Meta-Labeling Architecture

Michael Meyer, Jacques Francois Joubert, and Mesias Alfeus

Michael Meyer
is a quantitative researcher at Hudson and Thames Quantitative Research in London, UK.
michael@hudsonthames.org

Jacques Francois Joubert
is the chief executive officer of Hudson and Thames Quantitative Research in London, UK.
jacques@hudsonthames.org

Mesias Alfeus
is a senior lecturer in the Department of Statistics and Actuarial Science at Stellenbosch University in Stellenbosch, South Africa.
mesias@sun.ac.za

KEY FINDINGS

- By separating the side and size of a position, sophisticated strategy architectures can be developed by incorporating meta-labeling.
- Practitioners are presented with several meta-labeling architectures to aid in strategy development.
- The concept of inverse meta-labeling is introduced as an additional tool for building better overall models.

ABSTRACT

Separating the side and size of a position allows for sophisticated strategy structures to be developed. Modeling the size component can be done through a meta-labeling approach. This article establishes several heterogeneous architectures to account for key aspects of meta-labeling. They serve as a guide for practitioners in the model development process, as well as for researchers to further build on these ideas. An architecture can be developed through the lens of feature- and/or strategy-driven approaches. The feature-driven approach exploits the way the information in the data is structured and how the selected models use that information, whereas a strategy-driven approach specifically aims to incorporate unique characteristics of the underlying trading strategy. Furthermore, the concept of inverse meta-labeling is introduced as a technique to improve the quantity and quality of the side forecasts.

In the field of quantum mechanics, the Heisenberg uncertainty principle states that it is impossible to simultaneously determine the exact position and momentum of an electron. One could argue that a similar principle exists in the field of quantitative investment regarding the simultaneous determination of the position side and size. The crux of the argument is that strategies that employ machine learning (ML) models to capture both the side and size of a position to trade introduce unwarranted model complexity, inevitably leading to suboptimal trading decisions.

An ML technique that addresses this issue, which has recently gained momentum among academics (Lopez de Prado 2018a,b; Man and Chan 2021; Joubert 2022) and practitioners alike, is meta-labeling. The basic concept of meta-labeling is to fit a secondary model on top of an existing base primary strategy to improve strategy metrics, such as the Sharpe ratio and maximum drawdown, by filtering out false positives and providing a new mechanism to determine position sizing (Joubert 2022). Furthermore, Lopez de Prado (2018a) posited that, by separating the side and size decisions of a position, sophisticated strategy structures could be developed by incorporating meta-labeling.

An ML architecture defines how the various components interact with each other, as well as the key stages through which the entire model progresses. Neural networks, for example, have well-established architectures that are widely used in practice to solve specific problems. For example, artificial neural networks are used to solve pattern recognition problems (Yegnanarayana 2009), convolutional neural networks for image classification (Gu et al. 2018), and recurrent neural networks such as the long short-term memory architecture for speech recognition (Graves, Abdel-Rahman, and Hinton 2013). We aim to establish architectures for meta-labeling that can serve as a guide for practitioners to adapt and use for the suited purposes, as well as academics looking to expand on the ideas presented here.

The article is organized as follows. First, the theoretical principles established in the previous literature regarding the separation of position side and size, meta-labeling, and ML model architecture are reviewed. Second, various meta-labeling architectures (MLAs) are proposed with the underlying motivation for each construction. Third, the concept of inverse meta-labeling is introduced as a framework to enhance the primary model and overall strategy structures. Lastly, conclusions are drawn concerning the architecture framework, and potential future research questions and directions are suggested.

THEORETICAL FRAMEWORK

Side vs. Size

In his article, "The 10 Reasons Most Machine Learning Funds Fail," Lopez de Prado (2018b) asserted that a common mistake ML funds make is that they use a single model to determine both the side and size of the position to take. He argued that the position side decision is a purely fundamental one that aims to capture the fair value of a security under a specific set of circumstances, whereas the position size decision is one of risk management. He proposed modeling each facet separately, using one model to predict the side and another to predict the size of the position.

Meta-Labeling

The concept of meta-labeling was first introduced by Lopez de Prado (2018a) in his book *Advances in Financial Machine Learning*. He defined meta-labeling as a secondary ML model that learns how to use a primary exogenous model. The secondary model's target variables are the meta labels, which are defined as binary labels with the values 0 or 1 and indicate whether the primary model's forecast was accurate. The secondary model's output can then be interpreted as the probability of a profitable trade.

The final position size can be determined from this output probability, in which the higher the probability, the larger the position size. Meta-labeling only filters out potential false positives and does not produce any new trading signals. Therefore, recall (the number of trades) is sacrificed for a higher degree of precision (proportion correctly identified profitable trades), leading to a higher F1 score and improving the model's effectiveness.

Joubert (2022) distinguished three key advantages of meta-labeling:

1. Improving the information advantage
2. Modeling for false positives
3. Sizing positions according to model output

The information advantage is the benefit gained when the secondary model is better able to utilize the data used in the primary model than the primary model itself. For example, if the primary model only models linear relationships, the secondary model could capture nonlinear and interaction effects if the features exhibit these behaviors. To check whether an information advantage exists, a separate ML algorithm could be built to see whether it outperforms the original primary model. If there is indeed an information advantage, it is also an indication that a better primary model can be built.

The modeling for false positives advantage can be divided into two components. First, model evaluation statistics can be used to evaluate recent model performance. These statistics can be used to filter out false positives when the primary model displays poor performance. Because the primary model is a classification problem that predicts the direction of future price, the usual classification metrics derived from the confusion matrix, such as precision, recall, F1 score, and the area under the receiver operating characteristic (AUROC), can be included.

Furthermore, structural breaks in the error terms can be measured using a cumulative sum control chart (CUSUM) filter (Severo and Gama 2006), a Kalman filter applied to the rolling log loss (Harvey 1990), and the rate of change in other relevant statistics.

Second, market state and regime-related features can be included. Investment and trading strategies often perform differently in various market environments. For example, if a short volatility strategy is deployed under normal market conditions, the conditional probability of profit may be quite high. However, if the strategy is applied during a period of market distress, it could be very low (Bongaerts, Kang, and van Dijk 2020).

We can distinguish between market state and regime-related features. Market state features refer to short- to intermediate-term conditions that can, for example, be determined by the first four moments of its distribution. Other market-related features can also be included, such as the Cboe Volatility Index (VIX) or 30-day exponentially weighted moving average market return.

A regime refers to the fundamental structural characteristics that govern the behavior of asset classes in a certain period. A regime change occurs when there are significant changes in the price behavior in financial markets (Chen and Tsang 2020). Regime-related features that often differ across regimes are the means, volatilities, autocorrelations, and cross covariances of asset returns (Ang and Timmermann 2012). Macroeconomic factors such as the inflation rate, short-term interest rate, the slope of the yield curve, and money supply could also be used for regime detection (Bahloul, Mroua, and Naifar 2017).

Various strategies can be employed to detect regimes. Recent proposals in the academic literature have included directional change (Chen and Tsang 2020) and unsupervised learning with Wasserstein k-means clustering, which classifies market regimes based on the distance of observed points in a metric space (Horvath, Issa, and Muguruza 2021).

Moreover, practitioners in the industry commonly make use of ML regime detection techniques. For example, the hedge fund Two Sigma (Botte and Bao 2021) uses unsupervised learning with Gaussian mixture models to fit various distinct Gaussian distributions to capture regime state data, and the ML-driven investment platform PredictNow.ai (2022) uses supervised ensemble learning with random forests, which relate the market state to values of regime-relevant time series.

The position size can be determined as a function of the output from the secondary model. The output probabilities often first need to be transformed into well-calibrated probabilities that better reflects the true posterior probabilities (Niculescu-Mizil and Caruana 2005). If the output probabilities could be calibrated into a frequentist interpretation, sophisticated techniques such as the risk-constrained Kelly criterion (Busseti, Ryu, and Boyd 2016) might also be applied.

Model Architecture

The ML architecture is a schematic that details the process of and interactions among all the ML components. A visualization of the architecture helps interested parties easily interpret the process and gives them a better idea of how everything fits together. Complex concepts such as neural networks have been made considerably more accessible by simple schematics that illustrate the interactions among different layers and their components.

It is important to note that the architecture and the model are not identical. The architecture describes the overall approach to the problem, whereas a model is one piece of the architecture—a specific instance that is trained on a chosen set of data.

ML architectures often combine several base models to produce one final model or output. These base models are referred to as ensemble members. The different combinations in which these members are assembled are referred to as the ensemble architecture or system.

There are three pillars of any ensemble architecture (Zhang and Ma 2012):

1. The selection of the training data for each ensemble member
2. The criteria for selecting ensemble members to include in the ultimate architecture
3. The way ensemble members are used in combination to produce the final output

Dynamic ensemble selection techniques can be employed to select the classifiers (Cruz, Sabourin, and Cavalcanti 2018). The final decision is often based simply on majority voting or a weighting mechanism. Ensemble methods can be deployed in one of two general contexts:

1. Classifier selection: Classifiers are trained on some local region of the entire feature space and regarded as an expert in that region. Given new input features, a distance metric is used to weigh the ensemble members to determine the final classification decision.
2. Classifier fusion: All ensemble methods are trained over the entire feature space selection, and the model outputs are weighed according to a certain scheme. This reduces the variance of the final model.

Though distinct, MLAs bear a resemblance to ensemble architectures in that they also consist of the three pillars previously mentioned. One of the fundamental differences between meta-labeling and ensemble techniques is that the target variable for any meta-labeling model is always the meta labels, whereas ensemble techniques do not have this restriction. Meta-labeling is also always built on top of an exogenous primary model while ensemble techniques can be used in parallel.

Layering a secondary model on top of the primary model also introduces model diversity. Li, Turkington, and Yazdani (2020) showed that disparate models employ features in different ways; therefore, if model diversity is present, additional performance may be gained. For example, applying linear or logistic regression to a large, disparate set of features to predict a label usually fails because many relationships cannot be captured by a linear model. However, if the primary model is linear and the secondary model nonlinear, these relationships can be modeled for in the final prediction. The next section describes the specific architectures proposed by the authors.

META-LABELING ARCHITECTURES

Joubert (2022) presented a general MLA that is used in three controlled experiments to highlight the main advantages of the technique. This article expands on these ideas by developing novel architectures to specifically target and exploit the information advantage, modeling for false positives, and position sizing components.

The process of developing and building sophisticated strategies would benefit greatly from well-established heterogeneous model architectures that can capture unique characteristics of the feature space and strategy implemented, which would streamline the model development process. This fact, in addition to the benefits of meta-labeling presented by Joubert (2022), emphasizes the need to have different architectures at hand when aiming to develop a profitable trading strategy.

There are two main approaches to the development of MLAs; namely, there are feature-driven and strategy-driven architectures. Feature-driven architectures emphasize the way the information is extracted and exploited from the data using different meta-labeling schemes or arrangements. The primary model is viewed as a black box, and the emphasis is on the input data (information) provided in each layer of the architecture.

Strategy-driven architectures, on the other hand, specifically take into consideration the properties of the underlying primary strategy and aims to apply meta-labeling in a way that best captures and enhances unique characteristics of the strategy. It should be noted that there is considerable overlap between the two approaches, and the approaches simply indicate the thought process underlying the development of the architecture.

Each architecture aims to account for specific aspects of the MLA process. Further systems or architectures could be developed by combining the different architectures, resulting in more advanced strategies. For example, a feature-driven approach could be combined with a strategy-driven approach to capture all necessary aspects.

Primary Model Architecture

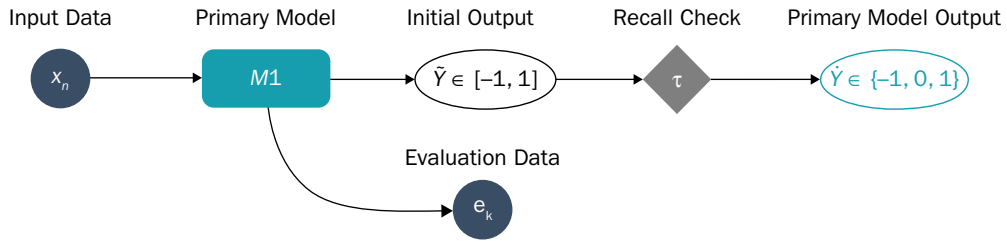
Exhibit 1 presents a simple architecture of a ML primary model. The primary model, however, can be any strategy such as econometric equations, technical trading rules, fundamental analyses, factor-based strategies, or discretionary trades. For simplicity, we chose an ML model for all the architectures, but they can easily be adapted to suit the strategy chosen.

The primary model takes as its inputs n features, which are indicative of a directional move, and aims to predict the side of the trade. The predicted side is given as the labels $\dot{Y} \in \{-1, 0, 1\}$, signalling the position to take, as follows: –1 for a short position, 0 to close any open position and 1 for a long position. The initial output from the ML primary model, $\tilde{Y}$, is between $[-1, 1]$, and the use of a threshold value τ to adjust precision and recall can be implemented as a further step (recall check). Values above τ are denoted by 1, those below τ by –1, and all other values by 0.

Lipton, Elkan, and Narayanaswamy (2014), for example, showed that if classifier outputs are well-calibrated conditional probabilities, then the optimal threshold is half the maximum F1 score. To enable the secondary model to remedy the low precision and produce a better F1 score, it is crucial to fit a primary model with a high recall early in the process. Additionally, the primary model offers k model evaluation statistics e_k, which the secondary model could incorporate as a sign of false positives.

EXHIBIT 1

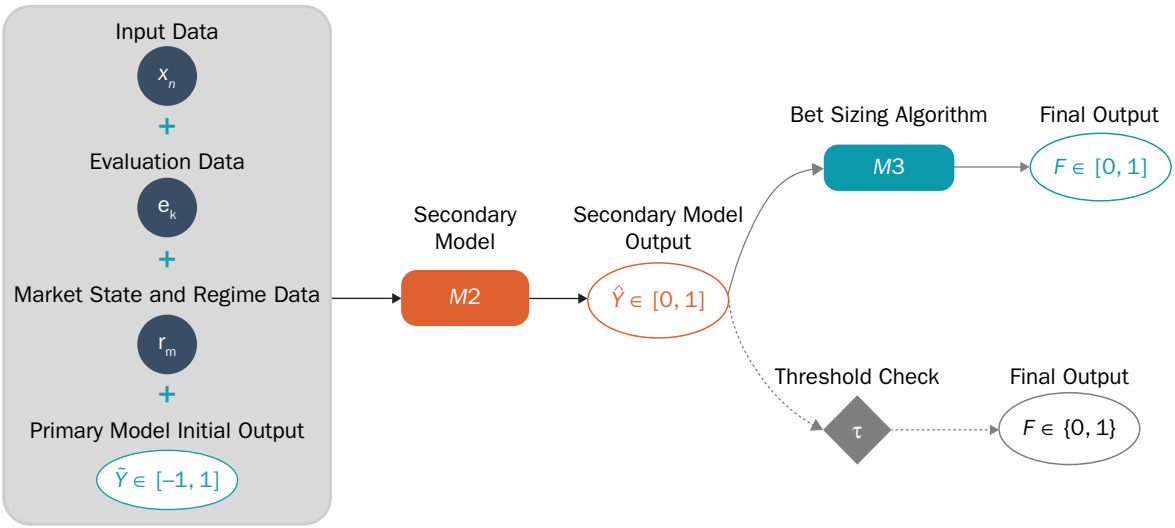
Primary Model Architecture



SOURCE: Recreated from Exhibit 1 in Joubert (2022).

EXHIBIT 2

General MLA



SOURCE: Recreated from Exhibit 2 in Joubert (2022).

General MLA

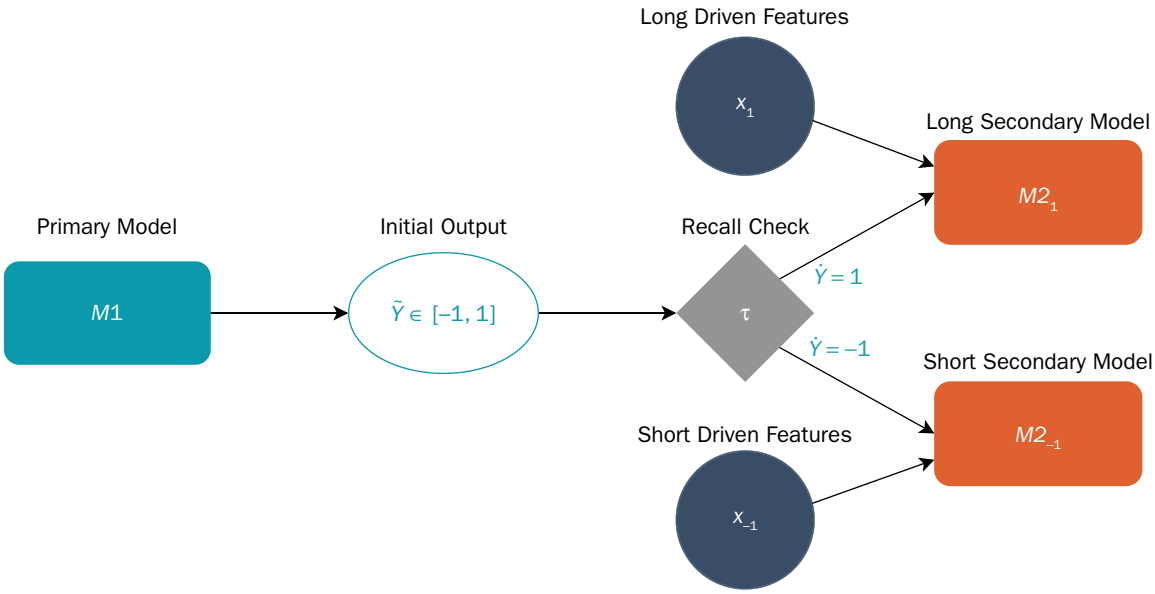
Exhibit 2 presents a general MLA. The target variable is the meta labels, which indicate whether the directional prediction of the primary model is correct. The features that the secondary model considers can be divided into four distinct components.

First, if there is an information advantage, the original dataset that was used in the primary model should be included in the secondary model. Second, to measure recent primary model performance, evaluation statistics should be included. Third, market state statistics and regime-related features could be included because the primary model may not exhibit adequate performance in all market conditions. Last, if the primary model is an ML algorithm, the raw output from the primary model can be included to signify model confidence. The last three components contribute to the modeling for false positives advantage of applying meta-labeling.

The secondary model output can be interpreted as the probability of a true positive. Position sizing algorithms, represented by *M*3, can then be applied to the output. Alternatively, an all-or-nothing approach can be applied using a threshold check. The general architecture can be thought of as a simple long–short architecture that is derived from a feature-driven approach.

EXHIBIT 3

Discrete Long and Short MLA



Discrete Long and Short MLA

If both long and short positions are allowed, then one could simply fit a secondary model as described in the long–short architecture. However, if the features driving a long or short position are fundamentally different from each other, then a single *M*2 model would not be able to effectively capture the features' different characteristics and relationships to the target variable. The secondary model will, therefore, likely underfit the data if these factors differ fundamentally.

Lopez De Prado (2018a) noted, for example, that the features driving a market rally may be different from those of a panic sell-off. An example of this, is to use Volume Synchronized Probability of Informed Trading (VPIN) as a signal that volatility is going to increase, which could indicate a market crash when combined with certain other features (Easley, Lopez De Prado, and O'Hara 2011). These features would be more informative for a short position but not necessarily as informative for a long position.

Separate *M*2 models in this case should, therefore, be applied to the long and short positions to pick up on characteristics that are more relevant to the position taken and thus fully utilize the information advantage. Separating these models can be thought of in the context of classifier selection because each model becomes a local expert in some region of the feature space (the individual labels). Some potential strategies that could benefit from this architecture are momentum and trend-following strategies.

Exhibit 3 displays this type of architecture for an ML primary strategy, in which $M2_1$ represents the secondary model trained on the long positions and $M2_{-1}$ the secondary model trained on the short positions. X_1 and X_{-1} are input features related to the long and short positions. A recall check is used on $\tilde{Y}$ and then passed on to the $M2_1$ or $M2_{-1}$, depending on whether a long or short position should be taken. A similar architecture follows for any other strategy, where the recall check is simply replaced by some decision function separating the labels.

This is the discrete long and short architecture, and it falls under the category of strategy-driven architectures because the attribute (even if it is a common one) of the primary model determines the architecture. Though not shown in the architecture,

the modeling for false positives components (as shown in Exhibit 2) should also be included that relate to each position. These separate models should also be better able to filter out false positives because the market conditions leading to the positions taken are likely to be different.

Sequential MLA

An important advantage of meta-labeling that has thus far not been fully discussed is its capacity to include model evaluation statistics from the primary model in the *M*2 model. Including these statistics will shed light on the recent poor performance of the primary model and help to filter out false positives more effectively.

To further expand on this idea, evaluation statistics could be obtained from the primary model, the output from the first secondary model, and the *M*1 model evaluation statistics. This process could be repeated *K* times to create *K* M_2 models, denoted as $M2_1, \ldots, M2_K$. The concept of sequential feeding secondary models into one another was conceived through a discussion with Lopez de Prado.

The first case would be where $M2_1$ takes in parameters $\tilde{Y}$ and $X_{t,n}$, where $X_{t,n}$ represent *n* features used at time *t* Additionally, evaluation statistics e_{M1} are produced by the primary model at time $t - 1$ and also fed into the $M2_1$ model at time *t*. The $M2_1$ model then produces the output $\widehat{Y}_1$.

Evaluation statistics are obtained from $M2_1$ at time *t*, denoted as e_{M2_1}. The second secondary model $M2_2$ could, in turn, absorb the parameters $X_{tN}, \tilde{Y}, e_{M1}, \widehat{Y}_1, e_{M2_1}$ and subsequently produce $\widehat{Y}_2$ and e_{M2_2}, which could be fed into $M2_3$. This process can be repeated *K* times, after which $\widehat{Y}_k$ is produced, which is the final forecast of the target variable. A variation of this could also be used where the different outputs are fed into a final decision function, similar to stacking.

The choice of *M*2 models is up for debate; however, we suggest using different model types for different characteristics. This adheres to the principle of divide and conquer used in ensemble learning techniques, which states that, regardless of the amount of data, some problems are just too difficult to solve for one classifier (Polikar 2006). For example, $M2_1$ could be a logistic regression model, while $M2_2$ could be a support vector machine (SVM). These models use different principles to generate their output and can, therefore, be used in combination to model different attributes. If the previous secondary model exhibited poor performance, the next *M*2 can pick this up and produce outputs that take this into account.

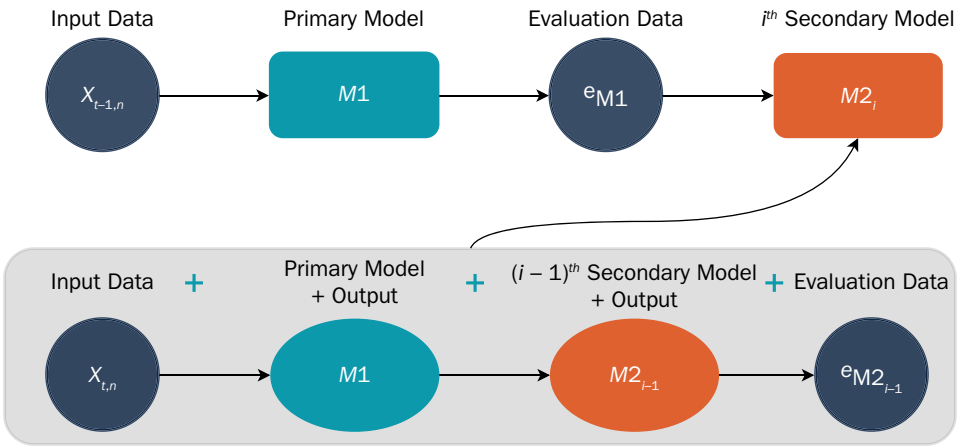
This system is called sequential MLA (SMLA). SMLA is a feature-driven architecture that increases the information advantage while simultaneously contributing to the component of filtering false positives. The architecture for the *i*th *M*2 model is provided in Exhibit 4. The SMLA has certain parallels with ensemble techniques and could help curve overfitting by averaging out the variance error (Polikar 2006).

One interesting way SMLA could be utilized is by considering the missed opportunities a secondary model could potentially filter out. If a secondary model filters out these positive outcomes, then the next model in the sequence could pick this up and adjust the output to include these outcomes.

Conditional MLA

The sequential nature of the SMLA was designed to enhance the advantages of meta-labeling by compensating for recent model performance through reducing the effects of recently underperforming model outputs given the input data. Rather than looking at the secondary models' performance given the input data, one could condition on the input data of the features to build separate secondary models for

EXHIBIT 4
Sequential MLA



each condition of the feature space. The principle underlying this approach is that, because inherent characteristics of certain conditions differ, a separate model should be applied to capture the specific features presented by each condition. This leads to the notion of conditional MLA (CMLA).

A natural choice for CMLA is to condition the model on market state and regime-related data. Secondary models are applied to a subspace of the original feature space, each relating to a specific market state or regime. The unique attributes of each specific state can be modeled by fitting heterogeneous secondary models. Additionally, unique features could be fed into each model to better encapsulate the characteristics of each specific state. A further benefit of this architecture is that inferences can be drawn regarding under which conditions the primary model is profitable, or not. The regime CMLA is derived from a feature-driven approach and is shown in Exhibit 5.

A CMLA can also be specifically created for a tailored strategy. For example, a strategy that entails selling options when implied volatility is high will be specific to the volatility characteristics of the traded period. The market states should thus be conditioned on regimes that encapsulate this approach. Such an architecture relies on a strategy-driven approach. The drawback of a CMLA architecture is that insufficient data points may be available for each condition, resulting in underfitted models.

Ensemble MLAs

Achieving a low variance while maintaining a low bias is crucial for any ML model. One technique that has consistently been shown to achieve this is the use of ML ensemble architectures. Two architectures are discussed in this section, namely bagging and boosting. Although ensemble methods were originally introduced to reduce model variance and thereby increase accuracy, they have been able to successfully address several other ML problems such as feature selection, confidence estimation, incremental learning, error correction, and learning concept drift from nonstationary distributions (Zhang and Ma 2012). Dietterich (2000) also found that ensembles often perform better than any single classifier.

Bagging MLAs. One of the first and most basic ensemble-based algorithms is Breiman's bagging (Breiman 1996), which is short for bootstrap aggregation. Bagging involves training N separate classifiers, each learned by sampling with replacement B times, using a training dataset with features X (or some percentage B from X).

EXHIBIT 5
Regime CMLA

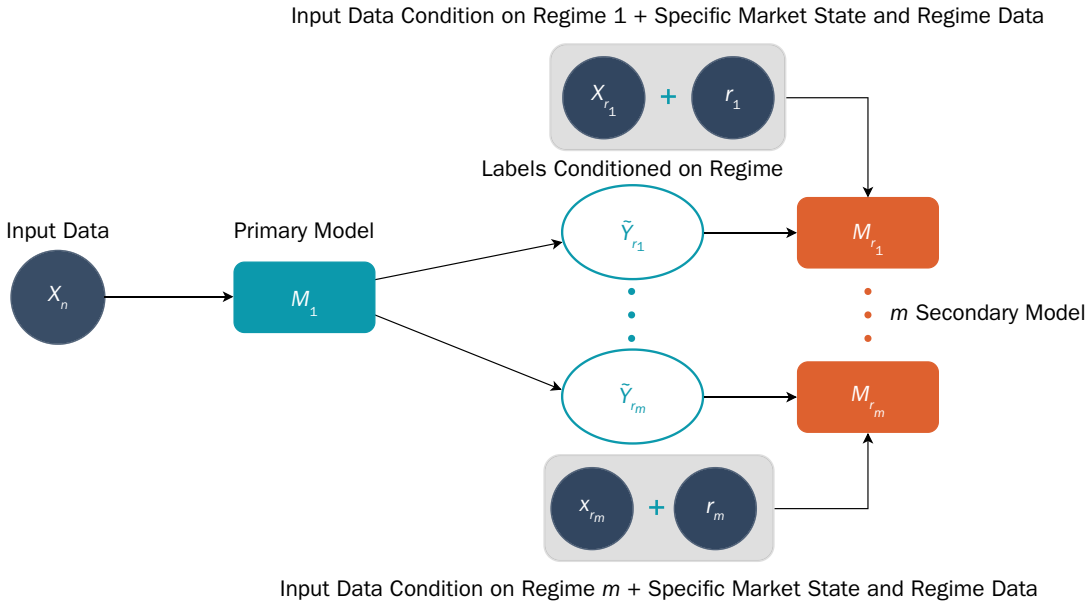
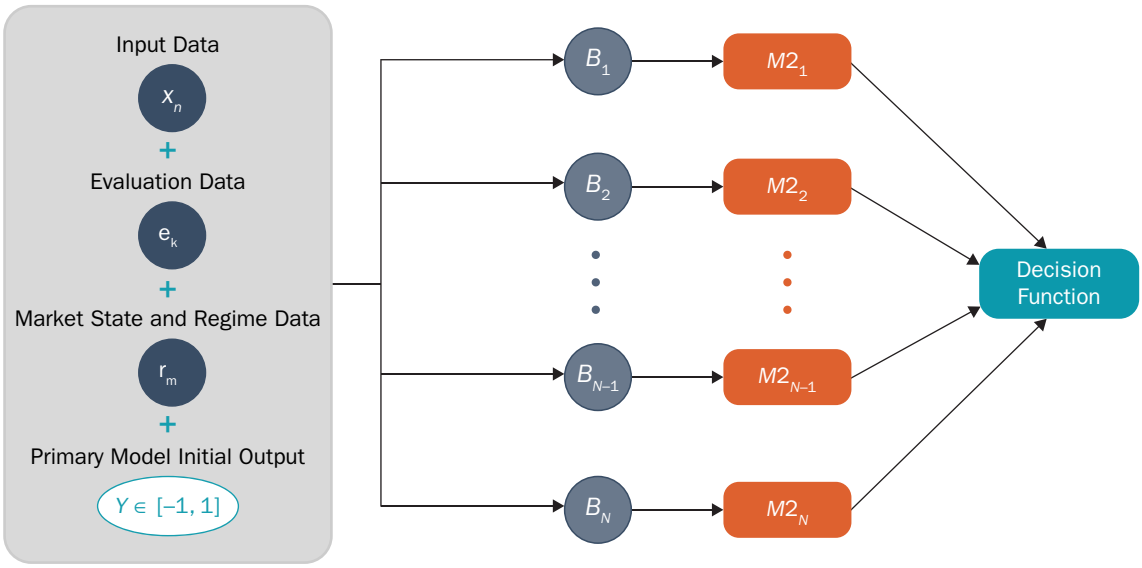


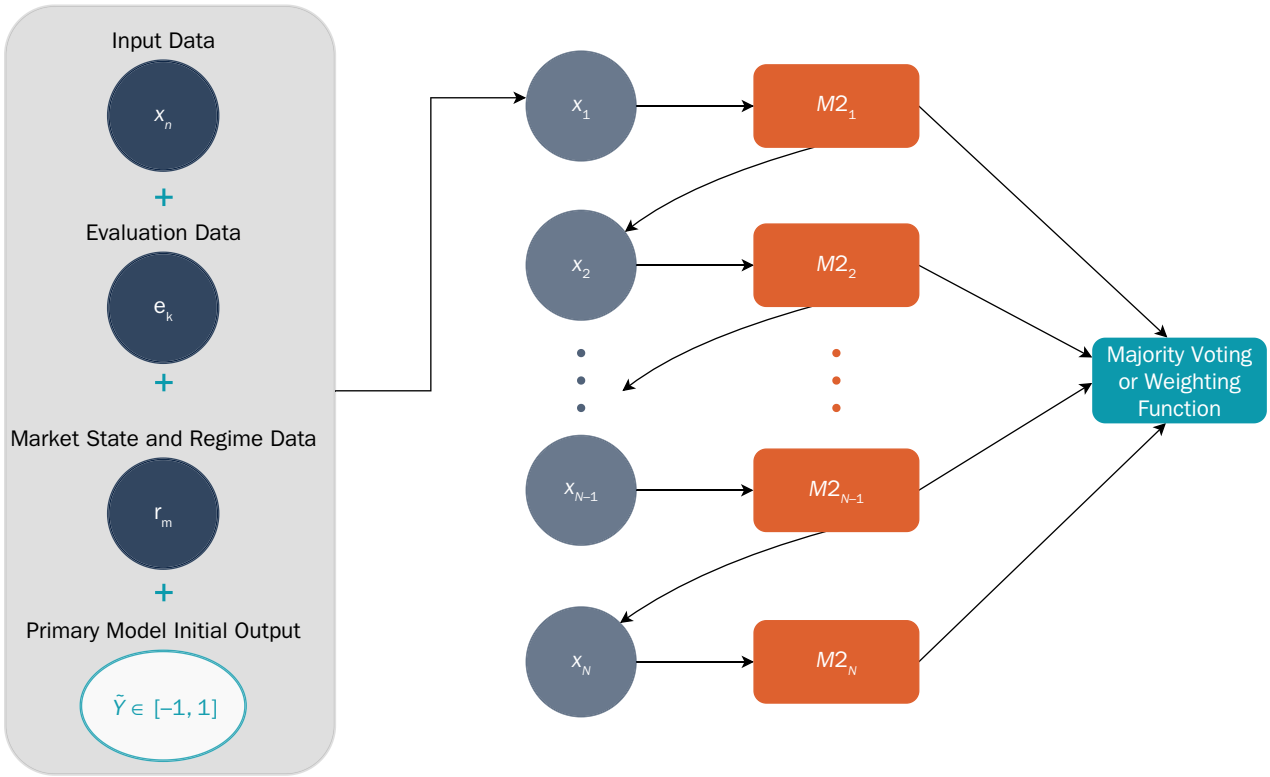
EXHIBIT 6
Bagging MLA



Each classifier is trained on a set of bootstrapped samples, and the variations within these samples ensure that each classifier's decision boundaries shift significantly in response to relatively modest perturbations in the training data. Single-layer perceptrons and linear classifiers such as linear discriminant analysis and SVMs are suitable options for bagging. The classifiers trained are then combined according to some weighting scheme or simply by majority voting. The bagging MLA is displayed in Exhibit 6.

The random forest algorithm (Breiman 2001), an ensemble of decision trees trained with a bagging mechanism, is another inventive form of bagging. A random

EXHIBIT 7
Boosting MLA



forest can also include a random subset selection of features in addition to picking cases. For example, a random forest algorithm can be used as secondary model to discover the hypothetical relationship between the VIX, one-day Standard & Poor's (S&P) 500 Index (SPY) return, short-term interest rate, and other features to determine when it is optimal to short volatility.

Boosting MLA. Boosting is an ensemble learning technique that, like bagging, makes use of a set of base learners to improve the stability and effectiveness of an ML model (Freund and Schapire 1996). A boosting architecture works by sequentially training single classifiers, each of which aims to correct the errors of the previous classifier.

The main principle of boosting is the sequential application of homogenous ML algorithms, in which each ML algorithm attempts to increase the stability of the model by concentrating on the mistakes produced by the previous ML algorithm. The primary distinction between the various forms of this technique is how the error of each base learner is corrected by the subsequent base learner in the sequence. The fundamental benefit of boosting is that it lessens forecast bias and variance. Correcting bias, however, comes at the cost of a higher risk of overfitting. The boosting MLA is displayed in Exhibit 7.

The poor signal-to-noise ratio displayed by financial data complicates the use of an ML model for forecasting purposes because it becomes prone to overfitting. Although bagging mainly solves the problem of overfitting, boosting offers a solution to underfitting. Therefore, one could argue that bagging is more suitable for financial applications, and, because it addresses overfitting, the bagging architecture is especially suitable when the primary model is an ML algorithm. Another advantage of bagging over boosting is that it can be executed in parallel, whereas boosting must always be executed sequentially.

Position Sizing Considerations for MLA

The output from the secondary model are predictions made by a learning algorithm that can be used to determine the position size. The higher the prediction, the larger the position size. For example, the empirical cumulative distribution function can be implemented to size the position accordingly (Joubert 2022).

One could also calibrate these probabilities into a frequentist interpretation and use a well-known position sizing algorithm such as the Kelly criterion. Niculescu-Mizil and Caruana (2005), for example, discovered that methods such as neural nets and bagged trees prediction offer well-calibrated probabilities, whereas other methods such as naïve Bayes, SVM, and random forest must first be calibrated using Platt scaling (Platt 1999) or isotonic regression (Zadrozny and Elkan 2001).

INVERSE META-LABELING

Inverse meta-labeling is the process of using feature importance to build new theories and identify new transformations of original features, as well as the necessary additional features, to build the best primary model possible. Meta-labeling does not produce any new signals, rather it only filters out potential false positives. Therefore, recall will always be lower after applying meta-labeling, and it would be advantageous to build a primary model with a high recall and better-quality trades before applying meta-labeling.

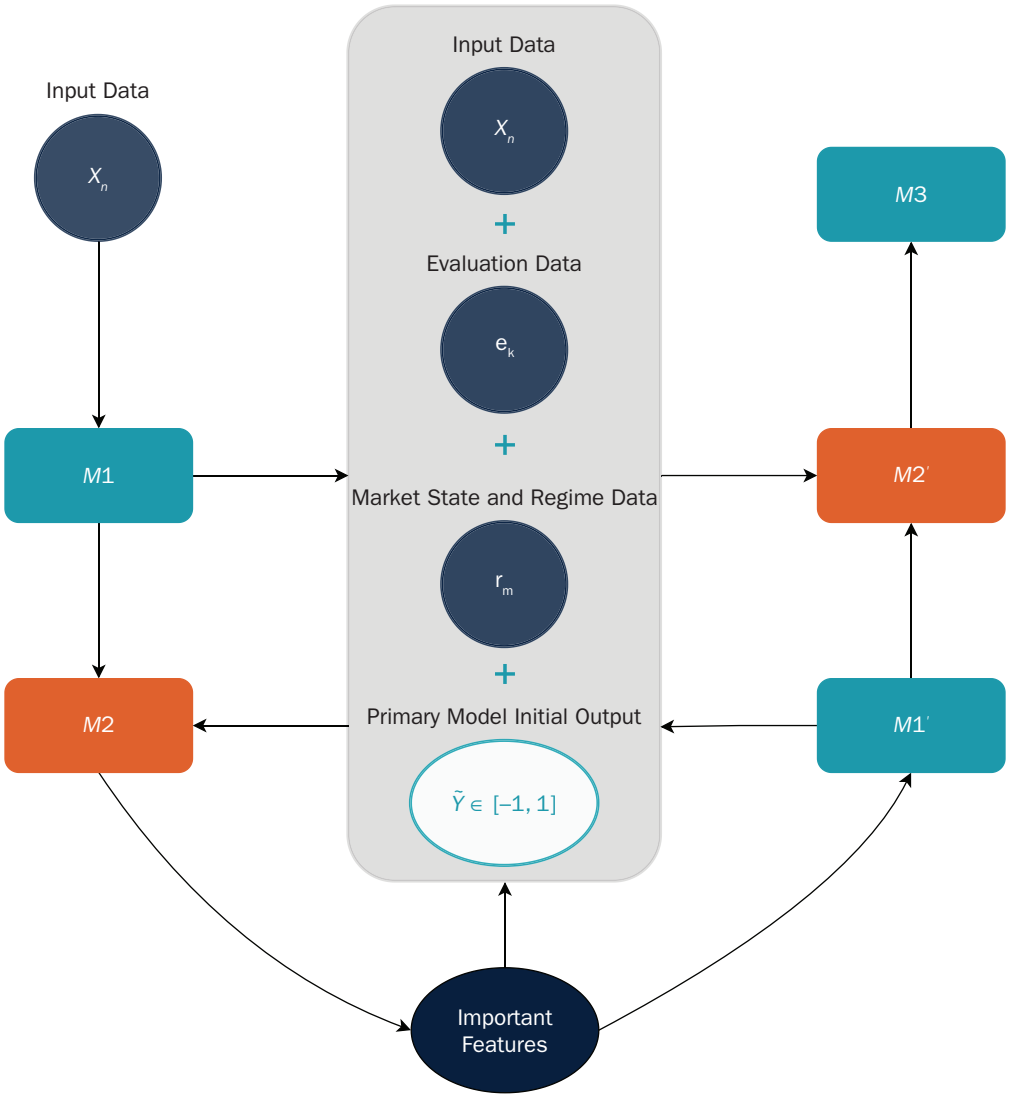
Inverse design is an engineering concept in which the normal design process is inverted. For example, in the field of materials informatics, the structure/property relationship is used to inform synthesis and processing strategies. In the normal setting, the property of the material would be the target variable and the structural characteristic the features used to predict the target variable. However, inverted property/structure relationships are highly desirable because a researcher usually knows which properties they need for a particular application and wants a recipe for the material they must attempt to make in the lab. Li and Barnard (2022), for example, used multitarget random forest regressors to predict these inverse relationships.

One could apply this inverse design process in a meta-labeling setting to inform about important features from the secondary model. These features can then be used to adjust the primary model to increase recall as well as the quality of trades. The target variables are the meta labels that indicate when a trade was profitable or not, and we want to infer the relationship of the features to meta labels. A new improved primary model can be built using these informative features that will amplify the strategy, after which a new secondary model can be fitted to the primary model.

The described methodology is called inverse meta-labeling and is a useful tool for model development. Exhibit 8 displays the inverse MLA, where *M*1' and *M*2' are the adjusted primary and secondary models. In the initial secondary model, we are much more interested in how features are related to the target variable than the actual accuracy of the model while the adjusted secondary is used to model for false positives and position sizing.

A simple case would be to use a random forest to provide information on the importance of the features. A variety of feature-important algorithms are available for use in inverse meta-labeling, such as the model fingerprint (Li, Turkington, and Yazdani 2020) and clustered feature importance (Lopez de Prado 2020); these allow users to drop uninformative features and guide the development of the adjusted primary model. Feature selection algorithms could also be used, such as

EXHIBIT 8
Inverse MLA



Mean Decrease Accuracy (MDA), Lime, or Shapley Additive Explanations (SHAP) (Man and Chan 2021). Man and Chan (2021) also showed that meta-labeling improves the Sharpe ratio and that a careful selection of features can boost this even further. Inverse meta-labeling can also be used to make an informed decision on which architecture to select.

CONCLUSION

Thus far, previous literature on the topic of meta-labeling have only considered the basic form of how this approach can be used; however, there are a multitude of different components that have not been fully explored. Throughout this article, we have highlighted some of the intricacies that have mostly been overlooked, and how the potential of meta-labeling can be maximized to lead to better model development practices.

Theoretical foundations of MLAs are established and ideas such as the discrete long and short, sequential, regime conditioned, bagging, and boosting MLAs are introduced as potential frameworks for practitioners. Inverse meta-labeling is a promising contribution that could potentially serve as a vital tool for building profitable trading strategies.

There are several key areas where further research would be beneficial. First, an empirical analysis of the proposed architectures would reveal what additional benefits they may provide. Second, an exhaustive list of architectures is not provided in this article but rather only a few architectures and how to go about developing them. We posit that there are many more that can be explored, especially strategy-driven architectures. Third, the development of position sizing algorithms from the secondary model's output could be further investigated. Last, we propose investigating model selection criteria for the various MLAs.

REFERENCES

Ang, A., and A. Timmermann. 2012. "Regime Changes and Financial Markets." *Annual Review of Financial Economics* 4 (1): 313–337.

Bahloul, S., M. Mroua, and N. Naifar. 2017. "The Impact of Macroeconomic and Conventional Stock Market Variables on Islamic Index Returns under Regime Switching." *Borsa Istanbul Review* 17 (1): 62–74.

Bongaerts, D., X. Kang, and M. van Dijk. 2020. "Conditional Volatility Targeting." *Financial Analysts Journal* 76 (4): 54–71.

Botte, A., and D. Bao. "A Machine Learning Approach to Regime Modeling." Two Sigma, 2021, https://www.twosigma.com/articles/a-machine-learning-approach-to-regime-modeling/.

Breiman, L. 1996. "Bagging Predictors." *Machine Learning* 24 (2): 123–140.

——. 2001. "Random Forests." *Machine Learning* 45 (1): 5–32.

Chen, J., and E. P. K. Tsang. *Detecting Regime Change in Computational Finance: Data Science, Machine Learning and Algorithmic Trading*. Boca Raton, FL: CRC Press, 2020.

Cruz, R. M. O., R. Sabourin, and G. D. C. Cavalcanti. 2018. "Dynamic Classifier Selection: Recent Advances and Perspectives." *Information Fusion* 41: 195–216.

Dietterich, T. G. "Ensemble Methods in Machine Learning." In *International Workshop on Multiple Classifier Systems*, pp. 1–15. Berlin, Heidelberg: Springer, 2000.

Easley, D., M. M. Lopez de Prado, and M. O'Hara. 2011. "The Microstructure of the 'Flash Crash': Flow Toxicity, Liquidity Crashes, and the Probability of Informed Trading." *The Journal of Portfolio Management* 37 (2): 118–128.

Busseti, E., E. K. Ryu, and S. Boyd. 2016. "Risk-Constrained Kelly Gambling." *The Journal of Investing* 25 (3): 118–134.

Freund, Y., and R. E. Schapire. "Experiments with a New Boosting Algorithm." In *ICMP*, vol. 96, pp. 148–156. San Francisco, CA: Morgan Kaufmann Publishers Inc., 1996.

Graves, A., M. Abdel-Rahman, and G. Hinton. "Speech Recognition with Deep Recurrent Neural Networks." In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 6645–6649. Washington, District of Columbia: IEEE, 2013.

Gu, J., Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, and T. Liu. 2018. "Recent Advances in Convolutional Neural Networks." *Pattern Recognition* 77: 354–377.

Harvey, A. C. *Forecasting, Structural Time Series Models and the Kalman Filter*. Cambridge, England: Cambridge University Press, 1990.

Horvath, B., Z. Issa, and A. Muguruza. 2021. "Clustering Market Regimes Using the Wasserstein Distance." *arXiv* 2110.11848.

Joubert, J. F. 2022. "Meta-Labeling: Theory and Framework." *The Journal of Financial Data Science* 4 (3): 31–44.

Li, S., and A. S. Barnard. 2022. "Inverse Design of Nanoparticles Using Multi-Target Machine Learning." *Advanced Theory and Simulations* 5 (2): 2100414.

Li, Y., D. Turkington, and A. Yazdani. 2020. "Beyond the Black Box: An Intuitive Approach to Investment Prediction with Machine Learning." *The Journal of Financial Data Science* 2 (1): 61–75.

Lipton, Z. C., C. Elkan, and B. Narayanaswamy. 2014. "Thresholding Classifiers to Maximize F1 Score." *arXiv* 1402.1892.

Lopez de Prado, M. *Advances in Financial Machine Learning,* 1st ed. New York City, NJ: Wiley, 2018a.

——. 2018b. "The 10 Reasons Most Machine Learning Funds Fail." *The Journal of Portfolio Management* 44 (6): 120–133.

——. *Machine Learning for Asset Managers*. Cambridge, UK: Cambridge University Press, 2020.

Man, X., and E. Chan. 2021. "The Best Way to Select Features? Comparing MDA, LIME, and SHAP." *The Journal of Financial Data Science* 3 (1): 127–139.

Niculescu-Mizil, A., and R. Caruana. "Predicting Good Probabilities with Supervised Learning." In *Proceedings of the 22nd International Conference on Machine Learning*, pp. 625–632. New York City: Association for Computing Machinery, 2005.

Platt, J. 1999. "Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods." *Advances in Large Margin Classifiers* 10 (3): 61–74.

Polikar, R. 2006. "Ensemble-Based Systems in Decision Making." *IEEE Circuits and Systems Magazine* 6 (3): 21–45.

PredictNow.ai. "Our Approach." 2022, https://predictnow.ai/approach/.

Severo, M., and J. Gama. "Change Detection with Kalman Filter and CUSUM." In *International Conference on Discovery Science*, pp. 243–254. Berlin, Heidelberg: Springer, 2006.

Yegnanarayana, B. *Artificial Neural Networks*. Connaught Circus, New Delhi: PHI Learning Pvt. Ltd., 2009.

Zadrozny, B., and C. Elkan. "Obtaining Calibrated Probability Estimates from Decision Trees and Naive Bayesian Classifiers." In *ICML*, vol. 1, pp. 609–616. San Francisco, CA: Morgan Kaufmann Publishers Inc, 2001.

Zhang, C., and Y. Ma, eds. *Ensemble Machine Learning: Methods and Applications*. New York: Springer Science & Business Media, 2012.