A Laboratory for Machine Learning in Finance

J. Joubert, A. Singh

Abstract

In the summer of 2018 we attended a conference organized by Quantopian in which we heard Dr. Marcos Lopez de Prado outline the challenges of building successful quantitative investment platforms. His book, Advances in Financial Machine Learning provides solutions to many of the problems faced by the quantitative finance community. We however could not find a cogent implementation of these ideas in the public domain. We considered this to be an opportunity to build an open source package in Python (a language that has gained considerable following) that implements the concepts present in the book while using good software engineering techniques. We aspire to make this a research platform and repository - where we along with contributors from the community add tools, techniques, algorithms and research papers to the benefit of all quantitative finance practitioners.

We have made good progress and the results and response from the community is very encouraging. In days and months to come we expect to add more functionality to this package.

Keywords: Python, Package, Open Source, Machine Learning,
Meta-Labeling, Triple Barrier Labeling, Volume Clock

1. Introduction

In recent years Python has been attracting a lot of attention for being the go to language for data science and machine learning. Many large organizations have contributed to this movement by making their in-house tools available on open-source platforms like Github. For example, Google's Tensorflow, Facebook's PyTorch, and AQR's pandas.

Prior to the release of Tensorflow in November 2015, there was a constant debate between which language was better - Python or R. In the plot below, generated using Google Trends, you can see how Python has become the language of choice. This is due to the release of new open source packages with widespread adoption coupled with Python's ability to produce high-quality production-ready code.

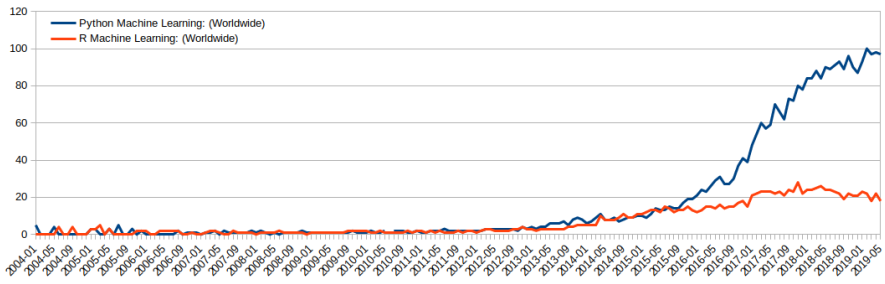

Figure 1: Google trends
Google 2019

"The Y axis represent search interest relative to the highest point on the chart for the given region and time. A value of 100 is the peak popularity for the term. A value of 50 means that the term is half as popular. A score of 0 means there was not enough data for this term." (Google 2019)

In 2018, Wiley published a first of its kind textbook on financial machine learning titled Advances in Financial Machine Learning by Marcos Lopez de Prado. This book quickly became an important source of foundational ideas, concepts and principles underlying the use of machine learning in finance. Although a year later, the techniques outlined in the book had become popular, there was no open-source implementation of the ideas. It was for this reason that we saw the opportunity to code up a well-designed package based on agile project management and modern day software engineering tools.

The motivation for the package was to contribute to the quantitative finance community as well as to help speed up the learning process for students. Overall the goal of this package is to reduce implementation friction, fast track learning, and to help bridge the gap between theory and practice. Typically research of a quantitative strategy is a process that entails reading an academic paper, coding up an implementation, trying it on proprietary data, and then analyzing

the results. This package would speed up this entire process by allowing users to jump directly to applying the methodology to their data.

The package also benefits institutions teaching post graduate courses in financial machine learning. We have provided two years of free sample data based on the new data structures along with a tested and well documented code base, allowing students to more easily build an intuition behind the techniques and play around with real data. We have additionally expanded on a few concepts to help answer questions that we struggled with, such as Meta-Labeling where we provide a toy example and results using a finance setting.

The rest of the paper is structured as follows: Section 2 reviews 6 of the 10 reasons why most machine learning funds fail and explains the package's solutions. Section 3 discusses the paradigm shift from Lopez de Prado's work relative to traditional methods such as factor investing and Section 4 elaborates by explaining how these concepts can be translated across strategies. Section 5 takes a high level overview regarding the package design, the tools we used, and the project management style we followed. Section 6 describes how we built an open source community around the package and section 7 provides some tutorials to help readers get started. Section 8 highlights the vision for the project going forward and a thank you note to WorldQuant University. Appendix A provides a toy example of Meta-Labeling.

2. Answering Challenges Regarding Financial Machine Learning

In 2018 a paper published in the Journal of Portfolio Management (Lopez de Prado 2018b) highlighted the ten challenges faced by most machine learning funds.

The 10 challenges are:

1. Working in Silos
2. Research Through Backtesting
3. Chronological Sampling
4. Integer Differentiation
5. Fixed-Time Horizon Labeling
6. Learning Side and Size Simultaneously
7. Weighting of Non-Independent Identically Distributed Samples
8. Cross-Validation Leakage
9. Walk-Forward (or Historical) Backtesting
10. Backtest Overfitting

The textbook, as well as the package address these problems. At the current time of writing we have addressed 6 of them listed below.

2.1. Setting up a Financial Research Lab

Discretionary portfolio managers are well known to work in silos. This is so that the fund can diversify its investments across various styles. A typical mistake is to follow the same process with a systematic team. A fund would hire two or three PhD quantitative analysts, split them and then require that each generate their own alpha strategy. This approach doesn't work as typically each member will settle for an investment strategy that looks great on an overfit backtest, leading to a false positive or they will revert to the tried and tested factor portfolios which are overcrowded and have poor Sharpe ratios.

The solution to this is to setup a financial laboratory in the same way that top universities have set-up their scientific research labs. There is an initial overhead associated with this, however, it takes just about the same amount of effort to build your first investment strategy as it does for 100.

To build such a team, the meta-strategy paradigm (Lopez de Prado 2015) is proposed. In this framework tasks of the assembly line are split up into subtasks. The key idea is that each quant specializes in a subtask, whilst having a high-level understanding of the entire process.

It is this philosophy which we have applied to the development of the open source package mlfinlab and the research that follows. On a higher level, we have titled it the Open Source Hedge Fund Project (Joubert and Singh 2019h) with the key idea being to bring people together in the development and research of cutting edge machine learning techniques applied to finance.

To do this we have made use of Github (Joubert and Singh 2019b) as our development platform. We host 3 repositories which are of interest:

1. Mlfinlab (Joubert and Singh 2019d) is the package developed from which all further research is built upon. It is an open source implementation of Lopez de Prados work but is open to other key authors works.
2. Research (Joubert and Singh 2019f) houses the various Jupyter Notebooks which each tackle a single concept. They provide useful insights into the techniques and show that the methods work using real data.
3. Presentations (Joubert and Singh 2019e) contain the various slide show presentations used at conferences and speaker events.

A key feature of this project is that it is open to the public, anyone can contribute and since our launch, we have had several key contributions from other members. At the time of writing, we have a team 5 and are growing. Our members also follow the meta-strategy paradigm, with each member focusing on a specific implementation of the literature.

Each member in our team is from a different geographic location and thus we have turned to tools which enable us to run in a virtual team environment.

2.2. Feature Analysis

The second pitfall is research through backtesting. This is the process where an analyst will build an investment strategy, backtest it, and then compare performance metrics. They will run this process in an iterative cycle until they get a favourable result. This is likely to lead to false discoveries and the failure of investment strategies.

A solution is rather to turn to an analysis of the features which were predictive of an outcome. To better understand which information contributed to the model performance. By doing this we can improve the features which added the most information and remove those that contributed to the noise.

Chapter 8 of the textbook addresses feature importance as a tool. We have made use of this tool when completing our research on Meta-Labeling, in order to understand which features were predictive in filtering out false positives. This was largely due to our use of the Random Forest algorithm. We still need to develop the tools to able to run the same analysis on other algorithms such as SVMs and Neural Networks.

An example of this technique can be found in a presentation titled Market Microstructure in the Age of Machine Learning (Lopez de Prado 2018a). Below is an image where he turns to feature importance to determine which features provide the most information.

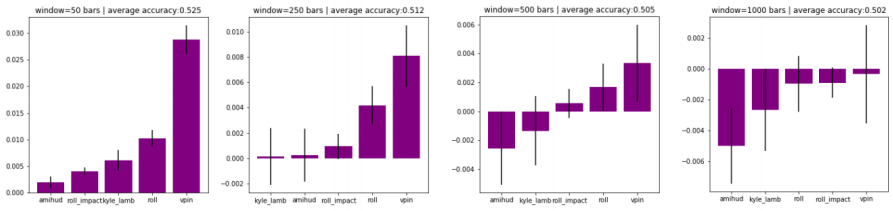

Figure 2: Example of Feature Importance
Lopez de Prado 2018a

We applied the same technique in our Meta-Labeling notebooks where we attempt to filter out false positives.

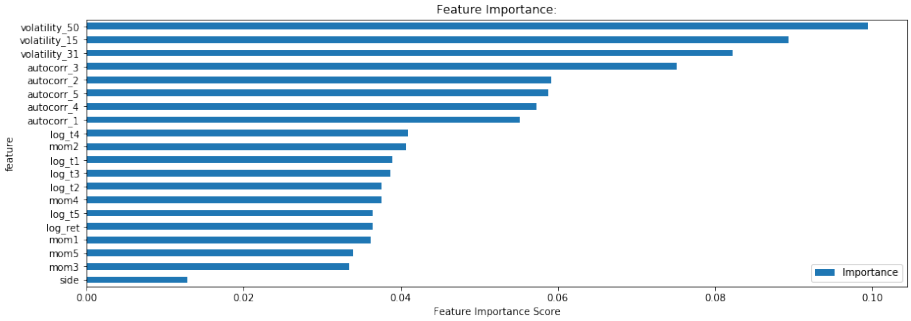

Figure 3: Features Importance: Filtering False Positives
Joubert and Singh 2019c

2.3. Better Sampling Techniques

Machine learning in finance focuses on forecasting stock price movements using stock market data such as price and volume. According to the literature (Thierry and Hélyette 2000), stock price movements are nonlinear and stochastic in nature. Specifically, trading activity is rarely uniform during a day or a week or a month. It varies with the information flow in the form of macro-economic data releases, news about political leaders or company specific announcements.

The vast majority of academic papers regarding financial machine learning will however, make use of chronological sampling, also known as fixed time interval sampling. This technique should be avoided for two reasons:

1. Markets do not receive information at fixed time intervals, there are periods where more information is present, for example before the market open or a large corporate event. Easley, López De Prado, and O'Hara 2012b provide insights into why chronological intervals may lack meaning. Chronological sampling means that we over-sample information during quiet periods such as midday and not enough during busy periods like the open and close.
2. In part due to the over and under-sampling - the chronological clock leads to poor statistical properties such as serial correlation, heteroskedasticity, and non-normality of returns (Easley, López De Prado, and O'Hara 2012a).

Fama and Blume 1966 showed that daily returns are longer-tailed than the normal density. It is, therefore, necessary to sample data using a paradigm called event-based time as discussed in Easley, López De Prado, and O'Hara 2012b. These techniques involve sampling a session into equal volume chunks or bars (for instance, 100,000 contracts or shares) or dollar bars ($1 million) etc. Empirical analysis shows that these methods have better statistical properties. In addition to dollar and volume bars, there are also tick bars (100,000 ticks).

Exhibit 1 - Partial recovery of Normality through a price sampling process subordinated to a volume, tick, dollar clock

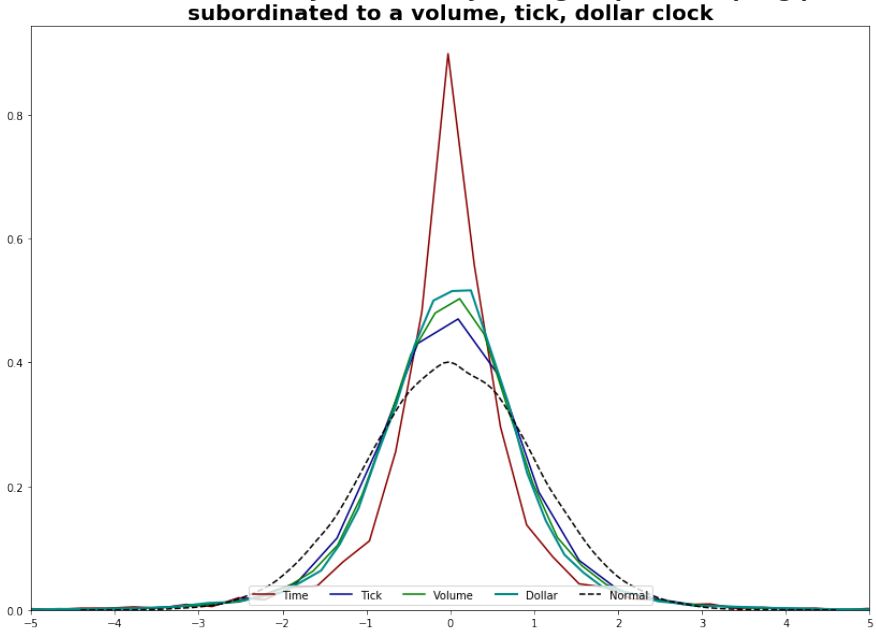

Figure 4: Partial recovery of normality
Joubert, Jacques. 2019b

The figure above shows how the S&P 500 E-Mini Futures log returns have a partial recovery of normality when we apply the new sampling techniques. This graph is inspired by (Easley, López De Prado, and O'Hara 2012b), we simply performed the same analysis but on our own data.

Below we show the Jarque-Bera test statistics for these bars which highlight that dollar bars are the closest to normality compared to all other bars (because its test statistic is the smallest).

Test Statistics:

1. Time: 1782853
2. Tick: 2898186
3. Volume: 337591
4. Dollar: 143045

Additionally, we computed the autocorrelation (ACF) for the various methods:

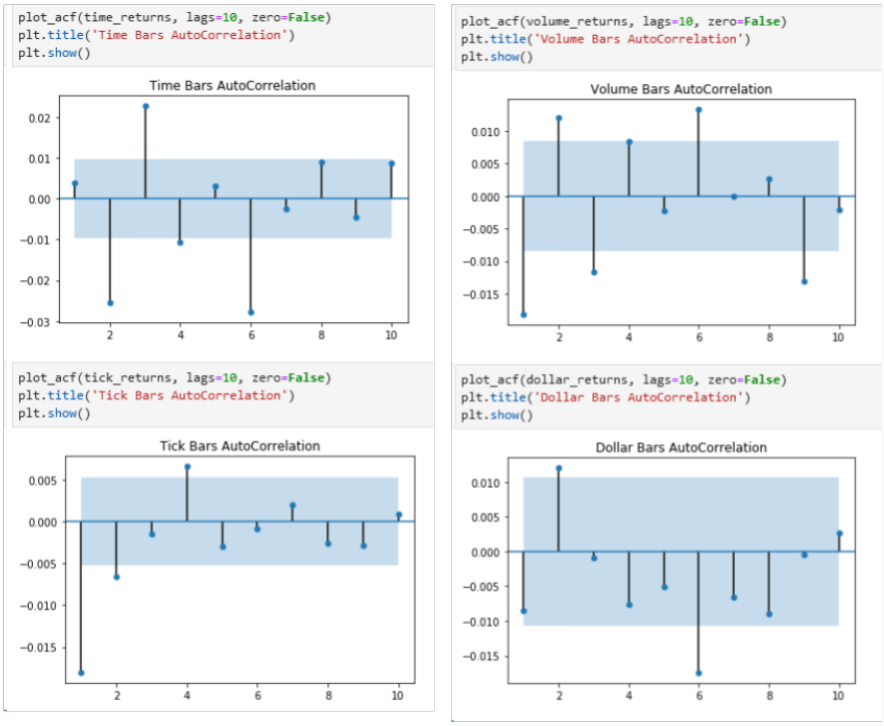

Figure 5: ACF of Sampling Techniques
Joubert, Jacques. 2019b

The bars we have discussed here are known in the textbook as standard bars. There are additional data structures such as information driven bars which consist of imbalance bars and run bars. We have coded up implementations of each in the package and further analysis of empirical evidence can be found on the research repository (Joubert and Singh 2019f).

2.3.1. Barriers to Entry

Implementing the various sampling techniques was a large barrier to entry that we lifted. To date, there wasn't an implementation that could be found online. There are a few problems that we needed to address:

1. The size of the tick data files is large, ours was 25 gigs and up. This is a problem because pandas make use of RAM to process operations and unless the end user had a significant amount of RAM, the program would crash. To overcome this problem we introduced a caching algorithm that would read the raw data in batches and write to disk when memory was running low. The end result is the new financial data structures saved to CSV file or if the user would prefer, an in-memory pandas DataFrame.

2. The code needs to run fast. We were mindful of computational complexity when implementing the algorithm. Currently, the algorithm completes in less than 10 minutes on a 25 Gig CSV file. This is within a reasonable waiting time limit.

The last big barrier to entry was that HFT data is expensive to source. We sourced our data from TickData LLC at the price of 750 US Dollars per instrument. We are not affiliated to TickData LLC and would recommend their data as $750 is very cheap in comparison to competitor products and the data is clean.

We realized that other users may not want to purchase their own data and thus we have provided a two-year sample of transformed tick data (Joubert and Singh 2019g), in the form of the new financial data structures. In particular, we provide a sample of Tick, Volume, and Dollar bars. As well Dollar Imbalance Bars. Our hope is that this will aid the community in trying out new methods and enable further research.

2.4. Feature Engineering: Maintaining Memory

Inferential analysis of data comprises of using a sample of data to describe the characteristics of a population (such as average height or mean return). For such an analysis and inference to be accurate, it is necessary that the underlying data generation process to remain constant. In the context of finance, the mean return and variance of those returns should not change over time. If they change then it is hard to predict the expected return (or risk as defined by the volatility) of that stock in some time in the future.

A similar requirement exists in the case of supervised machine learning (SML). In SML, unseen observations are mapped to a set of labeled examples (features) to know the label (decision or outcome) of the new observation. If the data (features, in the case of SML) is not stationary or constant then the machine learning algorithm would not be able to correctly infer the label of the new observation. Hence, stationarity becomes a necessary condition for inferential analysis and supervised machine learning. However, asset prices experience trends (or drifts) that make the time series non-stationary but the underlying trend is also useful in prediction. This leads to a challenge how can one make the time series stationary while retaining its predictive power (or memory).

Hosking 1981 showed that fractionally differenced processes exhibit long-term persistence and anti-persistence; the dependence between observations a long time span apart decays much more slowly with time span than is the case with commonly used time series models. Hence, fractional differentiation is used to make the time series stationary while retaining as much memory as possible.

We illustrate the concept in the figure (below) where we difference the E-Mini S&P 500 futures log-prices using different differencing fractions (d, shown on the

x-axis). The ADF statistic is on the right y-axis, with the correlation between the original series and the fractionally differenced series on the left y-axis.

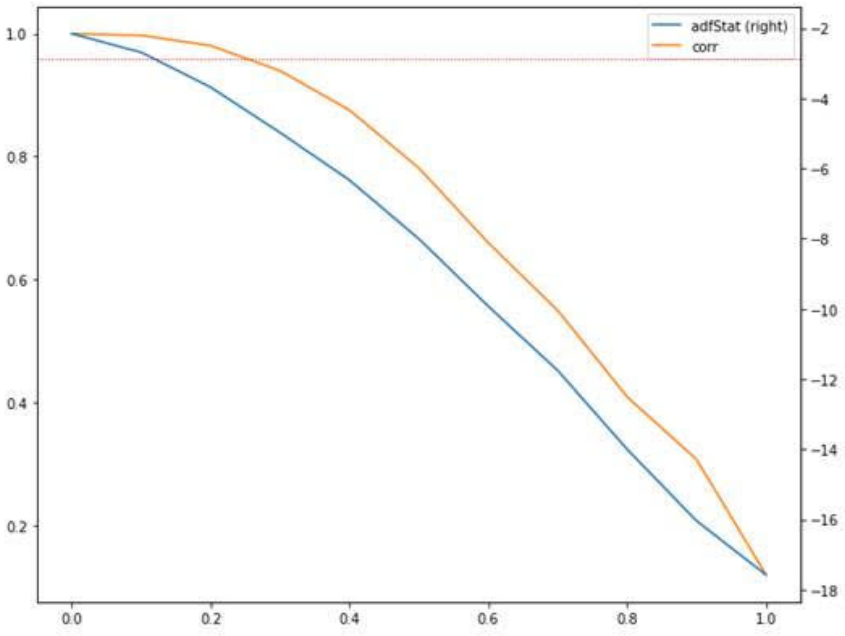

Figure 6: ADF statistic as a function of d, on E-Mini S&P500 futures log-prices
Singh, Ashutosh. 2019

The chart shows that ADF statistic reaches 95% critical value when the differencing amount is less than 0.2 and the correlation between the original series and the new fractionally differenced series is over 90%. This shows that the new series is not only stationary but also retains considerable memory of the original series.

This functionality has been included in the package with an accompanying research notebook to help users build a deeper intuition, Singh, Ashutosh. 2019.

2.5. Financial Labelling Techniques

In the majority of the literature, authors will make use of a labeling scheme where they classify the next period's directional move as either a 1 for a positive move, a -1 for a negative move, and some authors may add a threshold level that if the return is not above or below it, then a 0 label is provided.

This technique has a few flaws. First, the threshold level is usually static and stock returns are known to be heteroskedastic, the volatility changes over time and a fixed threshold value fails to account for this. Second, using this -1, 0, 1

scheme fails to account for positions that would have been closed by stop loss or profit taking orders. A more advanced technique such as the Triple Barrier method (Lopez de Prado 2018), addresses these concerns.

In derivatives pricing, a series of stock prices can be modeled using Geometric Brownian Motion. Similarly, in the Triple Barrier method, we assume that stock prices follow a random walk with some drift and variance, we then label this path.

At a given timestamp, three barriers are set. An upper and lower horizontal barrier to represent a take profit and stop loss levels respectively. A third and vertical barrier is placed to represent the end of the duration of the trade.

Should the path of a stock reach the upper barrier before the vertical then a value of 1 is returned, conversely if it reaches the bottom barrier then a -1, however should the stock price reach the vertical barrier first then a 0 is returned. This is still a -1. 0, 1 scheme, however, we are labeling a path of returns rather than the next directional move.

The horizontal barriers are determined by calculating the daily standard deviation of the log-returns multiplied by a user defined multiple. For example a [1, 1] tuple will set both barriers to be equal to 1 standard deviation.

The following figure provides an example:

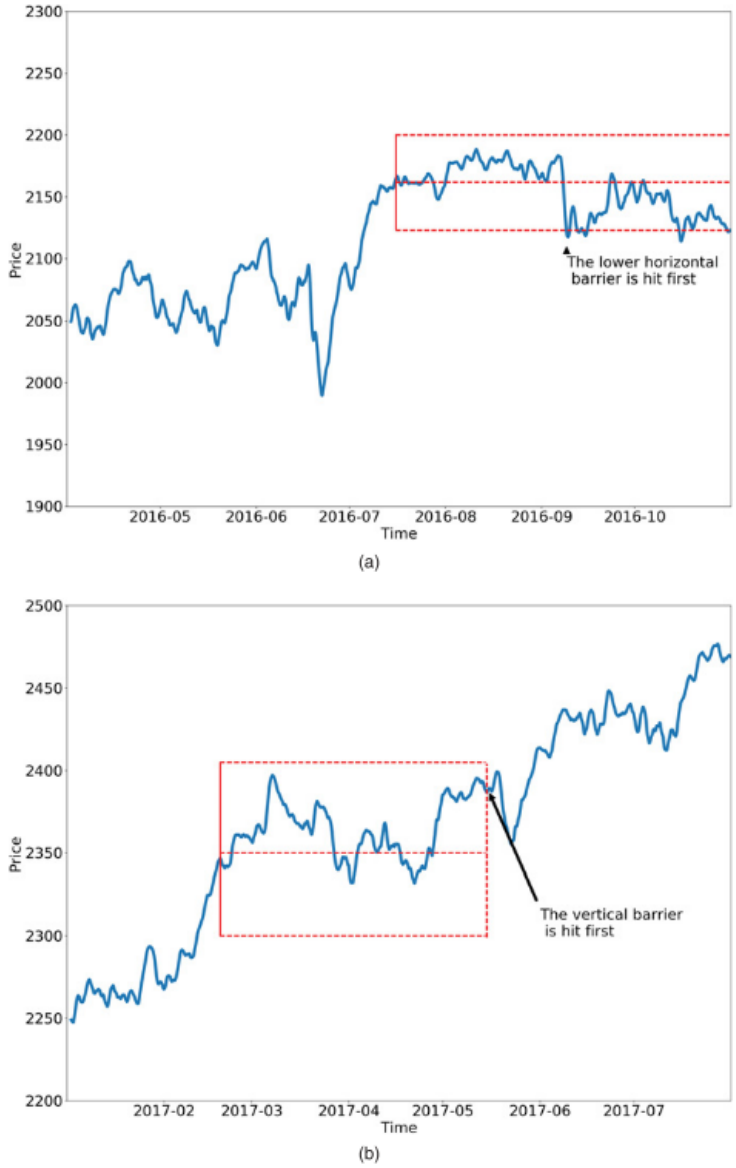

Figure 7: Triple Barrier Labeling
Lopez de Prado 2018

In chart (a) we can see that the lower horizontal barrier is first reached, a -1 value is returned. In chart (b) the path never reaches the horizontal barriers and triggers a 0 label when the vertical barrier is reached.

The code for this new labeling technique has been included in the package and for an example notebook users can turn to the research repository chapter 3 (Joubert and Singh 2019f).

2.6. Learning Side and Size of a Position

Many practitioners will build a single model that determines both the side of the position (long or short) as well as the size of the position. These two components are better suited for two different models.

The primary model is concerned with the side of the position, its job it to determine the fair value of an asset or the direction of the next move using a particular feature matrix. The size of the position is more of a risk management decision.

Lopez de Prado has introduced the concept of Meta-Labeling which helps to overcome the problem of determining the size. It is a technique that has been shown to improve the F1 score as well as other performance metrics (Joubert and Singh 2019c). For those interested we have included a toy example of meta-labeling in the appendix.

It should be noted that many machine learning algorithms have a high precision and low recall. The problem with this is that those strategies will trade very infrequently, should the strategy enter a drawdown it may stay there for an extended period of time. It is for this reason that we want to focus on improving the F1 score.

The core idea behind Meta-Labeling is to use a secondary model to help filter out false positives in the primary model. It should be setup so that the feature matrix consists of variables that are explanatory of false positives. The following figure provides a proposed architecture:

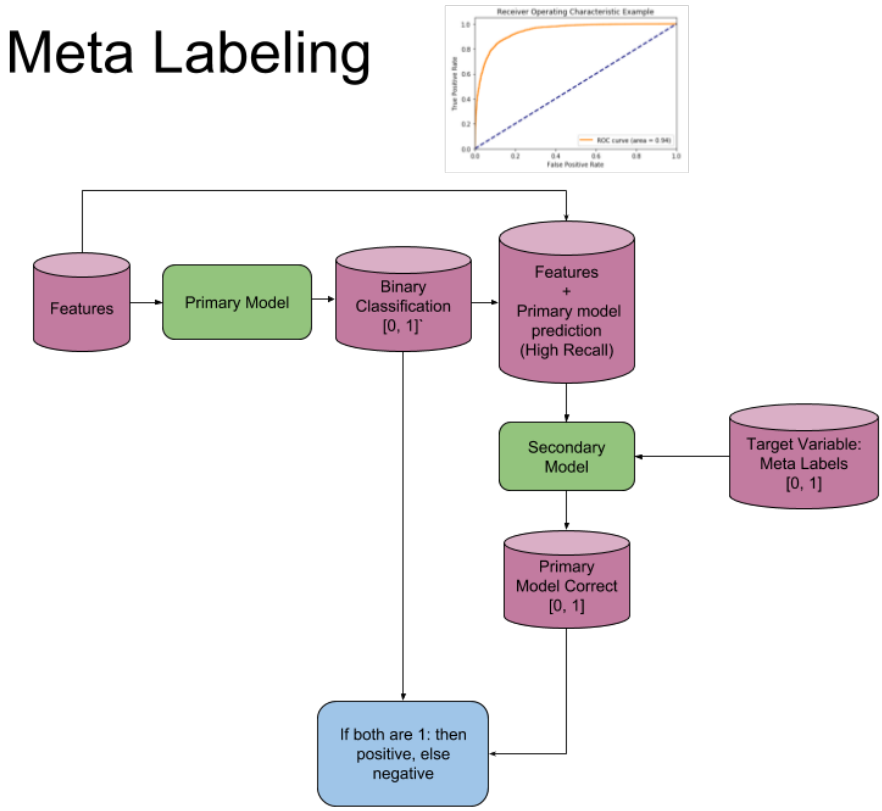

Figure 8: Meta-Labeling Architecture
Joubert and Singh 2019c

The output from the second model will be a value between 0 and 1 which can be viewed as the model's confidence value. We can then plug on a bet sizing algorithm to map these confidence values to a position size. Thus we will take large positions on trades for which the model is confident and smaller positions on those that have a lower probability.

The code for meta-labeling has been added to the package and was the core of our research for the first Capstone project titled: Does Meta Labeling Add to Signal Efficacy? (Joubert and Singh 2019a)

3. Paradigm Shift

Lopez de Prado's book provides us with a very different way of building investment strategies. Rather providing readers with alpha generation techniques, the book provides a framework which can be leveraged to produce robust investment strategies.

The techniques he proposes are rather different when compared to the style of factor-based investing from Grinold and Kahn 2000 or Chincarini and Kim 2006.

1. A lot of focus is spent on enhancing features and sampling techniques to boost statistical properties. The exciting thing is that those techniques could be applied to factor-based portfolios and the hope is that they would perform even better (perhaps a good future paper would be able to test this empirically).
2. Practitioners have long known that the covariance structure between various assets is an important feature for forecasting risk as well as returns. This often leads to a model which follows a many-to-many architecture such as a vector autoregression (VAR) model, Lopez de Prado however, makes use of a many-to-one structure. He proposes modeling one asset at a time, which is enforced by making use of the volume clock sampling techniques and its derivatives.
3. Lopez de Prado also takes more of a trading approach rather than an investing one, which makes a lot of sense in the context of machine learning. A good example is how structural breaks are used to set-up trades. These trades are then modeled using machine learning and the position sizes are determined using meta-labeling in combination with bet sizing algorithms.
4. At first glance, readers may assume that Lopez de Prado suggests using features derived from only price action such as market micro-structure features and structural breaks but that is only because he elaborated on those chapters. Our understanding is that the models discussed in his work can take a wide range of features, from traditional accounting ratios and macroeconomic data to satellite imagery and features compressed using dimensionality reduction techniques.
5. Another key contribution from his work is stressing the importance of keeping count of the number of trials you run in order to avoid a false discovery. He proposes a Deflated Sharpe ratio and is vocal about the implications of running backtests in an iterative process. Rather he suggests using metrics such as feature importance and correct cross-validation techniques which are finance specific.

A paper by researchers at AQR Capital offers a solution much closer to the paradigm of factor investing. It is Empirical Asset Pricing via Machine Learning by Gu, Kelly, and Xiu 2018.

4. Use of Techniques Across Strategies

Three ideas that stand out to us from the hoard of great ideas in the textbook are meta-labeling, data preparation, and feature engineering. These concepts are implemented into the mlfinlab package and are readily available.

We would like to give special attention to Meta-Labeling as it has solved several problems faced with strategies:

1. It increases your F1 score thus improving your overall model and strategy performance statistics. It also leads to shorter drawdown periods when compared to strategies with a higher precision and a lower F1 score.
2. It solves the problem of needing to use online algorithms due to the non-stationary nature of financial markets. A typical example is to train a model on 70% and then test it on 30% of the data. Given this model has favourable performance metrics, it would stop working when a structural break occurs. A toy example is to consider a trending strategy. When the market stops trending, the strategy will likely lose money. Meta-Labeling avoids this by identifying market regimes/states in which the primary model will perform poorly. It helps to filter out the false positives.
3. It allows us to determine optimal position sizes by weighting the trades based on the model confidences. Thus positions with high uncertainty receive less money.

5. Package Design

The following section outlines the design principles and tools that we made use of in the creation of mlfinlab. We also provide a brief explanation of the package structure.

5.1. Lean Startup Principles

The Lean Startup is a book written by Ries, Eric. 2011 which outlines the process of running a startup company based on validated learning and a build-measure-learn feedback loop.

The key idea is to set up a cycle of ideation, building a minimum viable product (MVP), releasing to market to measure various performance metrics, and learning from the data.

An example of we did this was:

1. Came up with the idea that a python package based on Advances in Financial Machine Learning would be useful.
2. Coded up a very small MVP giving users the ability to create the various data structures.

3. Released it to early adopters.
4. Measured the feedback in terms of Github stars, Reddit votes, and Twitter retweets.
5. Identified which groups of people wanted the product so we could target them in the future and listened to which parts of the literature they wanted us to develop next.

Overall this process has yielded great results by making sure that we build implementations which are in demand, it also had the added benefit of building a following as we developed the product.

Below are some of the metrics we are using to measure product/market fit:

- PyPi package downloads
- Github Stars and the number of unique clones
- Reddit comments and votes
- Twitter retweets and likes

Interestingly the two countries with the highest number of downloads are the United States of America and China.

5.2. Continuous Integration

Continuous Integration (CI) is a software engineering practice where developers push their iterative changes to the code base to a central and shared repository, where a build server runs automatic scripts to check for best practices such as code style checks, 100% code coverage, and unit tests passing are enforced. These checks allow the team to detect problems early on and provide guidelines for writing production-ready code.

We make use of the following tools:

- Github for version control and repositories
- Travis as a CI tool and automatic build server
- GitKracken as a version control GUI
- Bash scripts to execute checks
- Pylinter to enforce code coverage
- Test-driven development with 100% code coverage

5.3. Package Structure

We try to follow the principles of Object Oriented programming (OOP) as close as possible.

The package in its current form has the following useful directories:

- Data structures: Code relating to the creation of various standard and information driven bars.
- Filters: Specific event filters used to place trades. An example is the CUSUM filters used to determine structural breaks.
- Features: Tools for creating useful features for machine learning algorithms, such as fractional differentiation and entropy features.
- Labeling: Containing the logic for labeling financial data, such as the Triple Barrier technique and Meta-Labeling.
- Utils: Shared code throughout the code base, such as the multiprocessing engine.
- Tests: Containing all the unit tests which can be locally run to ensure that the package works on your local python environment.

6. Building an Open Source Community

Finding users for the package, welcoming them, getting them excited about contributing to your project, providing support and structure in a virtual team, hosting meetup events to evangelize the package, and creating community guidelines are all a part of running a successful open source project. The following section outlines some of steps we have taken in setting up our open source community.

6.1. Creating an Organisation

Building on the ideas of Lopez de Prado (Lopez de Prado 2018b) and pulling inspiration from AQR (Applied Quantitative Research) we decided to setup a brand called Hudson and Thames Quantitative Research, based on the rivers where the authors reside. It would be the platform for an open source finance research laboratory where anyone could contribute to the development of tools.

Figure 9: Hudson & Thames Logo

Setting up such an organization would allow us to leverage the project in various ways. For example, we could now launch a crowdfunding campaign to fund the development of mlfinlab or pivot to a consultancy/asset management business.

By doing this we can build a product, a brand, and a client base - before the product has reached its final form.

At the time of writing the organization hosts 3 main repositories:

1. Mlfinlab (Joubert and Singh 2019d): an implementation of the techniques mentioned advances in financial machine learning.
2. Research (Joubert and Singh 2019f): A collection of Jupyter notebooks which answer the questions at the back of every chapter. These questions are designed to help the reader develop a good intuition around the techniques employed.
3. Presentations (Joubert and Singh 2019e): Slide show presentations and white papers regarding the development of products centered on machine learning in finance.

6.2. Github Open Source Guidelines

Github acts as a platform to develop software and is well known as a repository for open source projects such as Numpy, Pandas, Scikit Learn, and Tensorflow.

They also provide a number of guidelines (Github 2019) for running an open source project. In particular, they recommend the following documents which have been included in the mlfinlab repository.

- ReadMe: Introduces and explains a project.
- Code of conduct: A welcoming and inclusive document that outlines the community standards and outlines procedures for abuse.
- Contributing Guidelines: Outlines how members of the community can participate in the project and the types of desired contributions.
- License: An MIT License which only requires that the copyright and license be preserved while the package may be used commercially, modified, and distributed under different terms and without source code.
- Issue & pull request templates: Templates are provided to help contributors include information in a commit that would be relevant. For example the bug they fixed, which operating system and IDE they used.

6.3. Online Community Channels

A project of this nature is very niche and thus we expect the community to be very small. In essence, we are targeting users that use python, are familiar with machine learning, and care about finance. It is even smaller when we subset it to those users that are actively reading academic literature and exploring modern techniques.

The following are the online sources that we made use of to reach users:

6.3.1. Reddit

We had by far the most success with Reddit. Due to the subreddit structure, we are able to reach groups of people that subscribe to specific subreddits. Overall we found that technology-focused groups were very vocal and pro the package where the more fundamental/discretionary investor communities disliked it.

Our conclusion is that it would be much harder to sell the idea of machine learning to companies that weren't already aligned with systematic investing. Thus when considering the idea of consulting or selling a machine learning product, avoid firms that are focused on fundamental style strategies, and focus on companies that were already exploring the idea.

We received a high number of votes in both the algorithmic trading and machine learning communities, as shown in the figure below.

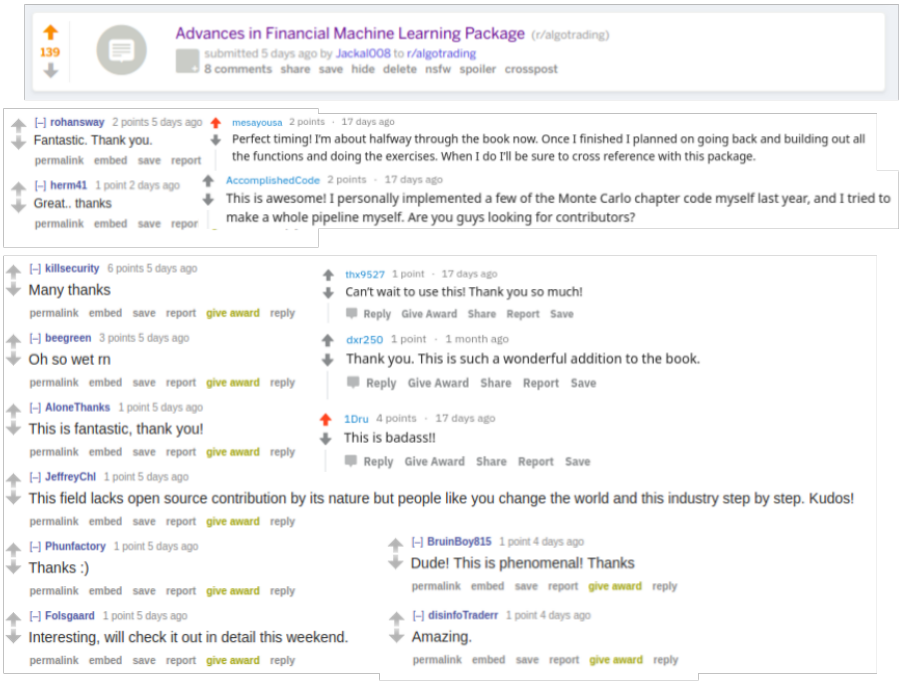

Figure 10: Reddit Comments

6.3.2. Twitter

Twitter has been great for instant responses from noteworthy persons and building somewhat of a brand name and reputation. In particular, Lopez de Prado has retweeted our research as well as liked several of our tweets regarding the package. Below are 2 noteworthy tweets:

Figure 11: Lopez de Prado retweet

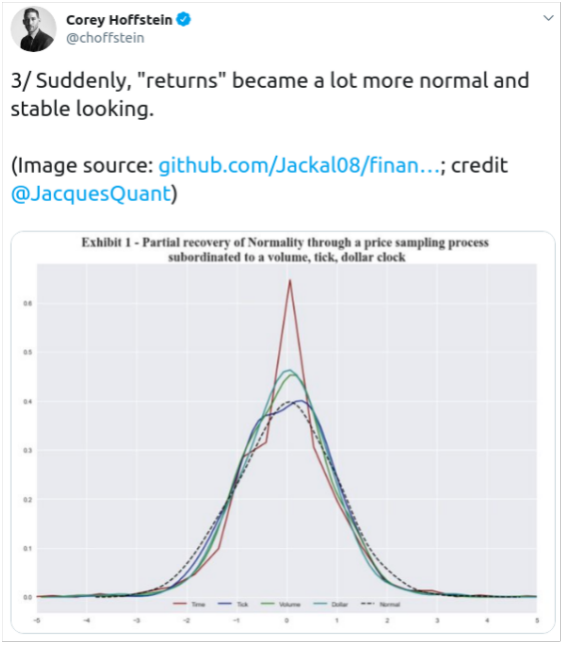

Figure 12: Corey Hoffstein retweet

6.3.3. Blog: Quantsportal.com

Jacques' personal blog has been around since 2015 and has built a small following in the quantitative finance community. It has acted as a portfolio of his work since his undergraduate days and we made use of its distribution channels to get the message out regarding package developments. In particular, the blog is linked to the well-known blog aggregator Quantocracy.com.

Currently, his website acts as a central location regarding news for mlfinlab and has also created a side project termed the Open Source Hedge Fund Project, which ties back to creating an open source financial research lab.

6.3.4. LinkedIn and Facebook

Overall the feedback from Linkedin and Facebook was disappointing. The message reaches our personal network with a few conversations starting but we didn't feel that the message carried beyond our network.

6.4. Offline Community Channels

Typically offline events refer to meetups and guest speaking opportunities in which you promote the package.

6.4.1. Guest Speaking

We have made use of the Meetup.com website and created a Machine Learning in Finance London (Joubert, Jacques. 2019a) group which at the time of writing has 250 members. Our first meetup is scheduled for the 23 May 2019 at Monticello House. We will also be guest speaking at the London Python for Trading Meetup (Jaunoo, Riyad. 2019) on the 22nd of May 2019.

6.5. Sponsorship

Thankfully we have secured sponsorship from GridGain Systems, a high-performance computing company who also hosts the In-Memory Computing Summit, which includes tickets for our members to events, fees covered for venue hire, and possible speakers on the topic of machine learning to host at our meetup.

7. Tutorials

We are in the process of creating a few tutorial notebooks which we will use at our meetup events in London. They can be found along with the other example notebooks on the Research repository(Joubert and Singh 2019f).

At the time of writing we have the following example notebooks:

1. Analysis of statistical properties for the various standard bars
2. Analysis of Imbalance bars
3. Stitching together futures contracts using the ETF trick
4. A toy example of Meta-Labeling using MNIST data
5. An end to end trend following strategy showcasing the benefits of Meta-Labeling
6. An end to end mean-reverting strategy showcasing the benefits of Meta-Labeling
7. Using Fractionally Differentiated Features

There is another developer which runs the BlackArbsCEO repository on GitHub which has notebooks covering multiple chapters. We recommend readers also view his work.

8. Final Remarks

Mlfinlab as a package will be in a constant state of development. Our vision is to implement all of the principals mentioned in the textbook and then move onto adding other recent developments in financial machine learning, as they emerge.

The success of the project will be based on user adoption of these techniques and if we can generate a source of revenue to justify the many hours spent developing it.

At the time of writing, we are a team 5 individuals all in different locations, implementing the various chapters. For some of us, this project is a platform to help us get placement at top employers, for others it a tool to help build a reputation in the industry.

8.1. Thank you

We would like to express our deepest appreciation to the Founder Igor Tulchinsky, Dean Gabriella Maiello, and Professor Tiberiu Stoica. Thank you for making WorldQuant University a possibility, for all of the support you have given us, and thank you for making the world we live in - that much better.

References

[CK06] Ludwig. B. Chincarini and Daehwan. Kim. *Quantitative Equity Portfolio Management*. McGraw Hill, 2006.

[ELO12a] David Easley, Marcos M. López De Prado, and Maureen O'Hara. "Flow toxicity and liquidity in a high-frequency world". In: *Review of Financial Studies* 25.5 (2012), pp. 1457–1493.

[ELO12b] David Easley, Marcos M. López De Prado, and Maureen O'Hara. "The Volume Clock: Insights into the High-Frequency Paradigm". In: *Journal of Portfolio Management* 39 (2012), pp. 19–29.

[FB66] Eugene F. Fama and Marshall E. Blume. "Filter rules and stock-market trading". In: *Journal of Business* 39.1 (1966), pp. 226–241.

[Git19] Github. *Github Open Source Guides*. [Online; accessed May 11, 2019]. 2019. URL: https://opensource.guide/.

[GK00] Richard. C. Grinold and Ronald. N. Kahn. *Active Portfolio Management*. McGraw Hill, 2000.

[GKX18] Shihao Gu, Bryan T. Kelly, and Dacheng Xiu. "Empirical Asset Pricing via Machine Learning". In: *National Bureau of Economic Research* w25398 (2018), pp. 1–68.

[Goo19] Google. *Machine Learning Trends*. [Online; accessed May 11, 2019]. 2019. URL: https://trends.google.com/trends/explore?date=all&q=Python%20Machine%20Learning,R%20Machine%20Learning.

[Hos81] J. R. M. Hosking. "Fractional Differencing". In: *Journal of Portfolio Management* 68.1 (1981), pp. 165–176.

[Jau19] Jaunoo, Riyad. *London Python for Trading Meet Up*. [Online; accessed May 11, 2019]. 2019. URL: https://www.meetup.com/meetup-group-XAbNZRBz/.

[Jou19a] Joubert, Jacques. *Machine Learning in Finance (London)*. [Online; accessed May 11, 2019]. 2019. URL: https://www.meetup.com/Machine-Learning-in-Finance/.

[Jou19b] Joubert, Jacques. *Sampling Techniques*. [Online; accessed May 11, 2019]. 2019. URL: https://github.com/hudson-and-thames/research/blob/master/Chapter2/2019-03-03_JJ_Sample-Techniques.ipynb.

[JS19a] Jacques. Joubert and Ashutosh. Singh. *Does Meta Labeling Add to Signal Efficacy?* [Online; accessed May 11, 2019]. 2019. URL: https://github.com/hudson-and-thames/presentations/blob/master/Does%20Meta%20Labeling%20Add%20to%20Signal%20Efficacy.pdf.

[JS19b] Jacques. Joubert and Ashutosh. Singh. *Github Group*. [Online; accessed May 11, 2019]. 2019. URL: https://github.com/hudson-and-thames.

[JS19c] Jacques. Joubert and Ashutosh. Singh. *Meta-Labeling on Trend Following*. [Online; accessed May 11, 2019]. 2019. URL: https://

github.com/hudson-and-thames/research/blob/master/Chapter3/2019-03-06_JJ_Trend-Follow-Question.ipynb.

[JS19d] Jacques. Joubert and Ashutosh. Singh. *mlfinlab Package*. [Online; accessed May 11, 2019]. 2019. URL: https://github.com/hudson-and-thames/mlfinlab.

[JS19e] Jacques. Joubert and Ashutosh. Singh. *Presentations Repo*. [Online; accessed May 11, 2019]. 2019. URL: https://github.com/hudson-and-thames/presentations.

[JS19f] Jacques. Joubert and Ashutosh. Singh. *Research Repo*. [Online; accessed May 11, 2019]. 2019. URL: https://github.com/hudson-and-thames/research.

[JS19g] Jacques. Joubert and Ashutosh. Singh. *Sampling Data*. [Online; accessed May 11, 2019]. 2019. URL: https://github.com/hudson-and-thames/research/tree/master/Sample-Data.

[JS19h] Jacques. Joubert and Ashutosh. Singh. *The Open Source Hedge Fund Project*. [Online; accessed May 11, 2019]. 2019. URL: http://www.quantsportal.com/open-source-hedge-fund/.

[Lop15] Marcos. Lopez de Prado. "Quantitative Meta-Strategies. Practical Applications." In: *SSRN* (2015), pp. 1–6.

[Lop18a] Marcos. Lopez de Prado. "Market Microstructure in the Age of Machine Learning". In: (2018), pp. 1–48.

[Lop18b] Marcos Lopez de Prado. "The 10 Reasons Most Machine Learning Funds Fail". In: *Ssrn* (2018). ISSN: 1556-5068. DOI: 10.2139/ssrn.3104816.

[Lop18c] Marcos. Lopez de Prado. "The 10 Reasons Most Machine Learning Funds Fail". In: *Journal of Portfolio Management* 44.6 (2018), pp. 120–133.

[Rie11] Ries, Eric. *The lean startup: How today's entrepreneurs use continuous innovation to create radically successful businesses.* Crown Books, 2011.

[Sin19] Singh, Ashutosh. *Fractionally Differentiated Features*. [Online; accessed May 11, 2019]. 2019. URL: https://github.com/hudson-and-thames/research/blob/master/Chapter5/Chap_5_Exercises.ipynb.

[TH00] Ané Thierry and Geman Hélyette. "Order Flow, Transaction Clock, and Normality of Asset Returns". In: 55.5 (2000), pp. 2259–2284.

[Wik19] Wikipedia, the free encyclopedia. *Precision and recall*. [Online; accessed March 18, 2019]. 2019. URL: https://en.wikipedia.org/wiki/Precision_and_recall#/media/File:Precisionrecall.svg.

Appendix A. Meta-Labeling Explained

Appendix A.0.1. Lopez de Prado's Original idea

The following section is taken directly from the de Prado and left in its original form as to make sure no errors are introduced by means of interpretation.

Advances in Financial Machine Learning, Chapter 3, page 50. Reads:

Suppose that you have a model for setting the side of the bet (long or short). You just need to learn the size of that bet, which includes the possibility of no bet at all (zero size). This is a situation that practitioners face regularly. We often know whether we want to buy or sell a product, and the only remaining question is how much money we should risk in such a bet. We do not want the ML algorithm to learn the side, just to tell us what is the appropriate size. At this point, it probably does not surprise you to hear that no book or paper has so far discussed this common problem. Thankfully, that misery ends here.

I call this problem meta-labeling because we want to build a secondary ML model that learns how to use a primary exogenous model.

The ML algorithm will be trained to decide whether to take the bet or pass, a purely binary prediction. When the predicted label is 1, we can use the probability of this secondary prediction to derive the size of the bet, where the side (sign) of the position has been set by the primary model.

How to use Meta-Labeling

Binary classification problems present a trade-off between type-I errors (false positives) and type-II errors (false negatives). In general, increasing the true positive rate of a binary classifier will tend to increase its false positive rate. The receiver operating characteristic (ROC) curve of a binary classifier measures the cost of increasing the true positive rate, in terms of accepting higher false positive rates.

The image illustrates the so-called confusion matrix. On a set of observations, there are items that exhibit a condition (positives, left rectangle), and items that do not exhibit a condition (negative, right rectangle). A binary classifier predicts that some items exhibit the condition (ellipse), where the TP area contains the true positives and the TN area contains the true negatives. This leads to two kinds of errors: false positives (FP) and false negatives (FN). Precision is the ratio between the TP area and the area in the ellipse. Recall is the ratio between the TP area and the area in the left rectangle. This notion of recall (aka true positive rate) is in the context of classification problems, the analogous to power in the context of hypothesis testing. Accuracy is the sum of the TP and TN areas divided by the overall set of items (square). In general, decreasing the FP area comes at a cost of increasing the FN area, because higher precision typically means fewer calls, hence lower recall. Still, there is some combination of precision and recall that maximizes the overall efficiency of the classifier. The

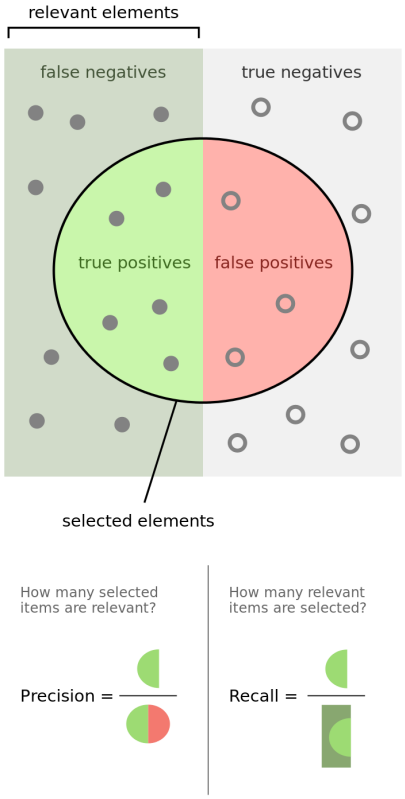

Figure A.13: Precision & Recall
Wikipedia, the free encyclopedia 2019

F1-score measures the efficiency of a classifier as the harmonic average between precision and recall.

Meta-labeling is particularly helpful when you want to achieve higher F1-scores. First, we build a model that achieves high recall, even if the precision is not particularly high. Second, we correct for the low precision by applying meta-labeling to the positives predicted by the primary model.

Meta-labeling will increase your F1-score by filtering out the false positives, where the majority of positives have already been identified by the primary model. Stated differently, the role of the secondary ML algorithm is to determine whether a positive from the primary (exogenous) model is true or false. It is not its purpose to come up with a betting opportunity. Its purpose is to determine whether we should act or pass on the opportunity that has been presented.

Additional uses of Meta-Labeling

Meta-labeling is a very powerful tool to have in your arsenal, for four additional reasons. First, ML algorithms are often criticized as black boxes.

Meta-labeling allows you to build an ML system on top of a white box (like a fundamental model founded on economic theory). This ability to transform a fundamental model into an ML model should make meta-labeling particularly useful to quantamental firms. Second, the effects of overfitting are limited when you apply meta-labeling, because ML will not decide the side of your bet, only the size. Third, by decoupling the side prediction from the size prediction, meta-labeling enables sophisticated strategy structures. For instance, consider that the features driving a rally may differ from the features driving a sell-off. In that case, you may want to develop an ML strategy exclusively for long positions, based on the buy recommendations of a primary model, and an ML strategy exclusively for short positions, based on the sell recommendations of an entirely different primary model. Fourth, achieving high accuracy on small bets and low accuracy on large bets will ruin you. As important as identifying good opportunities is to size them properly, so it makes sense to develop an ML algorithm solely focused on getting that critical decision (sizing) right. In my experience, meta-labeling ML models can deliver more robust and reliable outcomes than standard labeling models.

Appendix A.0.2. Toy Example

To illustrate the concept we made use of the MNIST data set to train a binary classifier on identifying the number 3, from a set that only includes the digits 3 and 5. The reason for this is that the number 3 looks very similar to 5 and we expect there to be some overlap in the data, i.e. the data are not linearly separable. Another reason we chose the MNIST dataset to illustrate the concept, is that MNIST is a solved problem and we can witness improvements in performance metrics with ease.

Figure A.14: Handwritten 5 and 3

Model Architecture

The following image explains the model architecture. The first step is to train a primary model (binary classification) with a high recall. Second a threshold level is determined at which the primary model has a high recall, ROC curves could be used to help determine a good level. Third the features from the first model are concatenated with the predictions from the first model, into a new feature set for the secondary model. Meta-labels are used as the target variable in the second model. Now fit the second model. Fourth the prediction from the secondary model is combined with the prediction from the primary model and only where both are true, is your final prediction true. I.e. if your primary model predicts a 3 and your secondary model says you have a high probability of the primary model being correct, is your final prediction a 3, else not 3.

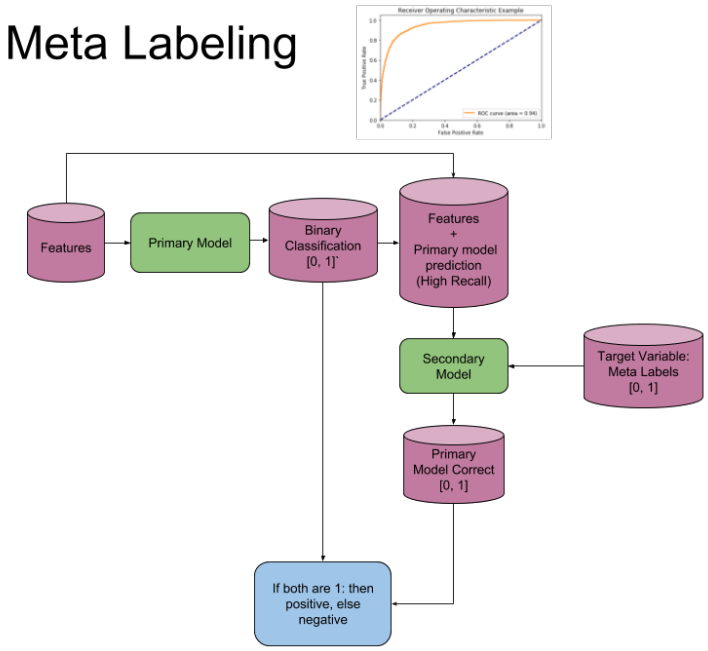

Figure A.15: Meta-Label Model Architecture

Build Primary Model with High Recall

The first step is to train a primary model (binary classification). For this we trained a logistic regression, using the keras package. The data are split into a 90% train, 10% validation. This allows us to see when we are over-fitting.

Second a threshold level is determined at which the primary model has a high recall, ROC curves could be used to help determine a good level. A high recall means that the primary model captures the majority of positive samples even if there are a large number of false positives. The meta-model will correct this by reducing the number of false positives and thus boosting all performance metrics.

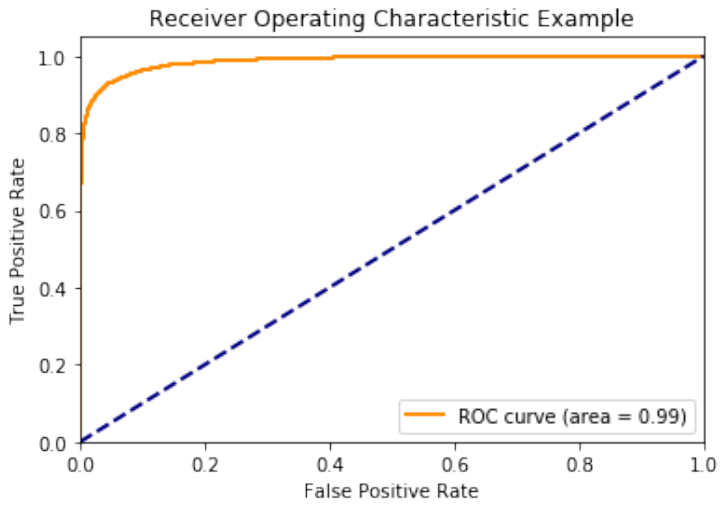

Figure A.16: Receiver Operating Characteristic (ROC) Curve

Build Meta-Model

Third the features from the first model are concatenated with the predictions from the first model, into a new feature set for the secondary model. Meta-labels are used as the target variable in the second model. Now fit the second model.

Meta-labels are defined as: If the primary model's predictions matches the actual values, then we label it as 1, else 0. In this example we said that if an observation was a true positive or true negative then label it as 1(i.e. the model is correct), else 0 (the model in incorrect). Note that because it is categorical, we have to add One Hot Encoding.

Evaluate Performance

Fourth the prediction from the secondary model is combined with the prediction from the primary model and only where both are true, is your final prediction true. e.g. if your primary model predicts a 3 and your secondary model says you have a high probability of the primary model being correct, is your final prediction a 3, else not a 3.

The section below shows the performance of the primary model vs the performance of using Meta-labeling, on out-of-sample data. Notice how the performance metrics improve.

```
Base Model Metrics:
              precision    recall  f1-score   support

       False       0.95      0.94      0.94       892
        True       0.94      0.96      0.95      1010

   micro avg       0.95      0.95      0.95      1902
   macro avg       0.95      0.95      0.95      1902
weighted avg       0.95      0.95      0.95      1902

Confusion Matrix
[[700 192]
 [ 11 999]]
Accuracy:  0.8933

Meta Label Metrics:
              precision    recall  f1-score   support

       False       0.95      0.96      0.95       892
        True       0.96      0.95      0.96      1010

   micro avg       0.96      0.96      0.96      1902
   macro avg       0.96      0.96      0.96      1902
weighted avg       0.96      0.96      0.96      1902

Confusion Matrix
[[857  35]
 [ 47 963]]
Accuracy:  0.9569
```

Figure A.17: Meta-Labeling Performance Metrics

We can see that in the confusion matrix, that the false positives from the primary model, are now being correctly identified as true negatives with the help of meta-labeling. This leads to a boost in performance metrics. Meta-labeling works as advertised!