Draft - Chapter 1: Meta-labeling

AUTHOR: GUILLERMO CREUS BOTELLA
SUPERVISOR: DANIEL P. PALOMAR

The Hong Kong University of Science and Technology (HKUST)

Polytechnic University of Catalonia - BarcelonaTech (UPC)

Contents

1 Introduction

Labeling is a well-known area in Machine Learning. It consists of gathering a matrix X, known as features, whose rows are the observations. Once a label y is assigned to every observation $X_{i,\cdot}$ the main goal is to give a prediction $\hat{y}_i$. Whenever $y \in I$ s.t. $|I| = 2$, the problem can be referred as a binary classification, which is what this chapter will focus on.

When it comes to Finance, labeling is not as straightforward as a conventional case, i.e. predicting deterministic events; yes/no traffic light in the picture, yes/no happy face, etc. Since one is working with returns, one needs to determine whether a positive (negative) outcome will happen (or not) in a determined time horizon.

The investment literature has tried to label observations using what Marcos López de Prado (MLDP) [1] defines as *The Fixed-Time Horizon Method*:

$$y_i = \begin{cases} -1 & \text{if } r_{t_{i,0},t_{i,0}+h} < \tau \\ 0 & \text{if } |r_{t_{i,0},t_{i,0}+h}| \leq \tau \\ 1 & \text{if } r_{t_{i,0},t_{i,0}+h} > \tau \end{cases} \tag{1}$$

Where $r_{t_{i,0},t_{i,0}+h}$ is the linear return from time $t_{i,0}$ to $t_{i,0} + h$ (h is a time bar that can be days, months, etc.).

The problem with computing labels with a fixed threshold τ is that volatility, σ, changes overtime and should be updated regularly. Apart from that, restricting the model with a fixed h is not optimal at all since more flexible ways can be implemented.

2 Motivation

A portfolio of N securities is defined by the weights, $\mathbf{w} \in \mathbb{R}^N$ it gives to every instrument. In order to minimize the volatility, the Global Minimum Variance portfolio (**GMVP**) is defined as:

$$\begin{aligned} \min_{w} \quad & w^T \Sigma w \\ \text{subject to} \quad & w \geq 0 \\ & \sum_{i=1}^{N} w_i = 1 \end{aligned} \tag{2}$$

Where Σ is the Covariance matrix of returns. If they are linear, then they are defined as: $R_t^i = \frac{p_t^i}{p_{t-1}^i} - 1$.

At this point, the portfolio budget is defined as B and the amount of money in asset i is simply Bw_i. Now, the portfolio returns can be computed as:

$$R_t^P = \frac{\sum_{i=1}^{N} Bw_i \cdot (1 + R_t^i)}{B} - 1 = \sum_{i=1}^{N} (w_i + w_i \cdot R_t^i) - 1 = \mathbf{w} \cdot R_t \tag{3}$$

Note that if Bw_i is the initial wealth of asset i at time $t - 1$, then the final wealth is:

$$Bw_i \cdot \frac{p_t^i}{p_{t-1}^i} = Bw_i \cdot (1 + R_t^i) \tag{4}$$

With all these concepts laid out, it is appropriate to talk about a single time series. In other words, the portfolio indirectly transforms a multivariate time series into a univariate one. Consequently, from now on, only the portfolio time series will be analyzed.

As it can be seen in figure 1, the falls in late 2018 and early 2020 are a huge setback as far as the final wealth is concerned. Going back to section 1, the general idea is to label these periods accordingly so that these drawdowns can be predicted.

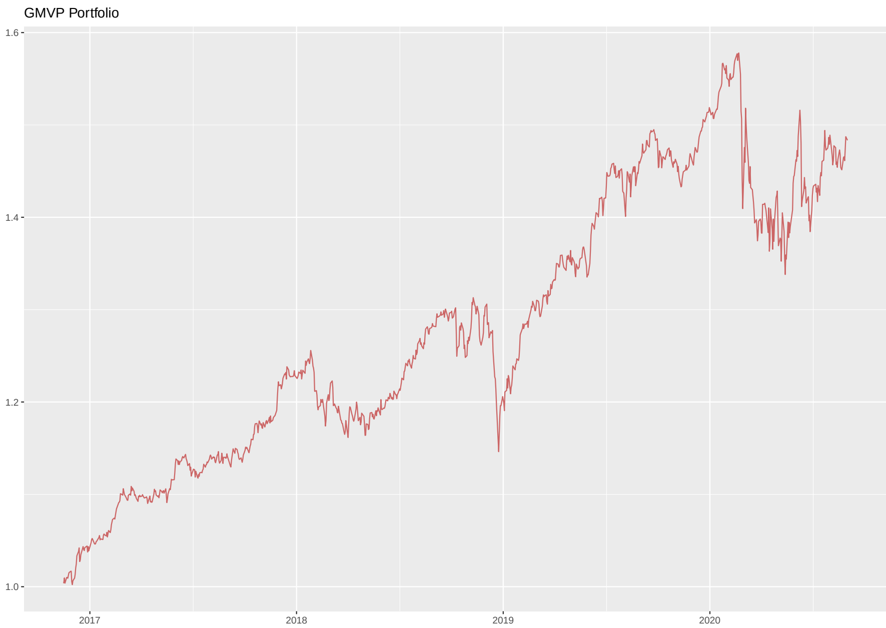

Figure 1: GMVP portfolio prices

3 Toy Project

Before applying Meta-labeling to financial data, an experiment has been designed in order to explain how meta-labeling works with synthetic data. Labels of side (long/short) of investments will be created in a way that a designated set of features are responsible for it. That is, features that have predicting power. The desired output is to predict whether one should open a position or not, and the side of it. This will be done in a sequential manner, with a primary and secondary model.

The primary Model will predict the side of the investment, and the secondary model will decide if the primary model was right. To be specific, the primary model will tell you to open a position (positive in the binary classification context) with a given side, and the secondary model will decide if it is a false or true positive.

Meta-labeling should be used when one wants to achieve higher F1-Scores (harmonic mean of precision and recall), because it lowers the high recall from the primary model while getting a much higher precision, thus boosting the F1-Score.

As for the data division, the primary model will use a set of features different than the secondary model ones. The latter will have some designated features plus the prediction from the primary model That way, with the new information and the prediction, the secondary model will evaluate if the primary model made a correct decision.

3.1 Data

The data for this Toy Project will consist on 1000 observations of the following data points:

- **Features:** $\mathbf{X}_{k,i} \sim N(\mu_i,\ \sigma^2)$ for $i \in \{1,\ \ldots,\ 5\},\ k \in \{1,\ \ldots,\ 1000\}$

- $\omega_k = \textbf{sigmoid}\left(\alpha + \sum_{i=1}^{5} \mathbf{X}_{k,i} \cdot \beta_i + \epsilon_k\right)$ where $\epsilon_k \sim N(0,\ \sigma_\epsilon^2)$ and $\textbf{sigmoid}(z) = \frac{1}{1+e^{-z}}$

  Where: $\alpha = -0.5970$,
  $\boldsymbol{\beta} = [-0.7862,\ 0.7695,\ -1.3740,\ 0.6722,\ -0.4536]$,
  $\boldsymbol{\mu} = [-0.3110,\ -0.1157,\ 0.0316,\ 0.3210,\ -0.5933]$,
  $\sigma = 0.5$ and $\sigma_\epsilon \in \{0,\ 0.1,\ \dots,\ 2.9,\ 3\}$

- **Labels:** The side will be designated using ω_k, which implies that all **5 features** have an effect on the correct side/investment.

$$y_k^{\mathrm{M1}} = \begin{cases} -1 & \text{if } \omega_k < 0.5 \\ \ \ 1 & \text{otherwise} \end{cases}$$

Note that, in order to avoid unbalanced labels, the following constraint has been applied: $0 = \alpha + \boldsymbol{\beta} \cdot \boldsymbol{\mu}$. That way, the expected value of $\alpha + \sum_{i=1}^{5} \mathbf{X}_{k,i} \cdot \beta_i + \epsilon_k$ is 0 ($\textbf{sigmoid}(0) = 0.5$).

By generating data this way, there are 5 explanatory variables that are responsible for the position opening. Lastly, the labels of the secondary model will be a 1 whenever the primary model gave a correct prediction of the side and 0 otherwise. To be specific:

$$y_k^{\mathrm{M2}} = \begin{cases} 1 & \text{if } \hat{y}_k^{\mathrm{M1}} = y^{\mathrm{M1}} = -1 \text{ or } \hat{y}_k^{\mathrm{M1}} = y^{\mathrm{M1}} = 1 \\ 0 & \text{otherwise} \end{cases}$$

3.2 Models

Since this section intends to give a general overview of the way meta- labeling works, the models will use different features in order to simulate relative abundance or scarcity of data. That is:

- M1: $\mathbf{X}_1$ ($N_{\mathrm{M1}} = 1$)
  M2: $\hat{y}^{\mathrm{M1}},\ \mathbf{X}_2,\ \dots,\ \mathbf{X}_5$ ($N_{\mathrm{M2}} = 5$)
- M1: $\mathbf{X}_1,\ \mathbf{X}_2$ ($N_{\mathrm{M1}} = 2$)
  M2: $\hat{y}^{\mathrm{M1}},\ \mathbf{X}_3,\ \mathbf{X}_4,\ \mathbf{X}_5$ ($N_{\mathrm{M2}} = 4$)
- M1: $\mathbf{X}_1,\ \mathbf{X}_2,\ \mathbf{X}_3$ ($N_{\mathrm{M1}} = 3$)
  M2: $\hat{y}^{\mathrm{M1}},\ \mathbf{X}_4,\ \mathbf{X}_5$ ($N_{\mathrm{M2}} = 3$)
- M1: $\mathbf{X}_1,\ \dots,\ \mathbf{X}_4$ ($N_{\mathrm{M1}} = 4$)
  M2: $\hat{y}^{\mathrm{M1}},\ \mathbf{X}_5$ ($N_{\mathrm{M2}} = 2$)

Primary Model (M1): It will use y^{M1} as labels. The underlying model used is a single layer neural network with a sigmoid activation function.

Secondary Model (M2): It will use y^{M2} as labels. The underlying model used is a neural network with a hidden layer (25 units - leaky ReLU) and an output unit with a sigmoid activation function.

Meta Model (M1 + M2): This model is the combination of the previous two models. It will decide to open a position with side ± 1 if the M1 predicts a side ± 1 and the M2 predicts a 1 (i.e. M1 is right). In contrast, if M2 predicts a 0 (M1 is wrong), then the Meta Model will not open a position.

The obvious question regarding the meta model is why cannot one train a single model instead of dividing the data between models. Although the single model will achieve higher precision scores, it will defeat the purpose of meta-labeling. In other words, one of the strengths of meta-labeling is being able to integrate ML into a fundamental/technical analysis approach or a model already up and running. That is, the secondary model will act as an exogenous model and not something that could have been designed from the start.

Lastly, the models will use 80% of data as in-sample and 20% of data as out-of-sample. The former will be further divided into training (80%) and validation (20%) so as to avoid over fitting.

3.3 Results

Before evaluating the results, a confusion matrix of this project, the "positive" and "negative" outcomes should be defined:

- **1:** Open a position
- **0:** Do not open a position

This gives way to:

- **TP:** Opened a position that was profitable.
- **FP:** Opened a losing position.
- **TN:** Did not open a position that was going to be unprofitable.
- **FN:** Took a pass at opening a position that was going to be profitable. wrong.

On the other hand, the metrics that will be used to evaluate the models will be the following:

- **Recall** $= \frac{\text{TP}}{\text{TP+FN}}$
- **Precision** $= \frac{\text{TP}}{\text{TP+FP}}$
- **F1 - Score** $= \frac{2}{\text{recall}^{-1}+\text{precision}^{-1}}$

3.3.1 Example

This subsection will exemplify what meta-labeling does in terms of relabeling false positives as true negatives, confusion matrices and metrics. The hyper parameters chosen are:

- $\sigma_\epsilon = 0.3$
- $N_{\text{M1}} = 2$
- $N_{\text{M2}} = 4$

As for the results, the confusion matrices of the primary model and Meta Model are shown in figures 2 and 3. The primary model (figure 2) only predicts opening positions, i.e., it does not have the ability to pass. The Meta Model (figure 3) corrects more FP than TP.

Table 1: Toy Project Metrics (Test)

Model	F1-Score	Precision	Recall
Primary Model	0.8338	0.7150	1.0000
Meta Model	0.8500	0.7684	0.9510

The metrics obtained are shown in table 1 and figure 4. As it can be implied from the confusion matrices, the recall has gone down but the precision and F1- score have gone up. In particular, the F1-Score has gone up by 2%, from 0.8338 to 0.85. This increase, even though on modest scale, is what meta-labeling was supposed to do.

		Prediction outcome		
		1	**0**	**Total**
Actual value	**1**	TP 143	FN 0	143
	0	FP 57	TN 0	57
	Total	200	0	

Figure 2: Confusion Matrix - Primary Model (Test)

		Prediction outcome		
		1	**0**	**Total**
Actual value	**1**	TP 136	FN 7	143
	0	FP 41	TN 16	57
	Total	177	23	

Figure 3: Confusion Matrix - Meta Model (Test)

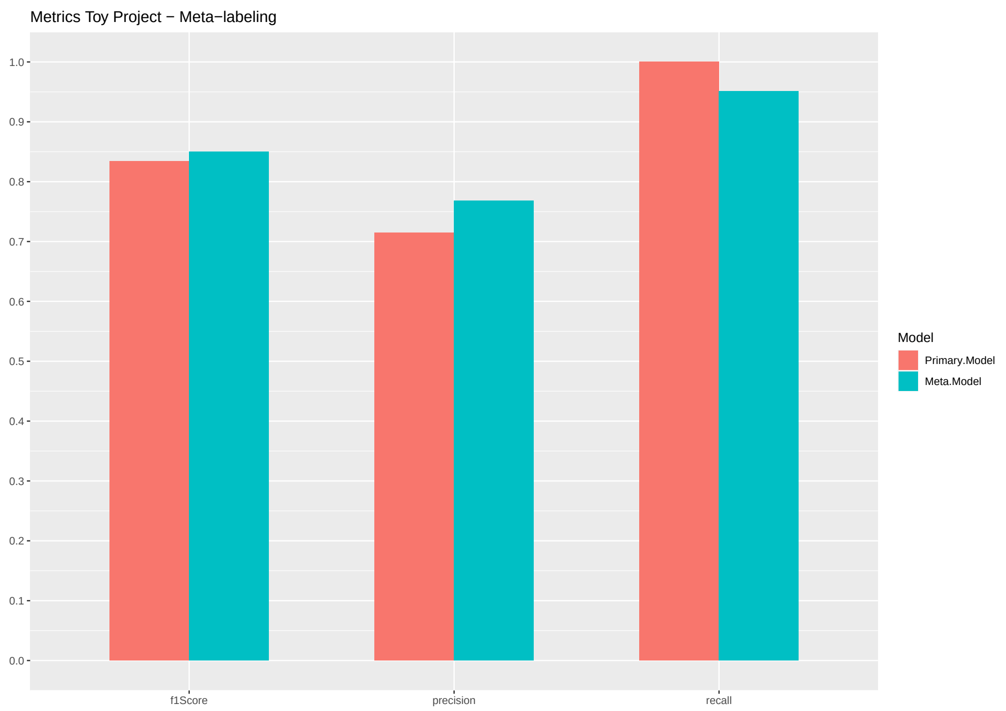

Figure 4: Toy Project - Metrics of example (Test)

3.3.2 Precision

In figures 5 to 8, the precision has been plotted for every σ_ϵ. Whenever $N_{\mathrm{M1}} \geq 3$, the meta model fails to improve the precision. This could be attributed to the primary model having the majority of the information, so the secondary model is not able to improve in the predictions given by the primary model.

On the other hand, if the primary model does not have a full picture of the information ($N_{\mathrm{M1}} \leq 2$), the model is similar to the random model, which has a precision $= 0.5$. That way, the secondary model has more room to improve. To be more specific, if $\sigma_\epsilon \in [0,\ 0.7] \bigcup [2.6,\ 3]$, i.e. the observations have low/high signal-to-noise ratio (high/low σ_ϵ), the meta model performs better.

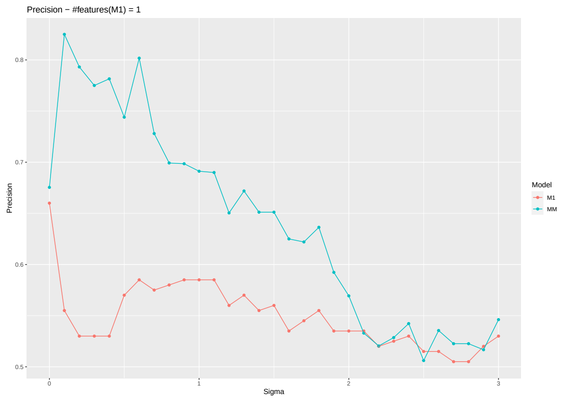

Figure 5: Precision (Test) - $N_{\mathrm{M1}} = 1$

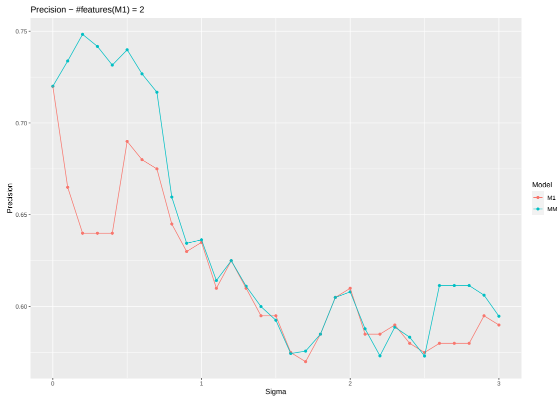

Figure 6: Precision (Test) - $N_{\mathrm{M1}} = 2$

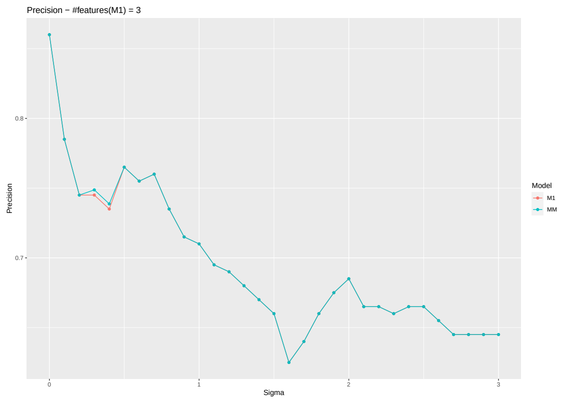

Figure 7: Precision (Test) - $N_{\mathrm{M1}} = 3$

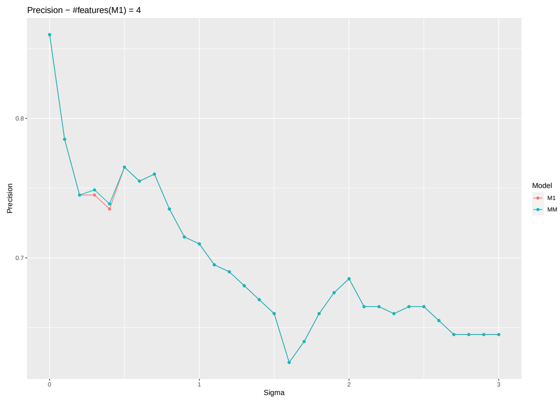

Figure 8: Precision (Test) - $N_{\mathrm{M1}} = 4$

3.3.3 Recall

In figures 9 to 12, the recall has been plotted for every σ_ϵ. Before anything, observe that the primary model, independently of σ_ϵ and N_{M1}, always has a recall of 1. That is because the primary model does not have the ability to pass on a position. In other words, it always spits out a side, so in any event, a position is always opened (it only predicts 1's).

That being said, as in section 3.3.2, note that the meta model does not do a thing whenever $N_{\mathrm{M1}} \geq 3$. Additionally, if $\sigma_\epsilon \in [0,\ 0.7]$, then it is observed that for $N_{\mathrm{M1}} \leq 2$ the recall falls considerably, so one can imply that if σ_ϵ is close to 0, then the recall falls due to the secondary model filtering trades.

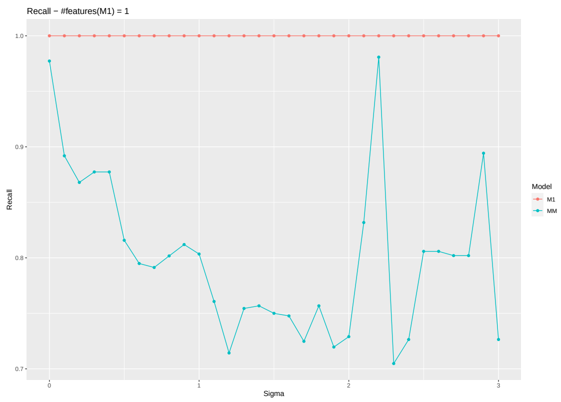

Figure 9: Recall (Test) - $N_{\mathrm{M1}} = 1$

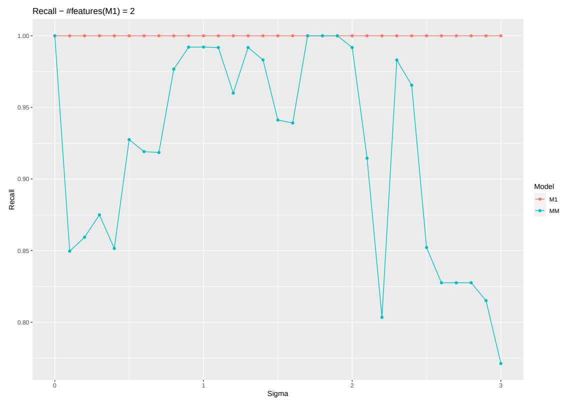

Figure 10: Recall (Test) - $N_{\mathrm{M1}} = 2$

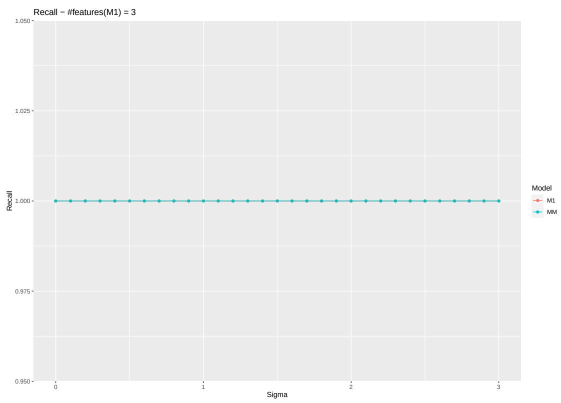

Figure 11: Recall (Test) - $N_{\mathrm{M1}} = 3$

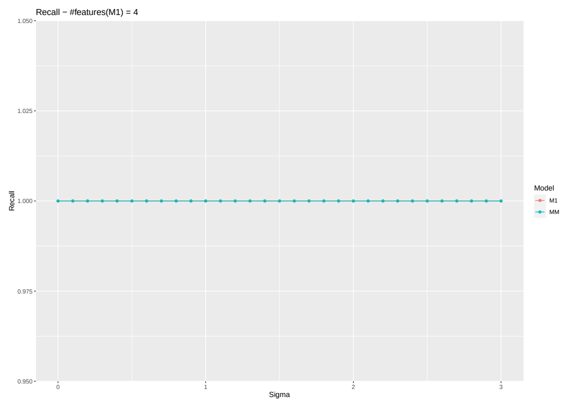

Figure 12: Recall (Test) - $N_{\mathrm{M1}} = 4$

3.3.4 F1-Score

In figures 13 to 16, the F1-Score has been plotted for every σ_ϵ. As it has been seen in 3.3.2 and 3.3.3, if $N_{\mathrm{M1}} \geq 3$, the performance of the primary and meta model is indistinguishable. Furthermore, if $N_{\mathrm{M1}} \leq 2$, the F1-Score slightly improved when σ_ϵ was low ($\sigma_\epsilon \leq 0.6$).

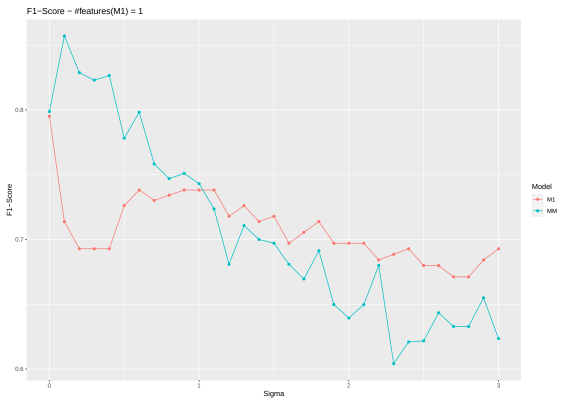

Figure 13: F1-Score (Test) - $N_{\mathrm{M1}} = 1$

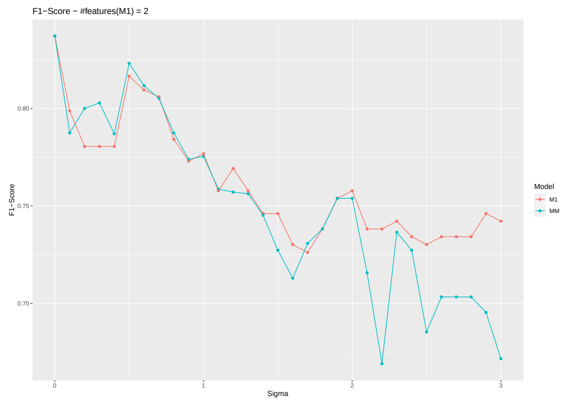

Figure 14: F1-Score (Test) - $N_{\mathrm{M1}} = 2$

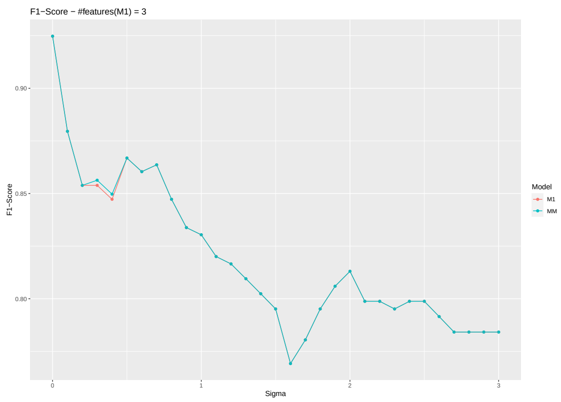

Figure 15: F1-Score (Test) - $N_{\mathrm{M1}} = 3$

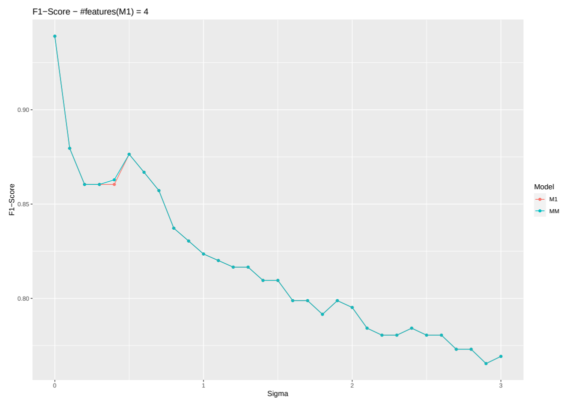

Figure 16: F1-Score (Test) - $N_{\mathrm{M1}} = 4$

3.4 Conclusions

In conclusion, the meta-labeling needs a specific setting so as to deliver better results. In fact, two situations have been identified:

1. M1 performs poorly (low precision) and/or has few features with predicting power (in this case, low N_{M1}).

2. M1 has the majority of the features with high predicting power (in this case, high N_{M1}), i.e., it is already a *good* model.

In situation 1, the meta model will perform better, in an F1-Score sense, whenever σ_ϵ is low and N_{M2} is high. To put it another way, if the observations do not have a lot of noise, then the secondary model is able to correct the bad performance of the primary model. Note that since the 5 features are shared, whenever the primary model has few features with predicting power then the secondary model has a lot of predicting power.

In situation 2, M1 is already a decent model (recall = 1 and high precision when σ_ϵ is low). Consequently, it is very difficult to improve its performance. In fact, the F1-Score of the meta model is identical to the one of M1 because the secondary model fails to introduce new information (low N_{M2}).

The next step will be to try this methodology in financial data, which will be more challenging, since the synthetic data was designed to be somewhat predictable, even though it had Gaussian white noise.

4 Data

The universe of stocks considered are the ones that have been part of the S&P500 in the period considered; 01-01-2000 to 09-01-2020. The stocks that have missing values or have gone bankrupt have been dropped out of the dataset.

The portfolio is not a proxy for the S&P500 GMVP since the portfolio considered presents look-ahead bias. That is, if one were to compute day-to-day the GMVP portfolio, one would not obtain the same results since the *losers* (companies that have gone bankrupt) have been removed, and the *winners* (future constituents of the index) have been added. Nonetheless, this chapter has not been designed in order to beat the index. In fact, this chapter has been designed with a focus on delivering better risk-adjusted results, i.e., higher Sharpe Ratios.

The tickers of the stocks considered are shown in tables 8 and 9 (pages 30 and 31 resp.).

4.1 Removal of outliers

In an attempt to reduce the influence of outliers on the model, the data has been cleaned using the R package imputeFin [2]. In particular, the function **impute_AR1_t** has been used with the following parameters:

- **outlier_prob_th** = 0.005 - Threshold of probability of observation to declare an outlier.
- **remove_outliers** = TRUE

In order to apply the function, data has been divided into 10 chunks. Then, log-prices will be fitted an autoregressive model of order 1 where the residuals follow a t-Student distribution.

$$\log(p_t) = \phi_0 + \phi_1 \cdot \log(p_{t-1}) + \epsilon_t$$

The AR(1) model together with **outlier_prob_th** identifies the outliers and imputes values so as to preserve statistical parameters in the time series. In figure 17 one can see the difference between the original log-price time series (GMVP Outliers) and the imputed one (GMVP Imputed).

4.2 Division of Data

The data is divided as following (see figure 18):

In-Sample:

- Train (2001-08-06/2013-10-18): Data set that will be used to train the ML models.
- Validation (2013-10-21/2016-11-04): Data set to assess the performance of the ML model. It will be used to tune the hyper parameters.

Out-of-sample:

- Test (2016-11-07/2020-08-31): In this data set, the model will generate *out-of-sample* predictions and the performance will be close to real since data will be seen for the first time.

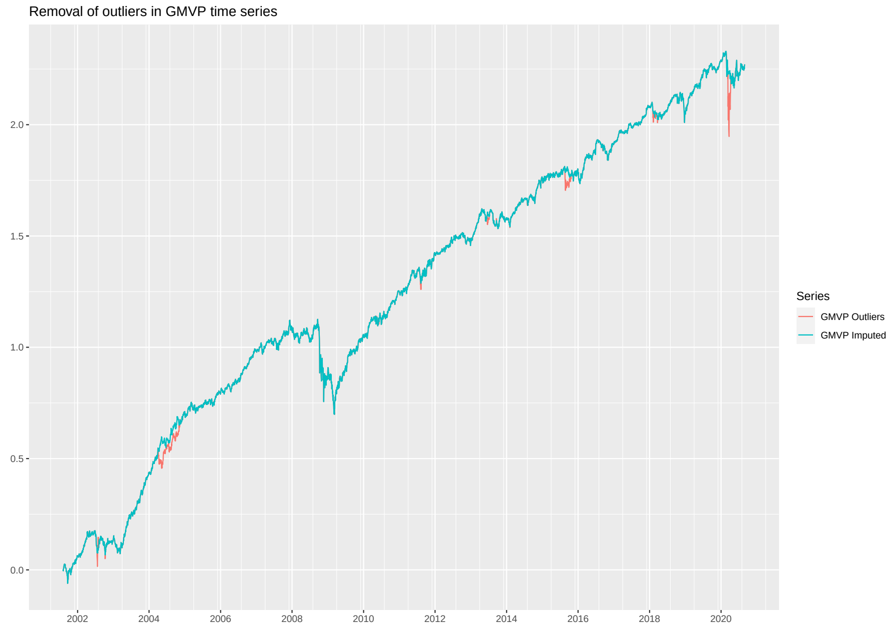

Figure 17: Imputed log-price time series of the GMVP

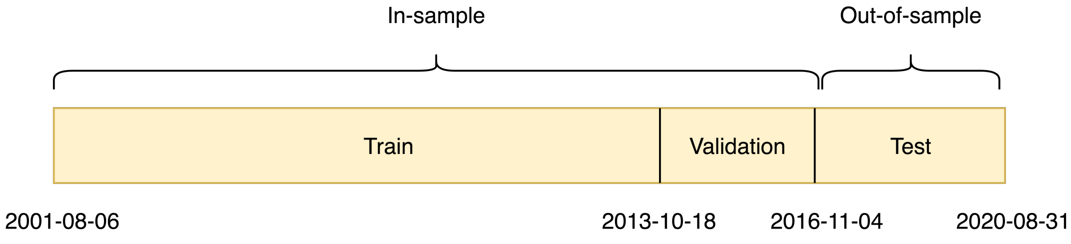

Figure 18: Data Division

5 Labeling (Triple barrier)

In section 1, *The Fixed-Time Horizon Method* was discussed. In an attempt to improve the previous method, MLDP defined the triple barrier method [1]. It consists in:

- **Horizontal barriers:** Dynamic levels that depend on the 10 day rolling volatility.
- **Vertical barrier:** Set as a fixed time horizon. In this case, 10 days.

Then, $t_{i,0}$ is the start day, $t_{i,0} + h$ the vertical barrier and $p_{t_{i,0}}$ the price at time $t_{i,0}$. Also, $p_{t_{i,0}} + \mathrm{pt} \cdot \sigma_{t_{i,0}}$ is the upper horizontal barrier and $p_{t_{i,0}} - \mathrm{sl} \cdot \sigma_{t_{i,0}}$ the lower horizontal barrier, where $\sigma^2_{t_{i,0}} = \frac{\sum_{k=0}^{19}(r_{t_{i,0}-k} - \bar{r}_{[t_{i,0}-19, t_{i,0}]})^2}{20-1}$. Lastly, hyper-parameters **pt** and **sl** both have been set to **2**.

With these variables, the condition to exit a position that was opened at time $t_{i,0}$ is hitting a barrier. Note that exiting is assured since at most the position will be open for 10 days (vertical barrier).

When a barrier is hit ($t_{i,1}$), the return is computed as:

$$r_i = \left[(1 - \mathrm{tc})^2 \cdot \frac{p_{t_{i,1}}}{p_{t_{i,0}}}\right] - 1 \tag{5}$$

Where **tc** is the one-way transaction cost, which has been fixed at **5 bps** (0.05%).

Combining everything, the observations are labeled as:

$$y_i = \begin{cases} 1 & \text{if } r_i > 0 \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

In figure 19, an observation **labeled 1** can be seen. In early January 2002 the barriers start and the price touched the upper horizontal barrier first, hence, obtaining a positive return.

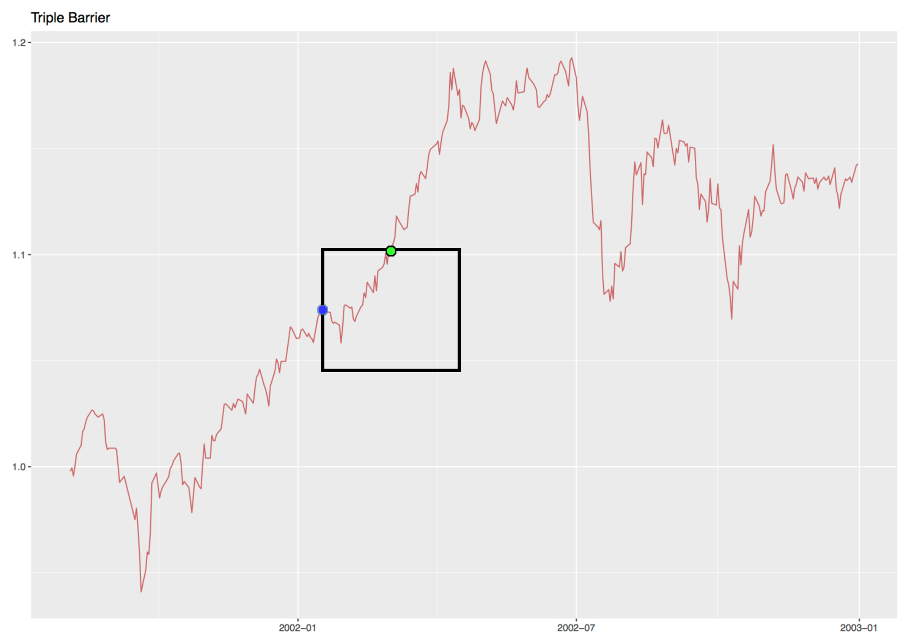

Figure 19: Triple Barrier Labeling (Symmetric barriers)

5.1 Adaptation when the side is known

Noting the side of the trade as $\hat{y}_i^{\text{M1}} \in \{1, -1\}$, that is, either long or short. Accordingly, the method can be modified to include these changes:

$$r_i = \left(\left[(1 - \text{tc})^2 \cdot \frac{p_{t_{i,1}}}{p_{t_{i,0}}} \right] - 1 \right) \cdot \hat{y}_i^{\text{M1}} \tag{7}$$

Combining everything, the observations are labeled as:

$$\hat{y}_i^{\text{M2}} = \begin{cases} 1 & \text{if } r_i > 0 \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

Apart from that, the horizontal barriers should be changed since it has been predicted that the stock will go up/down (long/short):

$$\delta_{+,t_{i,0}} = \begin{cases} \text{pt} \cdot \sigma_{t_{i,0}-1} & \text{if } \hat{y}_i^{\text{M1}} = 1 \\ \min(0.005,\ 0.5 \cdot \text{pt} \cdot \sigma_{t_{i,0}-1}) & \text{otherwise} \end{cases} \tag{9}$$

$$\delta_{-,t_{i,0}} = \begin{cases} \text{sl} \cdot \sigma_{t_{i,0}-1} & \text{if } \hat{y}_i^{\text{M1}} = -1 \\ \min(0.005,\ 0.5 \cdot \text{sl} \cdot \sigma_{t_{i,0}-1}) & \text{otherwise} \end{cases} \tag{10}$$

Therefore, the price will oscillate between $[p_{t_{i,0}} - \delta_{-,t_{i,0}},\ p_{t_{i,0}} + \delta_{+,t_{i,0}}]$. Also, note that $t_{i,0} - 1$ represents the last trading day before $t_{i,0}$ because to compute the barriers of dat $t_{i,0}$ only information **up to that day** should be used.

In figure 20 an example can be seen which presents a side $= -1$ (short). As the prediction says the price should go down, the upper horizontal barrier has been lowered. In this case, since the price has hit the upper horizontal barrier and the side was -1, then the observation will be **labeled 0**.

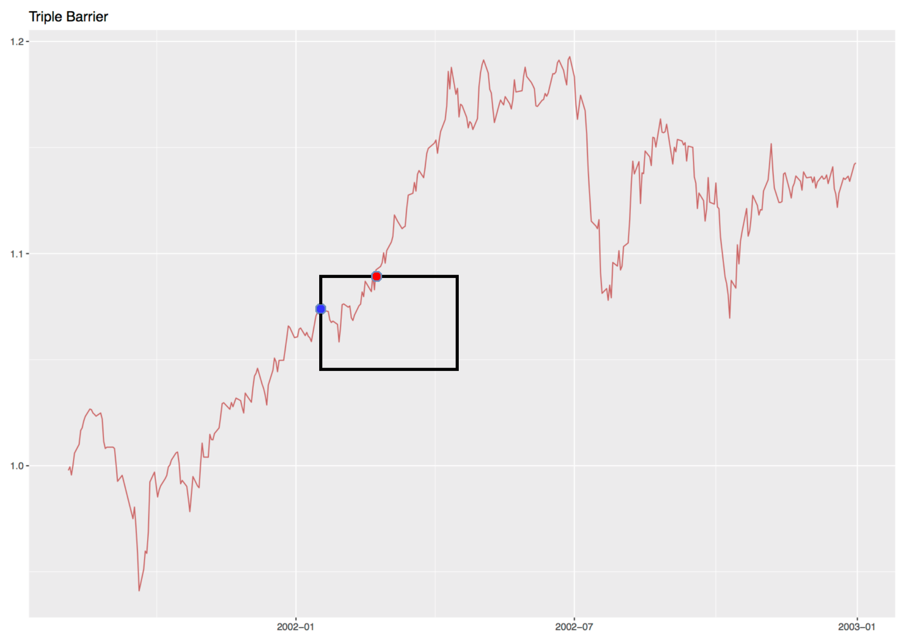

Figure 20: Triple Barrier Labeling - Side $= -1$ (Short)

6 Models

Bet-sizing is a common problem within the practitioner scene. That is, investors often know the side of the position they want to open (long/short or pass) but the size is unknown to them. Hence, an exogenous model that could base its predictions on some features, and the side prediction, comes into play.

The structure will be the following:

- **Primary Model:** A set of features is used to give a prediction on the side of the position. It can be based on fundamental analysis, ML, technical analysis, etc.
- **Secondary Model:** This model uses another set of features that contains the prediction of the primary model. It aims to give a prediction on the size of the position.
- **Meta Model:** It decides the side (primary model) and the size (secondary model). It is merely a combination of both models.

The following example shows how the models work:

$$f_{primary}(X_{\mathbf{M1}}) = 0.89$$
$$f_{secondary}(\{0.89, X_{\mathbf{M2}}\}) = 0.33$$

Table 2: Example Metalabeling

Model	Threshold	Prediction
Primary	0.63	0.89
Secondary	0.41	0.33

Considering that $Im(f_{primary}) \subset [0,1]$ (i.e., no shorting) and $Im(f_{secondary}) \subset [0,1]$ then, one can conclude from the results in table 2 that:

$$0.89 > 0.63 = \text{threshold}_{\text{M1}} \Rightarrow \text{Long position}$$
$$0.33 \leq 0.41 = \text{threshold}_{\text{M2}} \Rightarrow \text{Not enough confidence to open the position}$$

The primary model hinted that a long position should be opened. However, the secondary model evaluated the choice of the primary model and decided not to open the position, since it did not meet the principal criteria: exceeding the threshold. Note that the threshold column in table 2 will be determined in a way that maximizes a carefully chosen performance metric.

Summing all up:

- Develop a primary model (M1) that determines the side of a trade
- **Whenever M1 says that one should open a position**, determine whether M1 gave a correct output or it was wrong (**meta-labeling - labeling a model**)
- With the output of M1, a set of features (they can be the same) and the previous labels develop a secondary model (M2)
- If M1 tells you to trade and M2's prediction surpasses the threshold of M2, open a position with the side given from M1.

7 Binary classification problem

Now that the way observations will be labeled has been explained, and the models involved have been presented, it is relevant to address the binary classification problem that this chapter attempts to solve.

The events that one wants to predict are the entries to profitable investment opportunities, i.e.,:

- $1 =$ **Open a position**
- $0 =$ **Do not open a position**

Having laid out the objective of the problem, the following variables will be defined:

- **True Positive** (TP) $\hat{y}_i^{\text{M2}} = 1 = y_i^{\text{M2}}$
- **False Positive** (FP) $\hat{y}_i^{\text{M2}} = 1 \neq y_i^{\text{M2}}$
- **True Negative** (TN) $\hat{y}_i^{\text{M2}} = 0 = y_i^{\text{M2}}$
- **False Negative** (FN) $\hat{y}_i^{\text{M2}} = 0 \neq y_i^{\text{M2}}$
- **Recall** $= \frac{\text{TP}}{\text{TP+FN}}$
- **Precision** $= \frac{\text{TP}}{\text{TP+FP}}$
- **F1 - Score** $= \frac{2}{\text{recall}^{-1} + \text{precision}^{-1}}$

Where $\hat{y}^{\text{M2}}$ are the predictions and y^{M2} are the true labels of M2.

The idea behind meta-labeling lays in dividing the classification task into two parts. First, a model will act as an initial filter that has high recall, i.e., it catches most of the profitable opportunities, some being false positives. The second model, whenever the first has predicted that there is a profitable opportunity, will "double-check" the supposed positive and will decide whether it is a positive or a negative. By doing that, the overall precision will improve, getting better F1-scores (since recall was initially high).

8 Primary Model

The main role of the primary model is to find investment opportunities and output the side (long/short). In a sense, it could be thought as a meta model where the secondary model always tell you to open a position.

8.1 MA based Primary Model

This primary model uses a Moving Average (MA) crossover strategy with a 20 day look-back period to output the side of the investment. The day of entry will be determined with the CUSUM filter. It can all be summed up in the following equations:

$$\text{MA}_t = \frac{\sum_{i=0}^{19} p_{t-i}}{20} \tag{11}$$

Whit the MA computed, the predicted side of the position follows:

$$\hat{y}_{t+1}^{\text{M1}} = \begin{cases} 1 & \text{if } \text{MA}_t < p_t \\ -1 & \text{if } \text{MA}_t \geq p_t \end{cases} \tag{12}$$

Note that the side generated is of the following trading day $(t+1)$ because the information (MA_t) is received at the end of day t and one can invest with those insights the next trading day.

On the other hand, the true labels are:

$$y_i^{\mathrm{M1}} = \begin{cases} 1 & \text{if } \tilde{r}_i = \frac{p_{t_{i,1}}}{p_{t_{i,0}}} - 1 > 0 \\ -1 & \text{otherwise} \end{cases} \tag{13}$$

The entry points will be determined by structural breaks identified by the symmetric CUSUM (cumulative sum) filter, defined by:

$$\mathrm{CUSUM}_{+,\ t} = \max\left(0,\ \mathrm{CUSUM}_{+,\ t-1} + \log\left(\frac{p_t}{p_{t-1}}\right)\right)$$
$$\mathrm{CUSUM}_{-,\ t} = \min\left(0,\ \mathrm{CUSUM}_{-,\ t-1} + \log\left(\frac{p_t}{p_{t-1}}\right)\right)$$

With the boundary conditions $\mathrm{CUSUM}_{+,\ 0} = \mathrm{CUSUM}_{-,\ 0} = 0$.

The filters **are reset to 0 whenever** $\mathrm{CUSUM}_{+,\ t} > \hat{\sigma}_t$ or $\mathrm{CUSUM}_{-,\ t} < -\hat{\sigma}_t$, where $\hat{\sigma}_t$ is the volatility of logarithmic returns using a 20 day window. Whenever this reset happens, it will be said that **the filters have been activated.**

Hence at this point, at day $t+1$ one should enter if either of the CUSUM filters are activated at day t and there are no positions open. The side is computed using the formula shown in equation 12.

Finally, to exit the position, one needs to compute the vertical and horizontal barriers as it was explained in subsection 5.1. On the occasion that the price time series moves past a barrier, one should proceed to close the position.

8.2 ML based Primary Model

This model is slightly different that the one based on MA. It aims to train a neural network with a hidden layer (20 fully connected units - Leaky ReLU) and an output layer (Sigmoid). It will be in charge of predicting the side (1 = long, -1 = short).

Firstly, let's present the features that the neural network will use. Note that the features of day **t+1** use information **only** from previous days, not the current day.

- $\mathrm{F1}_{t+1} \equiv \log\left(\frac{\mathrm{MA}_t}{p_t}\right)$
- $\mathrm{F2}_{t+1} \equiv \frac{\mathrm{CUSUM}_{+,\ t}}{\sigma_t}$
- $\mathrm{F3}_{t+1} \equiv \frac{\mathrm{CUSUM}_{-,\ t}}{\sigma_t}$
- $\mathrm{F4}_{t+1} \equiv \mathbf{Reset}_{\mathrm{CUSUM}_{+},t}$: Binary variable that indicates if the positive CUSUM filter became activated at time t.
- $\mathrm{F5}_{t+1} \equiv \mathbf{Reset}_{\mathrm{CUSUM}_{-},t}$: Binary variable that indicates if the negative CUSUM filter became activated at time t.
- $\mathrm{F6}_{t+1} \equiv \mathrm{EWMSD}_t$: Exponentially weighted moving standard deviation of linear returns.

Where $\sigma_t^2 = \frac{\sum_{i=0}^{19}(r_{t-i} - \bar{r}_{t-19,t})^2}{20-1}$.

In this model one does not depend on the CUSUM filter to determine the date to open a position. The neural network, with the features given, will decide the side and the position will be opened. That being said, with the intention of matching features and labels, the latter have been defined as:

$$y_i^{\mathrm{M1}} = \begin{cases} 1 & \text{if } \tilde{r}_i > 0 \\ -1 & \text{otherwise} \end{cases} \tag{14}$$

Where $\tilde{r}_i = \frac{p_{t_{i,1}}}{p_{t_{i,0}}} - 1$ is the realized return using the triple barrier method, $t_{i,0}$ the start day and $t_{i,1}$ the day one of the barriers is hit. Also, remember that since side is unknown, the horizontal barriers will be defined with $\delta_{+,t_{i,0}} = \delta_{-,t_{i,0}} = \sigma_{t_{i,0}-1}$.

9 Secondary Model

As it was explained in section 6 the idea behind a secondary model is to train a model capable of learning how to use the primary model, deciding when one should/should not enter. Being more specific, it will filter the positives predicted from the primary model.

Similarly to what it was done in section 8 the models will be adapted to the 2 primary models defined.

9.1 MA based Secondary Model

This model will use a random forest trained with the following **labels:**

$$y_i^{\mathrm{M2}} = \begin{cases} 1 & \text{if } y_i^{\mathrm{M1}} = \hat{y}_i^{\mathrm{M1}} \\ 0 & \text{otherwise} \end{cases}$$

Note that this definition matches the one in subsection 5.1 since:

$$y_i^{\mathrm{M1}} = \hat{y}_i^{\mathrm{M1}} \Rightarrow r_i = \left(\frac{p_{t_{i,1}}}{p_{t_{i,0}}} - 1\right) \cdot \hat{y}_i^{\mathrm{M1}} > 0$$

Additionally, the following features for day $\boldsymbol{t_{i,0}}$ will be used:

- $\log\left(\frac{\mathrm{MA}_{(t_{i,0}-1)}}{p_{(t_{i,0}-1)}}\right)$
- $r_{(t_{i,0}-1)}$: return of the previous day.
- $\underline{r_{(t_{i,0}-1)}} \equiv \left(\prod_{k=0}^{4}(1 + r_{t_{i,0}-k-1})\right) - 1$: Cumulative return of a 5 day window.
- $\hat{y}_i^{\mathrm{M1}}$
- $\mathrm{RSI}_{(t_{i,0}-1),\ 9}$, $\mathrm{RSI}_{(t_{i,0}-1),\ 14}$, $\mathrm{RSI}_{(t_{i,0}-1),\ 25}$: Relative Strength Indicator (RSI) of linear returns using a 9, 14 and 25 days window respectively.
- $\tilde{\sigma}_{(t_{i,0}-1)}$: Exponentially weighted volatility using a 21 day ($\approx$ 1 trading month) window.
- $\sigma_{(t_{i,0}-1),\ 9}$, $\sigma_{(t_{i,0}-1),\ 14}$, $\sigma_{(t_{i,0}),\ 25}$: Volatility of linear returns using a 9, 14 and 25 days window respectively.
- $R_{(t_{i,0}-1),\ 1}$, $R_{(t_{i,0}-1),\ 5}$: Auto-correlation function of returns with 1 and 5 days lag respectively.

The meta model (primary + secondary), together with the threshold $\mathrm{thr}_{\mathrm{M1}}$, will work using the following cases:

1. Check if a position is currently open
2. If no position is open then compute the side ($\hat{y}_i^{\mathrm{M1}}$) and predict the size ($\hat{y}_i^{\mathrm{M2}}$) using the primary and secondary model respectively.
3. Go long if:
   - $\hat{y}_i^{\mathrm{M1}} = 1$ and $\hat{y}^{\mathrm{M2}} > \mathrm{thr}_{\mathrm{M2}}$ - The primary model was right about going long.

   Go short if:
   - $\hat{y}_i^{\mathrm{M1}} = -1$ and $\hat{y}^{\mathrm{M2}} > \mathrm{thr}_{\mathrm{M2}}$ - The primary model was right about going short.

Otherwise, the secondary model infers that there is not enough confidence to open a position and one should return to point 1 using data from the next possible entry point.

4. Open a position and exit using the triple barrier method.
5. Return to point 1 using data from the next possible entry point.

9.2 ML based Secondary Model

This model will use a neural network with a hidden layer (25 fully connected units - Leaky ReLU) and an output layer (Sigmoid). The labeling technique is the same as the one found in 9.1:

$$y_i^{\mathrm{M2}} = \begin{cases} 1 & \text{if } y_i^{\mathrm{M1}} = \hat{y}_i^{\mathrm{M1}} \\ 0 & \text{otherwise} \end{cases}$$

The features for day $\boldsymbol{t_{i,0}}$ are the same as the ones found in the MA based Secondary Model (9.1):

$\hat{y}_i^{\mathrm{M1}}$, $\log\left(\frac{\mathrm{MA}_{(t_{i,0}-1)}}{p_{(t_{i,0}-1)}}\right)$, $r_{(t_{i,0}-1)}$, $\underline{r_{(t_{i,0}-1)}}$, $\hat{y}_i^{\mathrm{M1}}$, $\mathrm{RSI}_{(t_{i,0}-1),\ 9}$, $\mathrm{RSI}_{(t_{i,0}-1),\ 14}$, $\mathrm{RSI}_{(t_{i,0}-1),\ 25}$, $\tilde{\sigma}_{(t_{i,0}-1)}$, $\sigma_{(t_{i,0}-1),\ 9}$, $\sigma_{(t_{i,0}-1),\ 14}$, $\sigma_{(t_{i,0}-1),\ 25}$, $R_{(t_{i,0}-1),\ 1}$ and $R_{(t_{i,0}-1),\ 5}$

The meta model (primary + secondary) will follow the same procedure from subsection 9.2 to open/close a position.

10 Hyper-parameter tuning via cross-validation

10.1 Cross-Validation MA based Models

As it has been seen in subsections 8.1, 8.2, 9.1 and 9.2, thresholds are important parameters when deciding whether to open a position or not. Consequently, attention should be paid so as to achieve the best performance possible. This is where the Sharpe Ratio comes into play.

Suppose one has obtained a daily time series (r_t) of returns from a model. Then one can define the annualized Sharpe Ratio:

$$\mathrm{SR} = \frac{\mathrm{mean}(\hat{r}_t)}{\mathrm{sd}(\hat{r}_t)} \cdot \sqrt{\frac{\#\text{oportunities}}{\#\text{years}}} \tag{15}$$

Where $\hat{r}_t$ is the time series without the zero returns (days where the model has decided not to enter), #opportunities is the number of observations of the time series $\hat{r}_t$ and #years are the number of years elapsed from the first to the last observation of the time series $\hat{r}_t$.

It is important to point out that a Buy & Hold strategy (buying the portfolio and only re balancing periodically) presents the following properties:

1. $\hat{r}_t = r_t$ because the strategy keeps always a position open. Hence, there are 0 days with zero returns.
2. $\frac{\#\text{opportunities}}{\#\text{years}} = 252$ (Trading days in a year) for the same reason as in point 1.

Having defined all these variables, the idea behind having a separate data set that has not been explicitly trained on, is to determine the threshold that performs best. More accurately, the threshold that obtains the best Sharpe Ratio, given that it has enough observations to assure that the Sharpe Ratio is not misleading.

The criteria that has been used to determine whether there are enough observations is the following:

- For every threshold compute the number of times the model decides to enter, which will be called number of discrete returns. Note that it is different than the number of days the model has kept a position open.
- Compute the median.
- If the number of discrete returns surpasses half the median, then it is labeled "Enough observations".

The idea behind this filter is to avoid over fitting. By choosing a threshold that has a low number of discrete returns then it is likely that it will not perform as good out-of-sample because there are few observations.

Since the primary model is based on MA crossovers, which are deterministic, there are not hyper parameters to tune. However, the CUSUM filter determined that a position should be opened when $\text{CUSUM}_{+,\ t} > \hat{\sigma}_t$ or $\text{CUSUM}_{-,\ t} < -\hat{\sigma}_t$. This can be adapted by scaling the volatility using $\text{thr}_{\text{CUSUM}}$, i.e., opening a position (and resetting) whenever:

- $\text{CUSUM}_{+,\ t} > \text{thr}_{\text{CUSUM}} \cdot \hat{\sigma}_t$
- $\text{CUSUM}_{-,\ t} < -\text{thr}_{\text{CUSUM}} \cdot \hat{\sigma}_t$

As it can be seen in table 3, the results are far from satisfactory. Negative Sharpe Ratios are misleading since a small standard deviation can throw things off when returns are close to 0 but negative.

Table 3: Results cross-validation Primary Model (MA)

$\text{thr}_{\text{CUSUM}}$	Final wealth	SR Primary Model (Validation)
0.25	0.6533	-1.3585
0.60	0.6449	-1.4021
0.95	0.6566	-1.4137
1.30	0.6290	-1.6683
1.65	0.6758	-1.5189
2.00	0.7419	-1.1885

That is why it has been decided to determine $\text{thr}_{\text{CUSUM}}$ indirectly. This will be done by computing the best Sharpe Ratio (using the meta model) for every $\text{thr}_{\text{CUSUM}}$. Lastly, the threshold that enables the meta model to achieve the highest SR will be chosen. Table 4 summarizes the results and figures 21 through 26 get into detail about the SR obtained for every threshold.

From table 4 one can conclude that $\text{thr}_{\text{CUSUM}} = 0.25$ performs best, since it achieves a SR of 0.4971. Hence, this threshold will be fixed from now on.

Table 4: Results cross-validation Meta Model (MA)

$\text{thr}_{\text{CUSUM}}$	SR Meta Model (Validation)
0.25	0.4971
0.60	-0.1333
0.95	0.4230
1.30	0.0051
1.65	0.2348
2.00	-0.4674

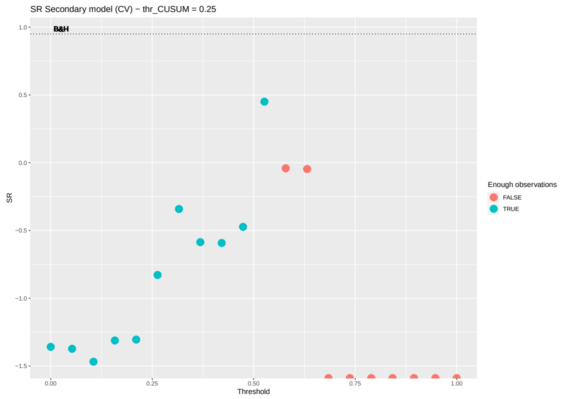

Figure 21: SR MM CV ($\text{thr}_{\text{CUSUM}} = 0.25$)

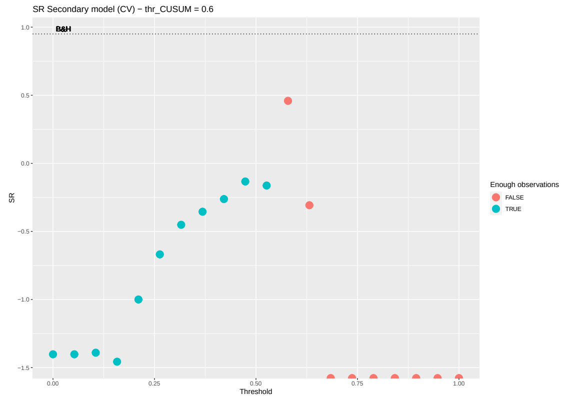

Figure 22: SR MM CV ($\text{thr}_{\text{CUSUM}} = 0.60$)

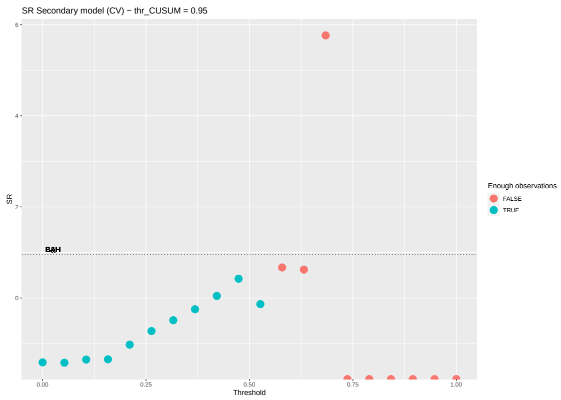

Figure 23: SR MM CV ($\text{thr}_{\text{CUSUM}} = 0.95$)

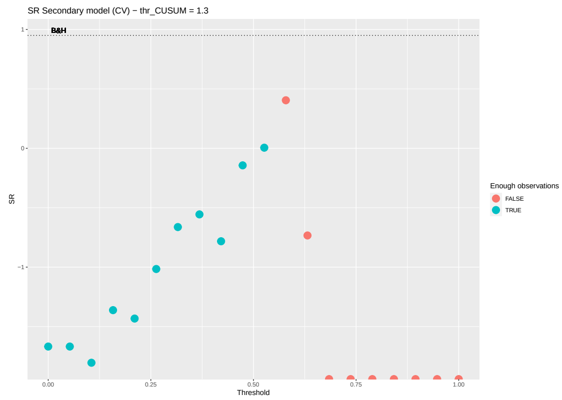

Figure 24: SR MM CV ($\text{thr}_{\text{CUSUM}} = 1.30$)

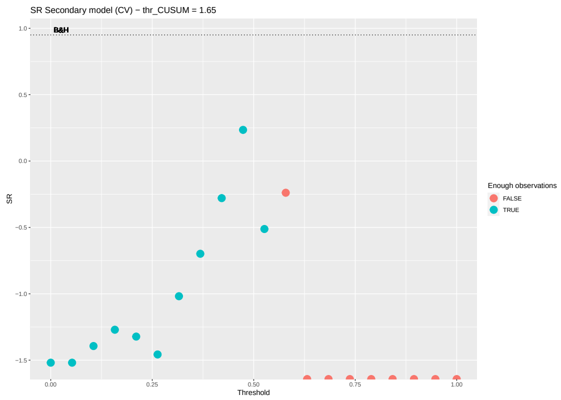

Figure 25: SR MM CV ($\text{thr}_{\text{CUSUM}} = 1.65$)

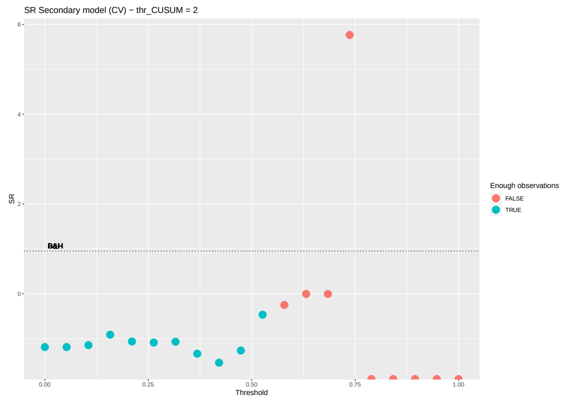

Figure 26: SR MM CV ($\text{thr}_{\text{CUSUM}} = 2.00$)

10.2 Cross-Validation ML based Models

The purpose of cross-validation in the ML based models has been different than the ones based in MA. The validation data set has been used as an independent stopping indicator. That is, the neural networks have been trained on the train data set and whenever the loss function value in the validation data set increases, while the train one decreases, the algorithm stops. This way, over fitting is avoided since the neural network was stopped before it became too good a predictor in a certain data set, without delivering the same results in unseen data sets.

That said, the threshold for both the primary and secondary model has not been tuned. So a $\hat{y}_i$ will be given a positive value whenever it surpasses 0.5, and negative otherwise.

11 Results

In subsections 11.1 and 11.2 a table and some graphs will be shown with the purpose of assessing the out-of-sample performance (Sharpe Ratio & Drawdown) of the models considered, which are the following:

- **Buy & Hold:** Buying the portfolio at the start and keeping the position until the end of the period.
- **Primary Model:** Similar as the previous model but it applies the corresponding side (long/short).
- **Meta Model:** Combination of the primary model and the secondary model. The latter is used to filter the trades from the former.

11.1 MA based Models

Since a huge emphasis has been put to select the threshold that performs best, SR-wise, in the validation set (in-sample), the results of the MA-based model in the validation data set are shown in table 5.

As it is seen in table 5, the meta model does not even achieve satisfactory results (beating the SR of B&H) in the validation model, so one should not expect to see positive results in the Test Dataset. Indeed, in table 6, the results in the test data set show how the meta model is unable to surpass the SR of the B&H.

Besides that, the precision of the primary model is so low (0.51 - validation, 0.58 - test) that the meta model has needed to lower the recall considerably in order to achieve better precision results. That leads to a decrease in F1-Score in both data sets.

For more information, one can refer to figures 27 through 30, where the prices and drawdown of the models have been plotted.

Table 5: Results in validation data set (MA)

Model	Max. Drawdown	SR	Precision	Recall	F1-Score
Buy & Hold	9.17%	0.9479	-	-	-
Primary Model	41.36%	-1.3585	0.5100	1	0.6757
Meta Model	4.97%	0.4504	0.6444	0.1657	0.2636

Table 6: Results in test data set (MA)

Model	Max. Drawdown	SR	Precision	Recall	F1-Score
Buy & Hold	15.20%	0.9682	-	-	-
Primary Model	34.41%	-0.7467	0.5758	1	0.7308
Meta Model	5.02%	0.4887	0.6769	0.1781	0.2821

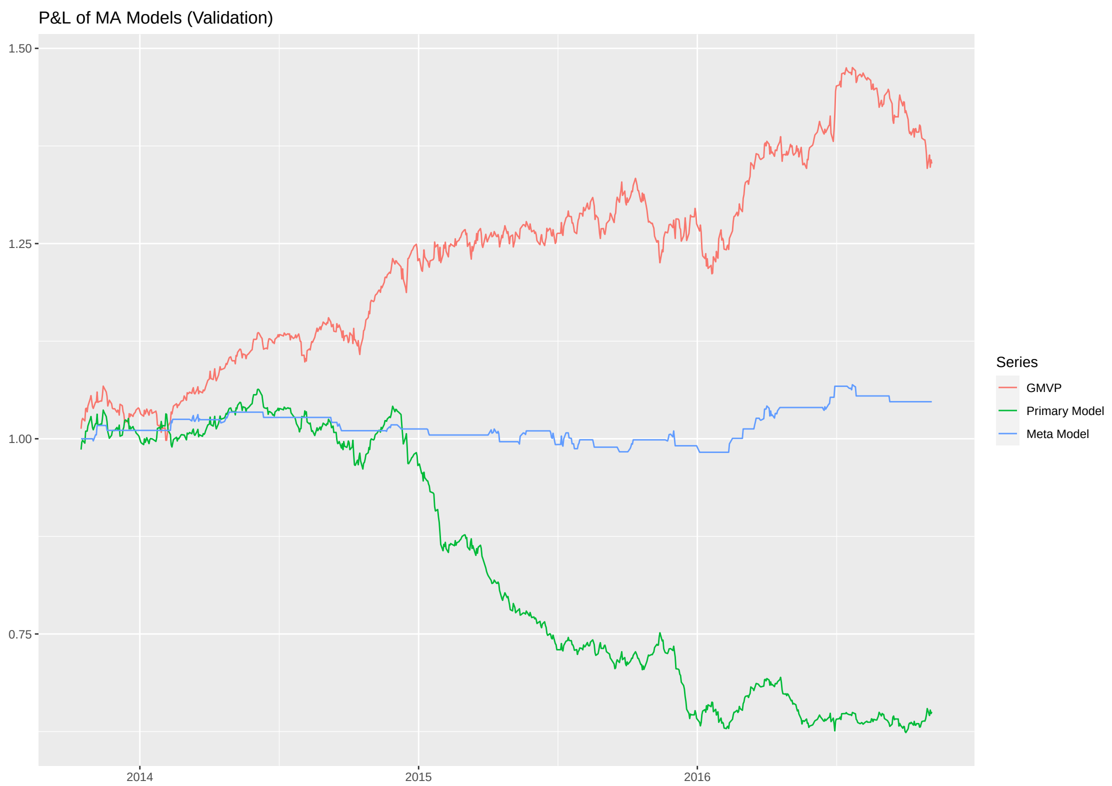

Figure 27: Returns of MA Models in the Validation Dataset

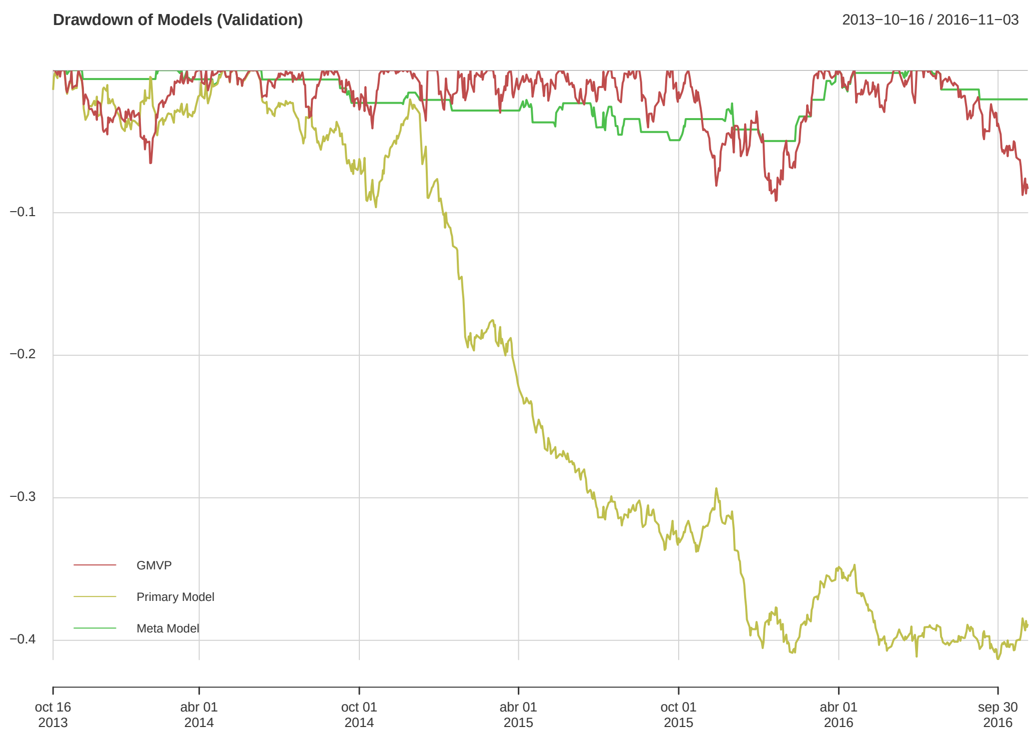

Figure 28: Drawdown of MA Models in the Validation Dataset

Figure 29: Returns of MA Models in the Test Dataset

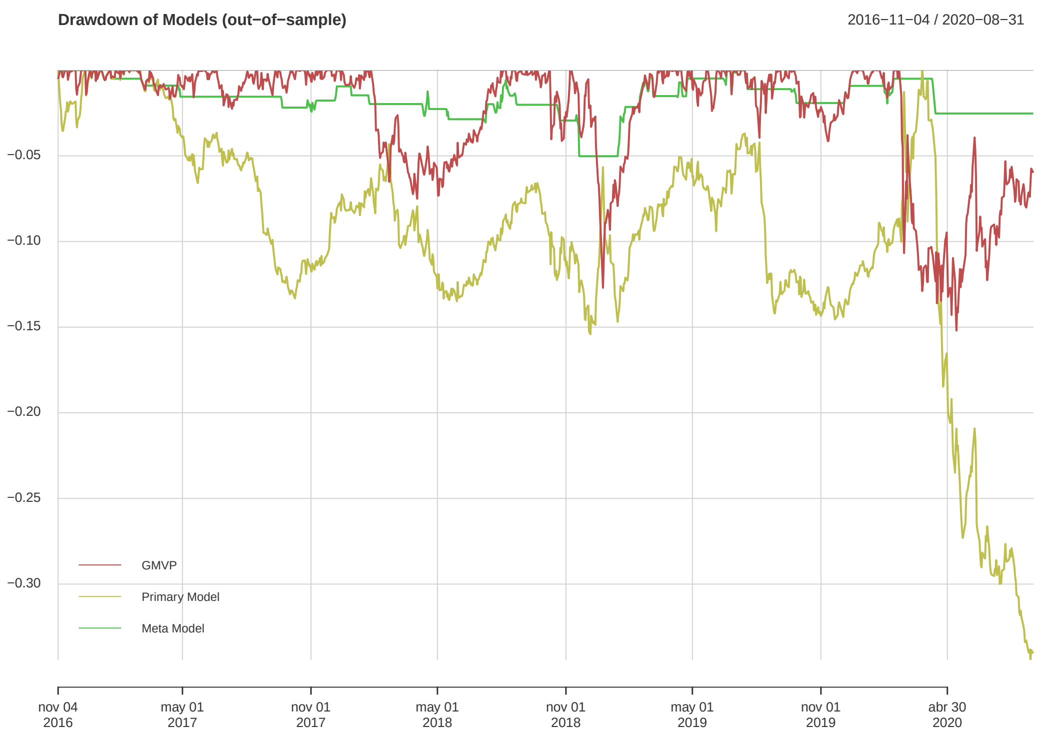

Figure 30: Drawdown of MA Models in the Test Dataset

11.2 ML based Models

Table 7 shows the results of the different models based on ML in the Validation and Test Dataset. Figures 31 and 32 show the prices and drawdown for the different models in the test data set.

The immediate thing to point out is that the performance between the primary model and the meta model is indistinguishable. That means that the secondary model has predicted that the primary model was right all of the times. That is attributed to the lack of predictive power of the features of the secondary model. Consequently, the secondary model has decided to ignore the features and always predict a 1 since it is the majority class (it will result in the best naive metrics). Apart from that, in the primary model a similar thing happens since it always predicts to go long.

All of this is reflected in the metrics, which fail to change from model to model:

Table 7: Results in Test Dataset (ML)

Model	Max. Drawdown	SR	Precision	Recall	F1-Score
Buy & Hold	15.20%	0.9682	-	-	-
Primary Model	16.42%	0.6630	0.6757	1	0.8065
Meta Model	16.42%	0.6630	0.6758	0.9969	0.8055

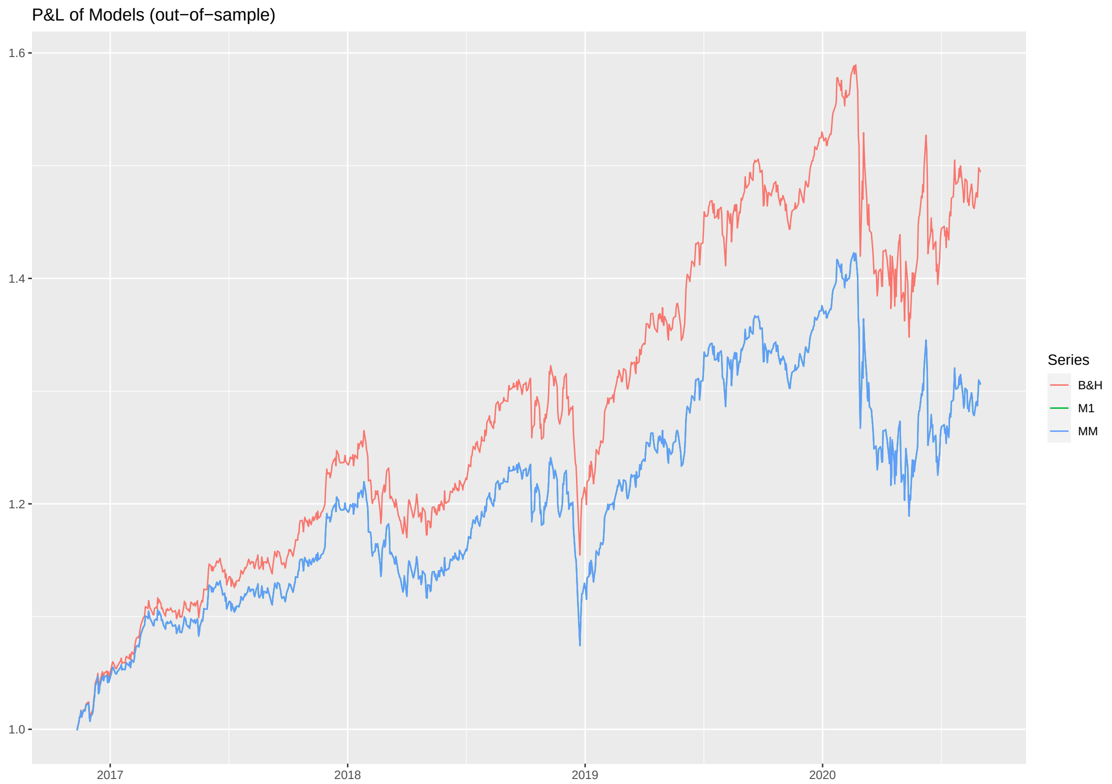

Figure 31: Prices of ML Models in the Test Dataset

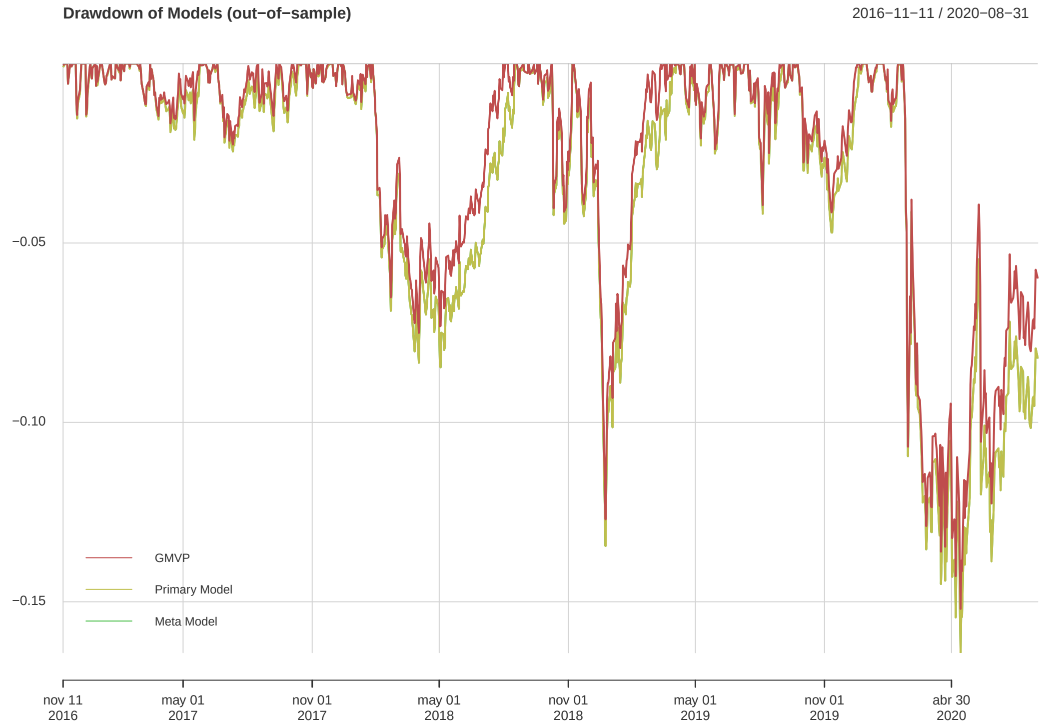

Figure 32: Drawdown of ML Models in the Test Dataset

12 Coin flip correction

As it has been seen in the previous sections, the performance of the primary model has been sub optimal. In an effort to remove this condition from this work, a strategy has been devised towards achieving better (but artificial) primary models.

The idea lays in the concept of a coin flip, $S \sim Be(p)$ $(p = \Pr(S = 1))$. Suppose that every observation is linked to a random variable $S_i \sim Be(p)$. Then, one can define a new feature:

$$F_i = (1 - S_i) \cdot y_i - S_i \cdot y_i = (1 - 2 \cdot S_i) \cdot y_i$$
$$\mathbb{E}[F_i \mid y_i] = (1 - 2p) \cdot y_i$$

where $y_i \in \{-1,\ 1\}$ is the label of the observation in the primary model.

Interpreting p as the probability of swapping a feature, the model that defines $S_i \sim Be(0)$ will not swap features and the primary model will be the "perfect classifier". On the other hand, if $S_i \sim Be(0.5)$ then $\mathbb{E}[F_i \mid y_i] = 0$ and the feature is useless since the feature will alternate (in a random way) between -1 and 1.

These artificial models are based on the ML primary model. That is, apart from the *coin flip* feature, it conserves the same features. The probabilities that define the r.v. S_i are:

$$\mathbf{p} = [0.00, 0.05, 0.10, 0.15, 0.20, 0.25, 0.30, 0.35, 0.40, 0.45, 0.50]$$

12.1 Results

In figures 33, 34 and 35 the P&L plots of the artificial Primary and Meta Model for the Test Dataset are shown.

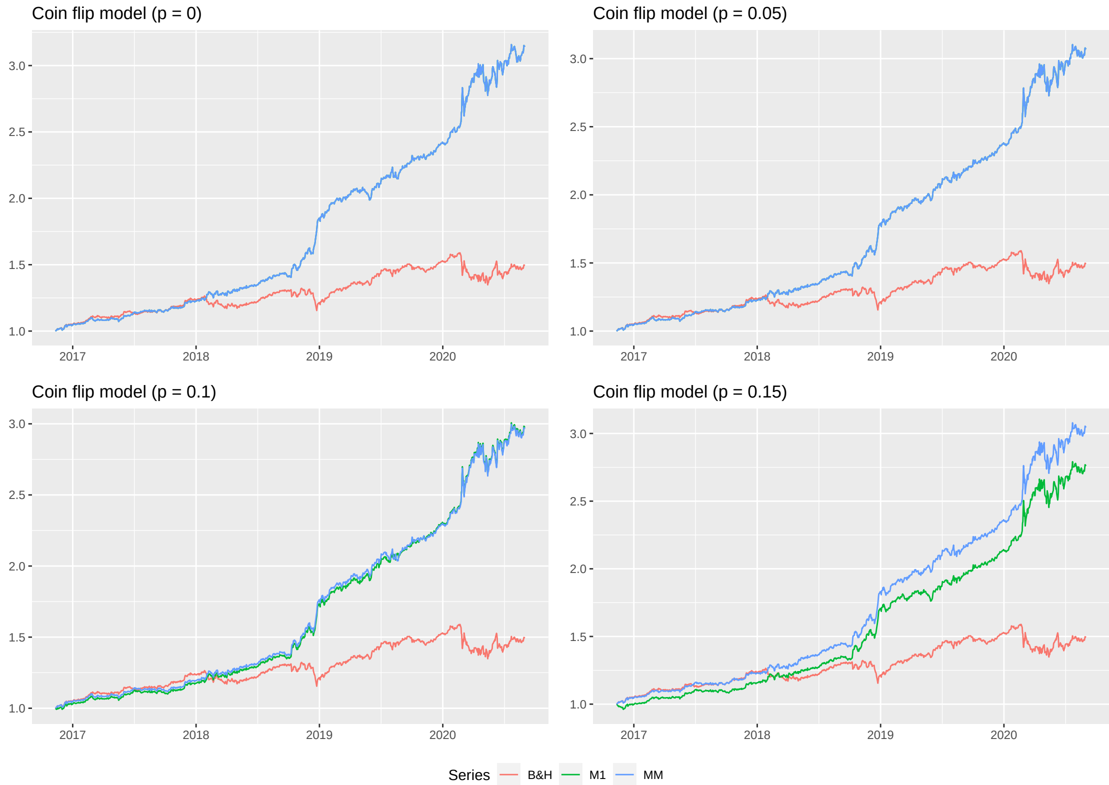

Figure 33: Out-of-sample P&L for $p \in \{0.00, 0.05, 0.10, 0.15\}$

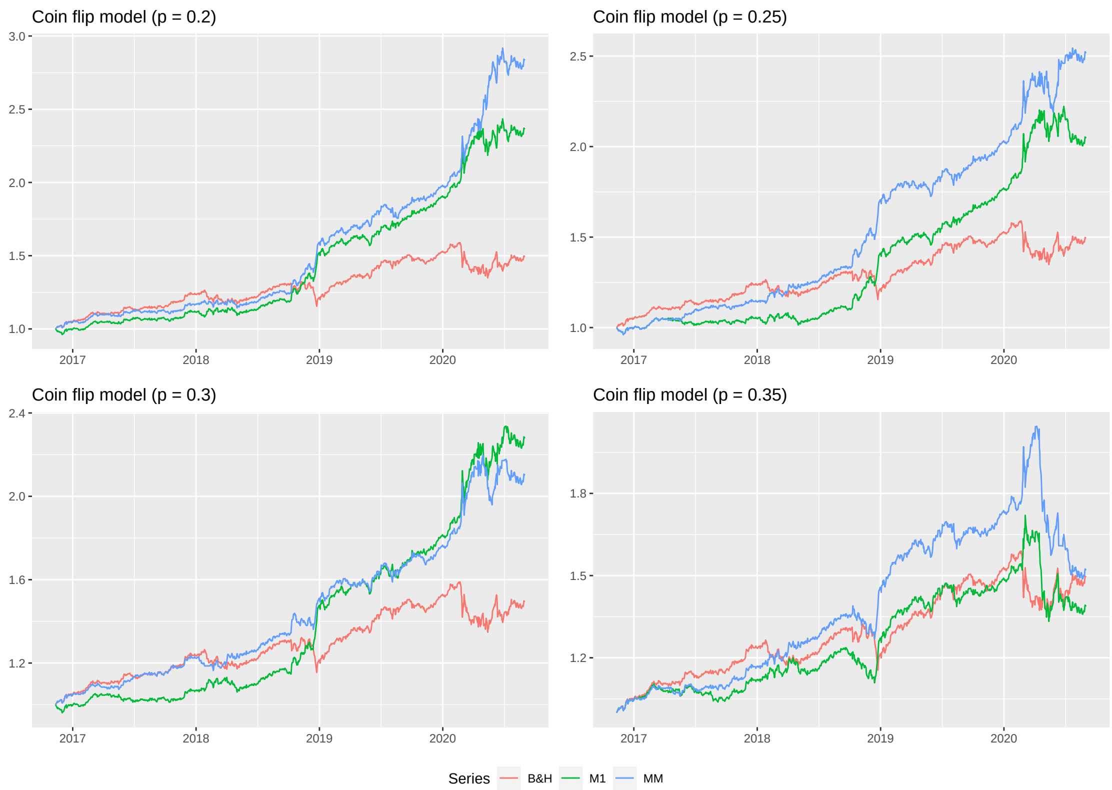

Figure 34: Out-of-sample P&L for $p \in \{0.20, 0.25, 0.30, 0.35\}$

Figure 35: Out-of-sample P&L for $p \in \{0.40, 0.45, 0.50\}$

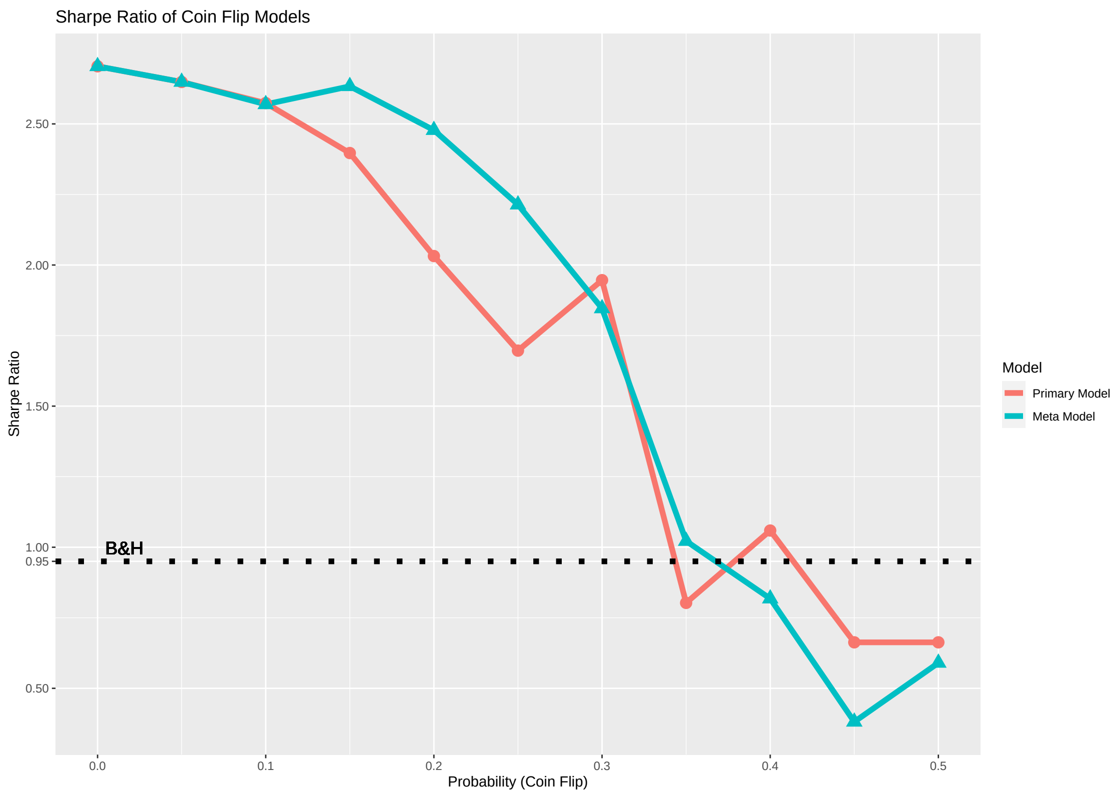

Figure 36: Sharpe Ratio of different coin flip models (*Out-of-sample*)

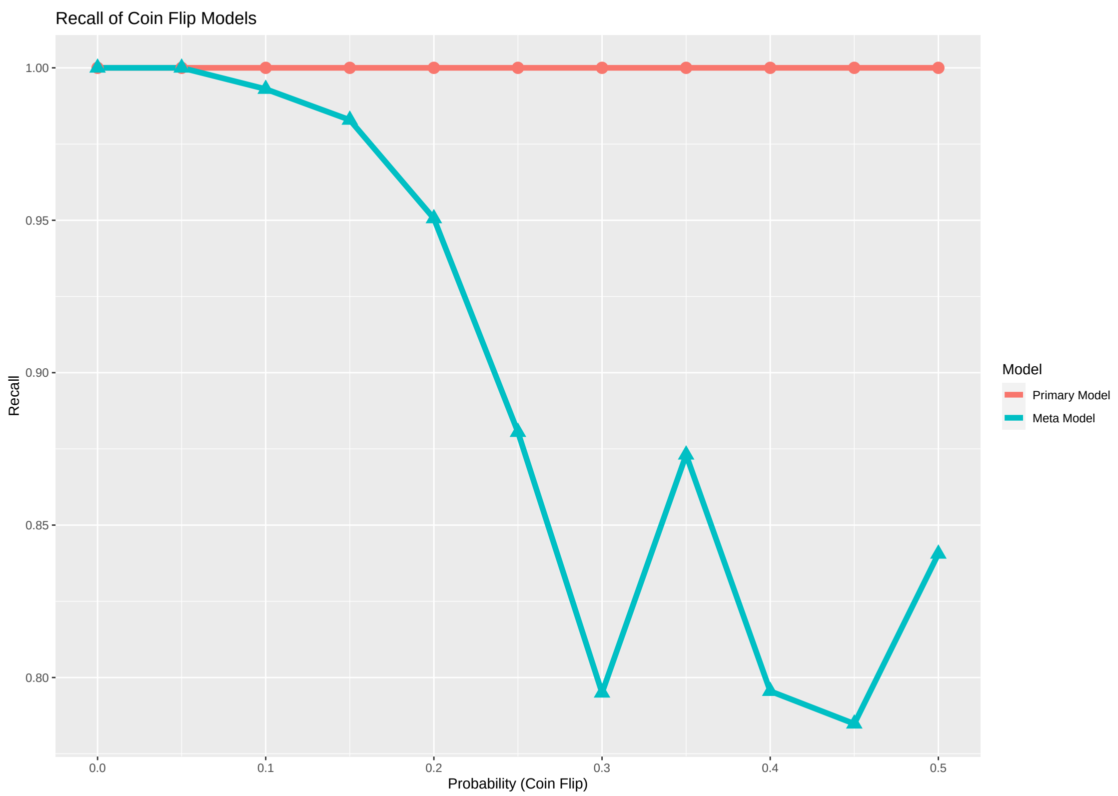

Figure 37: Recall of Coin Flip Models (*Out-of-sample*)

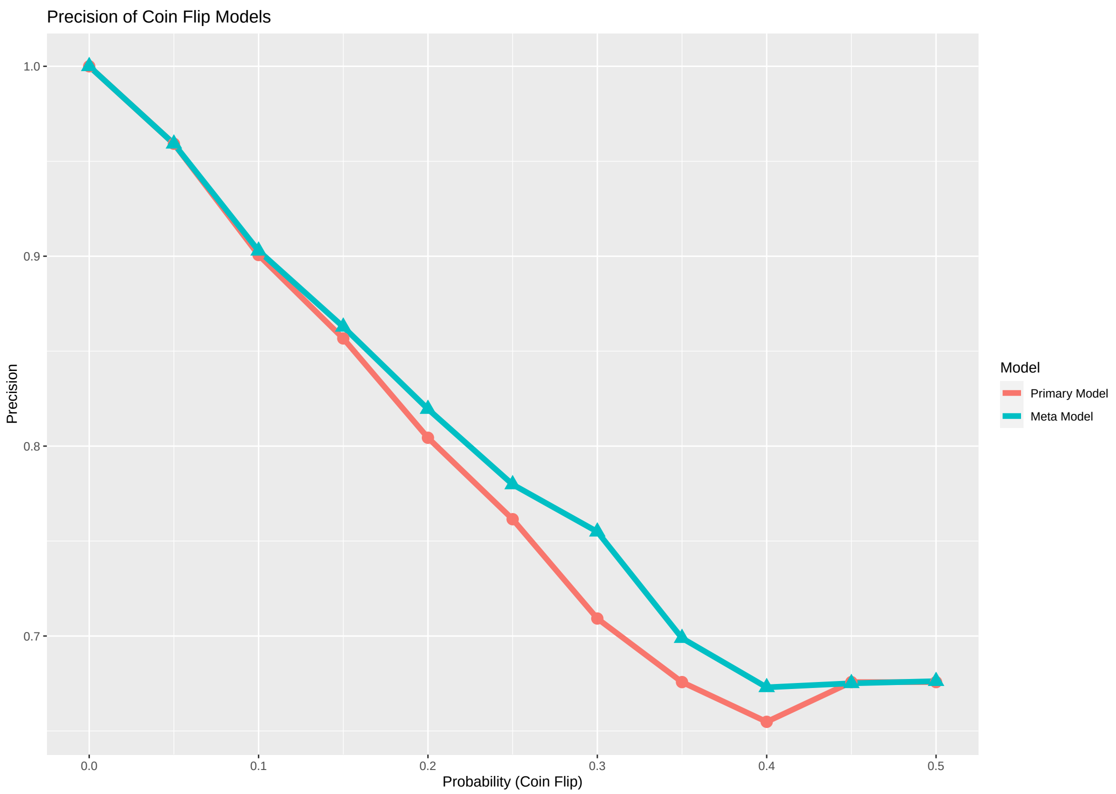

Figure 38: Precision of Coin Flip Models (*Out-of-sample*)

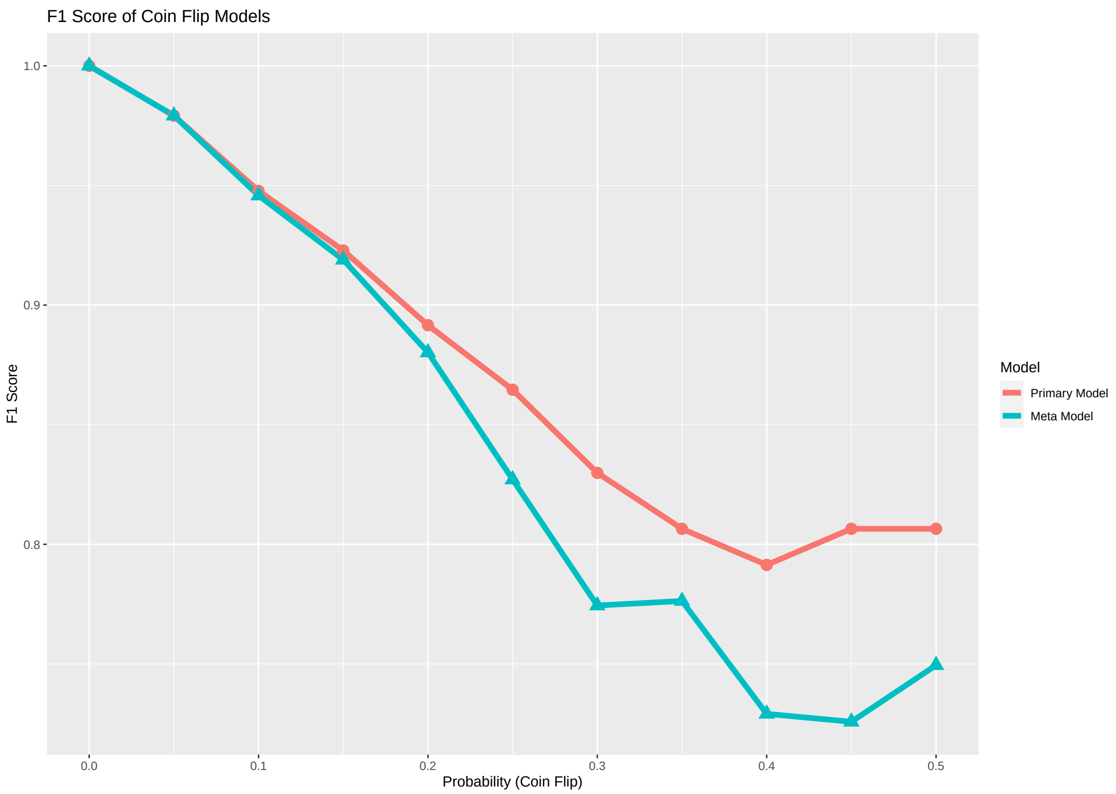

Figure 39: F1 Score of Coin Flip Models (*Out-of-sample*)

13 Conclusions

This chapter has explored the novel labeling technique meta-labeling. Beginning from the toy project, it was obvious that the main benefit of meta-labeling, improving the F1-Score, was hard to capture. In fact, the secondary model needed low levels of noise in order to work, besides having a primary model that was not near perfect (i.e., it made mistakes to could be corrected).

The conditions were obviously not met in the MA and ML based models. In the former, the primary model predictions were near random with precision of nearly 50%. That created a really bad model which was hard for the secondary model to correct. Even though it achieved some success in improving the SR, it was unable to beat a B&H strategy. Apart from that, the ML based models did not have enough data with predictive power, so the results of meta-labeling are inconclusive. In fact, one would dare to say that the models did not "learn" anything which is common in systems where the models can not distinguish the signal from the noise. Note that, this did not happen in the MA based model because even tough it was almost random, it did something. In contrast the ML M1 did not have anything to base its predictions on, so it decided to always go long, since it would achieve the "best" performance for a naive classifier.

However, not everything is unfavorable. By creating better artificial primary models (coin-flip method), the hypothesis in the toy project could be evaluated in a more controlled environment. It can be seen that the SR of the meta model significantly surpasses the one of the primary model whenever the probability of the coin flip is between 0.1 and 0.3. That fits with what was concluded in the toy project: M1 not near perfect and low noise-to-signal ratio.

All in all, meta-labeling applied to financial data is a tricky tool to use. First of all, a good primary model is required, which by itself is difficult to develop. Furthermore, the features of the secondary model need to have good predicting power so as to detect the signal behind the noise, which in financial data is challenging as well .

References

[1] Marcos López de Prado. *Advances in Financial Machine Learning.* Wiley, 2018.

[2] J. Liu and D. P. Palomar. *imputeFin: Imputation of Financial Time Series with Missing Values*, 2019. R package version 0.1.1.

Table 8: Tickers of the GMVP portfolio

MMM	ABT	ABBV	ABMD	ACN	ATVI
ADBE	AMD	AAP	AES	AFL	A
APD	AKAM	ALK	ALB	ARE	ALXN
ALGN	ALLE	LNT	ALL	GOOGL	GOOG
MO	AMZN	AMCR	AEE	AAL	AEP
AXP	AIG	AMT	AWK	AMP	ABC
AME	AMGN	APH	ADI	ANSS	ANTM
AON	AOS	APA	AIV	AAPL	AMAT
APTV	ADM	ANET	AJG	AIZ	T
ATO	ADSK	ADP	AZO	AVB	AVY
BKR	BLL	BAC	BK	BAX	BDX
BRK.B	BBY	BIO	BIIB	BLK	BA
BKNG	BWA	BXP	BSX	BMY	AVGO
BR	BF.B	CHRW	COG	CDNS	CPB
COF	CAH	KMX	CCL	CARR	CTLT
CAT	CBOE	CBRE	CDW	CE	CNC
CNP	CERN	CF	SCHW	CHTR	CVX
CMG	CB	CHD	CI	CINF	CTAS
CSCO	C	CFG	CTXS	CLX	CME
CMS	KO	CTSH	CL	CMCSA	CMA
CAG	CXO	COP	ED	STZ	COO
CPRT	GLW	CTVA	COST	CCI	CSX
CMI	CVS	DHI	DHR	DRI	DVA
DE	DAL	XRAY	DVN	DXCM	FANG
DLR	DFS	DISCA	DISCK	DISH	DG
DLTR	D	DPZ	DOV	DOW	DTE
DUK	DRE	DD	DXC	EMN	ETN
EBAY	ECL	EIX	EW	EA	EMR
ETR	EOG	EFX	EQIX	EQR	ESS
EL	ETSY	EVRG	ES	RE	EXC
EXPE	EXPD	EXR	XOM	FFIV	FB
FAST	FRT	FDX	FIS	FITB	FE
FRC	FISV	FLT	FLIR	FLS	FMC
F	FTNT	FTV	FBHS	FOXA	FOX
BEN	FCX	GPS	GRMN	IT	GD
GE	GIS	GM	GPC	GILD	GL
GPN	GS	GWW	HAL	HBI	HIG
HAS	HCA	PEAK	HSIC	HSY	HES
HPE	HLT	HFC	HOLX	HD	HON
HRL	HST	HPQ	HUM	HBAN	HII
IEX	IDXX	INFO	ITW	ILMN	INCY
IR	INTC	ICE	IBM	IP	IPG
IFF	INTU	ISRG	IVZ	IPGP	IQV
IRM	JKHY	J	JBHT	SJM	JNJ
JCI	JPM	JNPR	KSU	K	KEY
KEYS	KMB	KIM	KMI	KLAC	KHC
KR	LB	LHX	LH	LRCX	LW
LVS	LEG	LDOS	LEN	LLY	LNC
LIN	LYV	LKQ	LMT	L	LOW
LYB	MTB	MRO	MPC	MKTX	MAR
MMC	MLM	MAS	MA	MKC	MXIM
MCD	MCK	MDT	MRK	MET	MTD
MGM	MCHP	MU	MSFT	MAA	MHK

Table 9: Tickers (continued) of the GMVP portfolio

TAP	MDLZ	MNST	MCO	MS	MOS
MSI	MSCI	MYL	NDAQ	NOV	NTAP
NFLX	NWL	NEM	NWSA	NWS	NEE
NLSN	NKE	NI	NSC	NTRS	NOC
NLOK	NCLH	NRG	NUE	NVDA	NVR
ORLY	OXY	ODFL	OMC	OKE	ORCL
OTIS	PCAR	PKG	PH	PAYX	PAYC
PYPL	PNR	PBCT	PEP	PKI	PRGO
PFE	PM	PSX	PNW	PXD	PNC
POOL	PPG	PPL	PFG	PG	PGR
PLD	PRU	PEG	PSA	PHM	PVH
QRVO	PWR	QCOM	DGX	RL	RJF
RTX	O	REG	REGN	RF	RSG
RMD	RHI	ROK	ROL	ROP	ROST
RCL	SPGI	CRM	SBAC	SLB	STX
SEE	SRE	NOW	SHW	SPG	SWKS
SLG	SNA	SO	LUV	SWK	SBUX
STT	STE	SYK	SIVB	SYF	SNPS
SYY	TMUS	TROW	TTWO	TPR	TGT
TEL	FTI	TDY	TFX	TER	TXN
TXT	TMO	TIF	TJX	TSCO	TT
TDG	TRV	TFC	TWTR	TYL	TSN
UDR	ULTA	USB	UAA	UNP	UAL
UNH	UPS	URI	UHS	UNM	VFC
VLO	VAR	VTR	VRSN	VRSK	VZ
VRTX	V	VNO	VMC	WRB	WAB
WMT	WBA	DIS	WM	WAT	WEC
WFC	WELL	WST	WDC	WU	WRK
WY	WHR	WMB	WLTW	WYNN	XEL
XRX	XLNX	XYL	YUM	ZBRA	ZBH
ZION	ZTS				