university press, 1997. v. 1. Citado na página 16.

DIXON, M. Sequence classification of the limit order book using recurrent neural networks. *Journal of Computational Science*, Elsevier, 2017. Citado na página 6.

DIXON, M. F.; KLABJAN, D.; BANG, J. H. Classification-based financial markets prediction using deep neural networks. 2016. Citado na página 2.

FERREIRA, A. J.; FIGUEIREDO, M. A. Boosting algorithms: A review of methods, theory, and applications. In: *Ensemble Machine Learning*. [S.l.]: Springer, 2012. p. 35–85. Citado na página 13.

FLETCHER, T.; HUSSAIN, Z.; SHAWE-TAYLOR, J. Multiple kernel learning on the limit order book. In: *Proceedings of the First Workshop on Applications of Pattern Analysis*. [S.l.: s.n.], 2010. p. 167–174. Citado na página 7.

FLETCHER, T.; SHAWE-TAYLOR, J. Multiple kernel learning with fisher kernels for high frequency currency prediction. *Computational Economics*, Springer, v. 42, n. 2, p. 217–240, 2013. Citado 2 vezes nas páginas 2 and 7.

FREUND, Y.; SCHAPIRE, R. E. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, Elsevier, v. 55, n. 1, p. 119–139, 1997. Citado na página 2.

FRIEDMAN, J.; HASTIE, T.; TIBSHIRANI, R. *The elements of statistical learning*. [S.l.]: Springer series in statistics New York, 2001. v. 1. Citado na página 14.

HAN, J. et al. Machine learning techniques for price change forecast using the limit order book data. *Machine learning*, 2015. Citado 3 vezes nas páginas 7, 8, and 10.

HASBROUCK, J. Trading costs and returns for us equities: The evidence from daily data. 2005. Citado na página 3.

JONDEAU, E.; ROCKINGER, M. Gram–charlier densities. *Journal of Economic Dynamics and Control*, Elsevier, v. 25, n. 10, p. 1457–1483, 2001. Citado na página 5.

KALMAN, R. E. et al. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, v. 82, n. 1, p. 35–45, 1960. Citado na página 7.

KERCHEVAL, A. N.; ZHANG, Y. Modelling high-frequency limit order book dynamics with support vector machines. *Quantitative Finance*, Taylor & Francis, v. 15, n. 8, p. 1315–1329, 2015. Citado 2 vezes nas páginas 7 and 8.

KINDO, B.; WANG, H.; PENA, E. Mbact-multiclass bayesian additive classification trees. *stat*, Citeseer, 2013. Citado na página 17.

KINDO, B. P.; WANG, H.; PEÑA, E. A. Multinomial probit bayesian additive regression trees. *Stat*, Wiley Online Library, v. 5, n. 1, p. 119–131, 2016. Citado 4 vezes nas páginas 8, 17, 18, and 19.

LEHMANN, E. L.; CASELLA, G. *Theory of point estimation*. [S.l.]: Springer Science & Business Media, 2006. Citado na página 3.

LIU, L. Y.; PATTON, A. J.; SHEPPARD, K. Does anything beat 5-minute rv? a comparison of realized measures across multiple asset classes. *Journal of Econometrics*, Elsevier, v. 187, n. 1, p. 293–311, 2015. Citado na página 3.

MITCHELL, T. M. Machine learning. 1997. *Burr Ridge, IL: McGraw Hill*, v. 45, n. 37, p. 870–877, 1997. Citado na página 11.

O'HARA, M. Presidential address: Liquidity and price discovery. *The Journal of Finance*, Wiley Online Library, v. 58, n. 4, p. 1335–1354, 2003. Citado na página 6.

PERLIN, M.; RAMOS, H. Gethfdata: Ar package for downloading and aggregating high frequency trading data from bovespa. 2016. Citado na página 9.

ROSS, S. M. *Applied probability models with optimization applications*. [S.l.]: Courier Corporation, 2013. Citado na página 4.

SCHAPIRE, R. E. *The design and analysis of efficient learning algorithms*. [S.l.], 1991. Citado na página 13.

SCHAPIRE, R. E.; FREUND, Y. *Boosting: Foundations and Algorithms*. [S.l.]: The MIT Press, 2012. ISBN 0262017180, 9780262017183. Citado na página 13.

TSANTEKIDIS, A. et al. Forecasting stock prices from the limit order book using convolutional neural networks. In: IEEE. *Business Informatics (CBI), 2017 IEEE 19th Conference on*. [S.l.], 2017. v. 1, p. 7–12. Citado 3 vezes nas páginas 2, 3, and 7.