Online Ensemble of Models for Optimal Predictive Performance with Applications to Sector Rotation Strategy

Paweł Polak

Department of Applied Mathematics & Statistics (AMS)
Institute for Advanced Computational Science (IACS)
Stony Brook University (SBU)

a joint work with Jiaju Miao (AMS, SBU)
Presented at the joint seminar of International Association for Quantitative Finance IAQF & Thalesians Ltd

January 09, 2024

Outline

Summary of New Online Ensemble Algorithm

- **Online** : sequential update of the weights of individual models.
- **Model-Agnostic** : relies solely on recent forecasting performance for privacy preservation.
- Facilitates **integration of information from many models and datasets** developed by different groups, e.g., within a large organization or a web-scale social network, e.g., micropredictions as in Cotton (2022).
- Overcomes computational inefficiencies and challenges posed by serial correlation or potential non-stationarities in the time-series data.

Sector Rotations During COVID-19

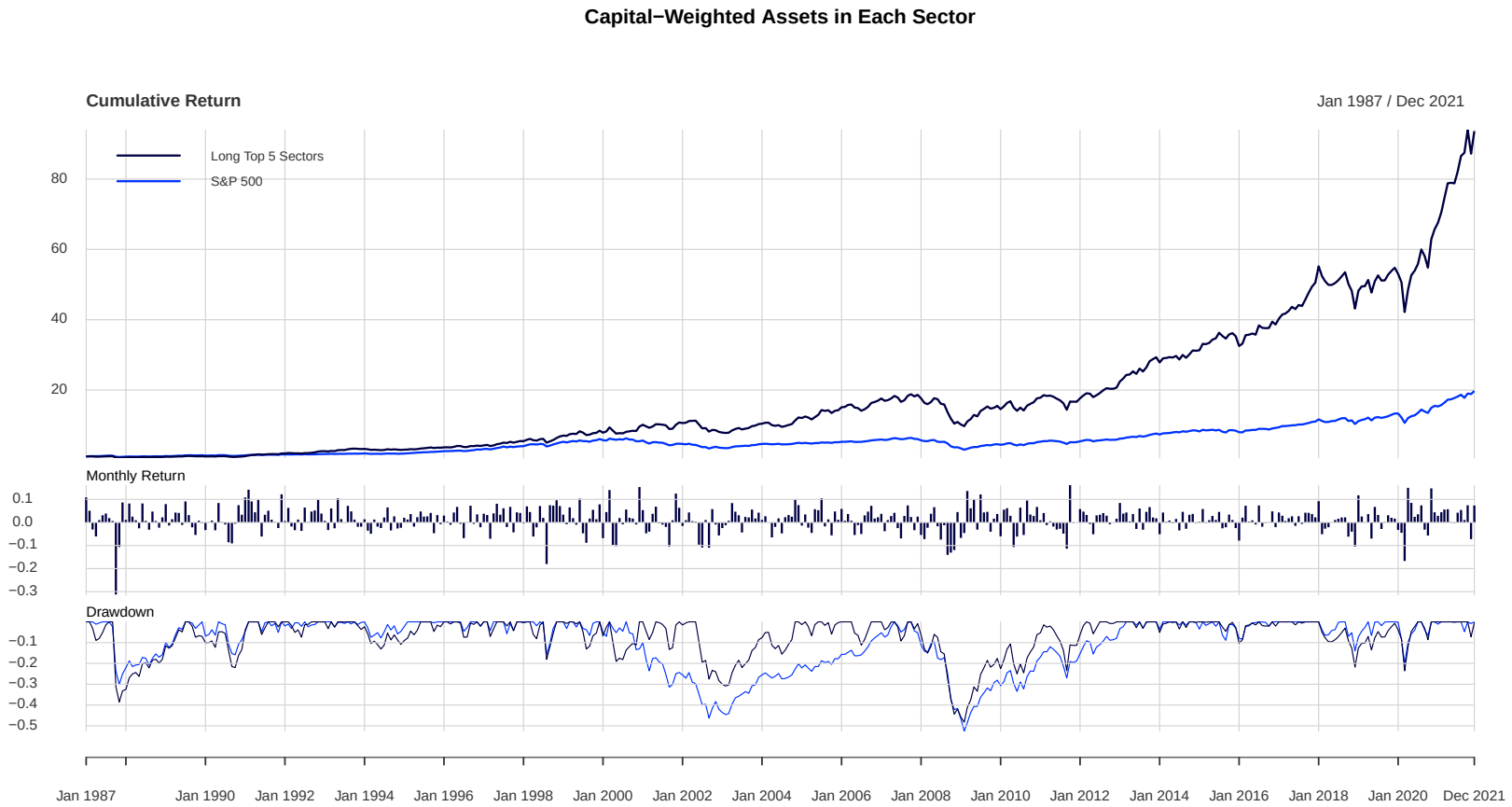
- **Technology Surge :** Growth in software, cloud computing, and online services.
- **Healthcare & Biotech :** Vaccine and treatment.
- **Travel & Hospitality Drop :** Lockdowns & Restrictions
- **E-commerce Boom :** Accelerated growth in online retail.
- **Energy Sector Volatility :** Initial decline, then recovery.
- **Financial Sector Challenges :** Economic uncertainty.
- **Real Estate Shifts :** Commercial challenges, residential and industrial growth.
- **Consumer Staples vs. Discretionary :** Stability in essentials, variability in non-essentials.
- **Home Improvement Growth :** Increased demand due to more time spent at home.
- **Telecommunications and Media :** Growth in streaming and digital entertainment

Application to Sector Rotation

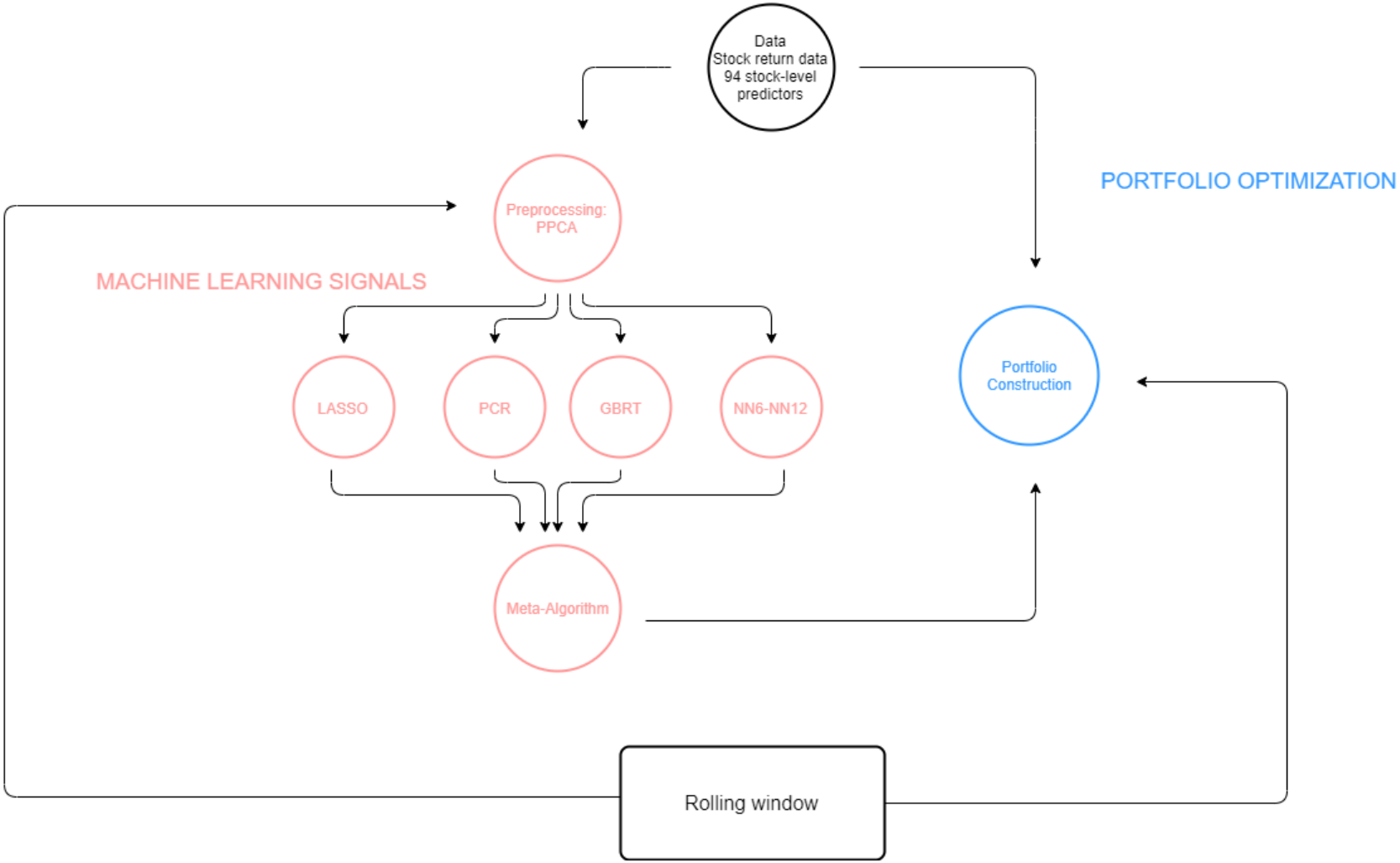
- Sector Rotation is a strategy used by investors whereby they hold on an overweight position in strong sectors and underweight position in weaker sectors.
- Gu, S., Kelly, B., and Xiu, D. RFS (2020) [1]show that Machine Learning methods (ML) lead to large economic gains to investors in terms of ***Asset Risk Premiums*** and higher ***out-of-sample Sharpe Ratios*** of the corresponding equities portfolio.
- We show that even higher gains in terms of ***Sector Risk Premiums*** and higher ***out-of-sample Sharpe Ratios*** to investors can be achieved when using ML to ***Sector Returns*** and ***Sector Rotation Strategy***.

1. "Empirical Asset Pricing via Machine Learning", Review of Financial Studies, Vol. 33, Issue 5, (2020), 2223-2273.

Preview of the Performance of the Online Ensemble



Flowchart of the *"Online Ensemble of Models"* Paper



Measuring Sector Risk Premia

- **Sector breakdown :** We break down the assets into industry level based on the first two digits of Standard Industrial Classification (SIC) codes and then present the data in a two-dimensional layout (i.e., time and sector). We consider $N = 60$ sectors and denote the excess return of sector i over one period (month) from t to $t+1$ by $r_{i,t+1}$, where $i \in \{1, \ldots, N\}$ and $t \in \{1, \ldots, T\}$.
- **Additive prediction error model :**

$$r_{i,t+1} = E_t(r_{i,t+1}) + \epsilon_{i,t+1}$$

where

$$E_t(r_{i,t+1}) = g(z_{i,t}).$$

Measuring Sector Risk Premia

- **Sector breakdown :** We break down the assets into industry level based on the first two digits of Standard Industrial Classification (SIC) codes and then present the data in a two-dimensional layout (i.e., time and sector). We consider $N = 60$ sectors and denote the excess return of sector i over one period (month) from t to $t+1$ by $r_{i,t+1}$, where $i \in \{1, \ldots, N\}$ and $t \in \{1, \ldots, T\}$.
- **Additive prediction error model :**

$$r_{i,t+1} = E_t(r_{i,t+1}) + \epsilon_{i,t+1}$$

where

$$E_t(r_{i,t+1}) = g(z_{i,t}).$$

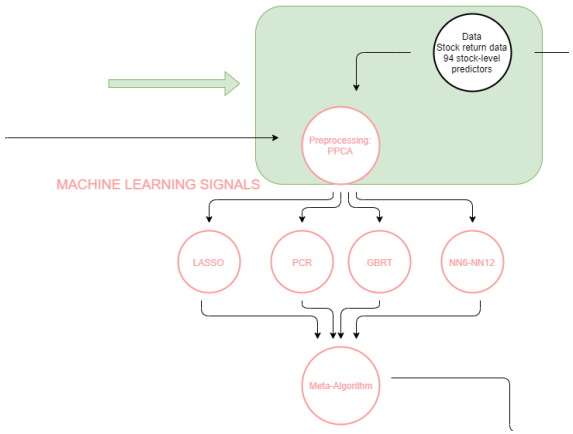
Preprocessing Approach for Sector Returns

We obtain the treasure bill rate to proxy for the risk-free rate from which we calculate sector excess returns. We have the following two versions of sector returns :

$$r_{i,t+1} = \begin{cases} \dfrac{1}{n_i} \sum_{j=1}^{n_i} r_{i,t+1}^{j} & \text{equally-weighted} \\ \dfrac{\sum_{j=1}^{n_i} c_{i,t}^{j} r_{i,t+1}^{j}}{\sum_{j=1}^{n_i} c_{i,t}^{j}} & \text{capital-weighted} \end{cases}$$

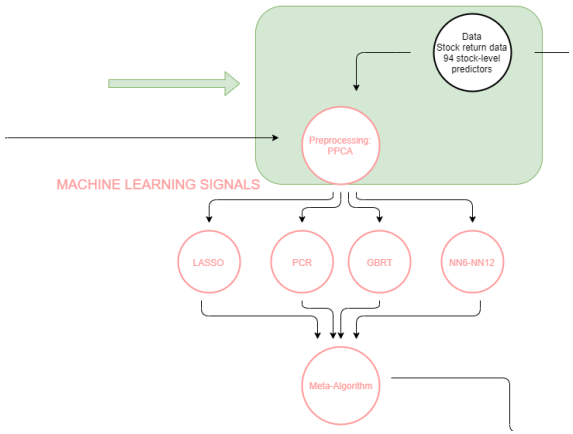
where $r_{i,t+1}^{j}$ is the return value for company j which belongs to sector i in time $t+1$, $c_{i,t}^{j}$ is the corresponding market cap of that company, and n_i is the total number of the companies in the sector i.

Probabilistic Principal Component Analysis(PPCA)



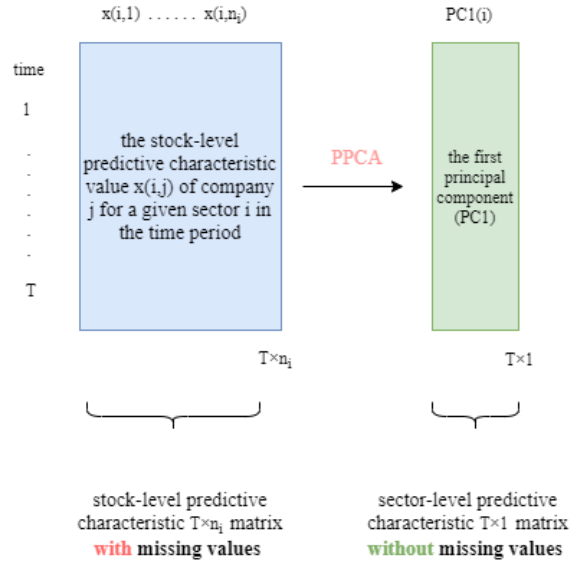
- **PCA :** rotates the original sample data into the orthogonal components.
- **PPCA :** It is PCA combined with an expectation–maximization (EM) algorithm to deal with the missing values in the data in the inputs.

Probabilistic Principal Component Analysis(PPCA)



- **PCA :** rotates the original sample data into the orthogonal components.
- **PPCA :** It is PCA combined with an expectation–maximization (EM) algorithm to deal with the missing values in the data in the inputs.

Preprocessing Approach Algorithm for Predictor Variables



Algorithm Preprocessing Approach Algorithm

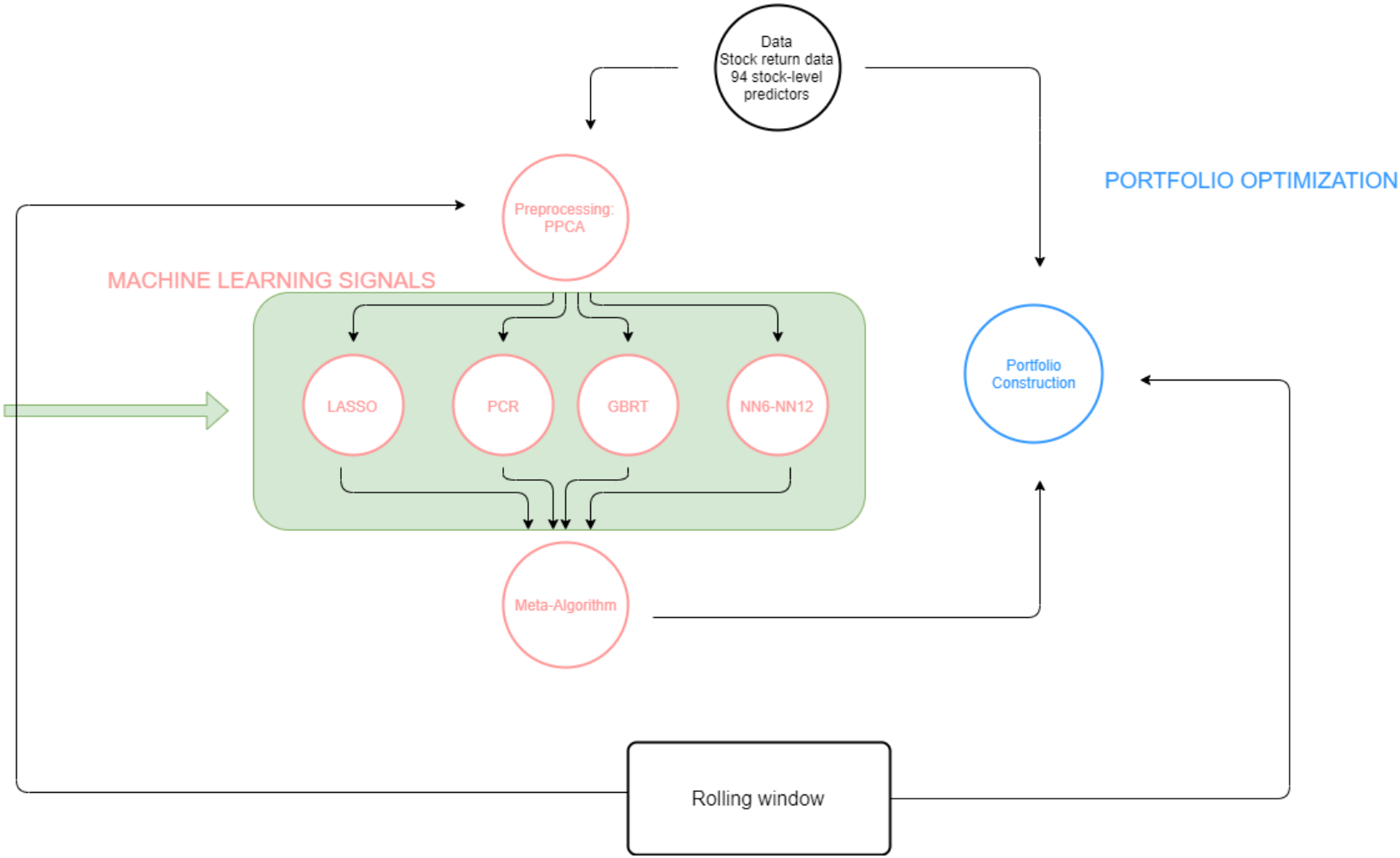
Input: n_i, the total number of the companies in the sector i; K, the number of total stock-level predictive characteristics; the kth stock-level predictive characteristic value $x_{i,t}^{j(k)}$ of company j for a given sector i in time t, where $i \in \{1,\ldots,N\}$, N number of sectors; $j \in \{1,\ldots,n_i\}$; $k \in \{1,\ldots,K\}$ and $t \in \{1,\ldots,T\}$.

Output: $z_{i,t}$, the explanatory variables for a given sector in time t.

1: **for** $i = 1$ to N **do**
2: **for** $k = 1$ to K in $t \in \{1,\ldots,T\}$ **do**
3: `PPCA to` $[x_{i,t}^{j(k)}]_{j=1,\ldots,n_i}^{t=1,\ldots,T} \in \mathbb{R}^{T\times n_i}$
4: `Store the first principal component (PC1):` $[f_{i,t}^{(k)}]_{t=1,\ldots,T} \in \mathbb{R}^{T\times 1}$
5: **end for**
6: **end for**
7: $[z_i] = [f_{i,t}^{(1)},\ldots,f_{i,t}^{(K)}]_{k=1,\ldots,K}^{t=1,\ldots,T} \in \mathbb{R}^{T\times K}$
8: **return** $[z_i]$
9: **end**

- $f_i^{(k)}$ (i.e., $PC1(i)$ - the first Principal Component) is a univariate summary of the variation of a given stock-level characteristic (k) in a sector i.
- $[z_i] = [f_{i,t}^{(1)},\ldots,f_{i,t}^{(K)}]_{k=1,\ldots,K}^{t=1,\ldots,T} \in \mathbb{R}^{T\times K}$ sector specific predictors

Machine Learning Signals



LASSO

Simple linear model is suitable for low dimension predictors problems. The linear model becomes unstable and shows poor predictive performance as the number of predictors increases or in the situation when the actual relation is sparse or when $T < P$.
LASSO regression is an L1 penalized model where we simply add the L1 norm of the weights to our least-squares objective function :

$$\mathcal{L}(\theta; \lambda) = \frac{1}{T} \sum_{t=1}^{T} \left(r_{i,t+1} - g(z_{i,t}; \theta) \right)^2 + \lambda \sum_{j=1}^{P} |\theta_j|.$$

Principal Components Regression(PCR)

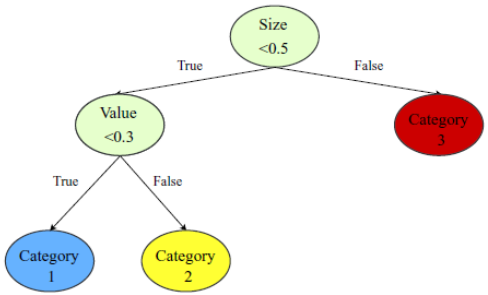
PCR is an alternative to multiple linear regression (MLR) and has many advantages over MLR. The PCR can perform regression especially when the predictor variables are highly correlated. Mathematically, PCR is the two-step process as following :

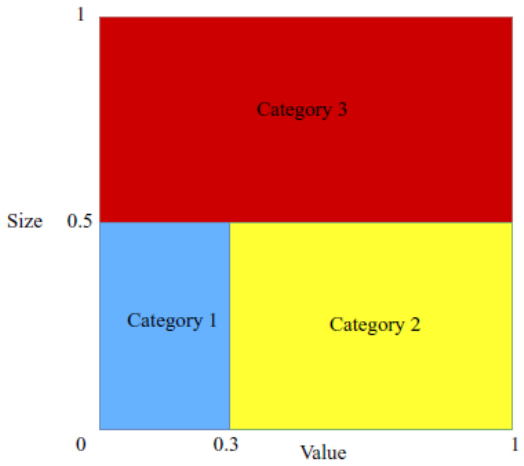
$$\underbrace{K}_{T\times M} = \underbrace{Z}_{T\times P}\underbrace{\Phi}_{P\times M},$$

$$\underbrace{\hat{R}}_{T\times 1} = \underbrace{K}_{T\times M}\underbrace{\theta_M}_{M\times 1},$$

where Z is the $T \times P$ matrix of stacked predictor variables $z_{i,t}$ for a given sector i, R is the $T \times 1$ vector of $r_{i,t+1}$ and θ_M is the $M \times 1$ coefficient vector, where $M \ll P$. The columns in K represents the scores from PCA are orthogonal to each other, obtaining independence for the next regression step.

Gradient Boosting and Random Forests





The partition of the tree structure is built by recursively splitting the predictor space into rectangular regions. A typical regression tree follow a model of form denoted

$$\hat{f}(z_{i,t};\theta,M)=\sum_{m=1}^{M}\theta_m\cdot\mathbb{1}_{\{z_{i,t}\in R_m\}},$$

where R_m represents one of the total M partitions of feature space. Each partition is a product of up to M indication functions. θ_m is denoted as the sample mean of each corresponding partition.

Gradient Boosting and Random Forests

To generate a random forest, we calculate the $\hat{f}(z_{i,t})$ for each separate bootstrapped training sample and get the average estimation by

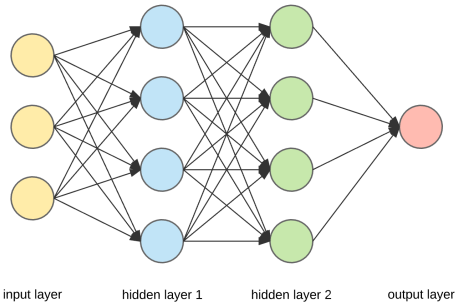
$$\hat{f}_{avg}(z_{i,t}) = \frac{1}{B}\sum_{t=1}^{B}\hat{f}^b(z_{i,t}).$$

In gradient boosting, decision trees are added in sequence :

$$\hat{f}(z_{i,t}) = \sum_{t=1}^{B}\lambda\hat{f}^b(z_{i,t}).$$

The shrinkage parameter λ controls the learning rate. Small value λ may require a very large number B to achieve good performance.

Neural Networks



- The feed-forward networks refers to the networks where connections between the neurons do not form a cycle.
- A typical network includes three components : input layer, hidden layer and output layer.
- The activation function of a node defines the output of that node given an input or set of inputs. The most commonly used form is Rectifier linear unit, defined as

$$ReLU(x) = max(0, x).$$

Neural Networks

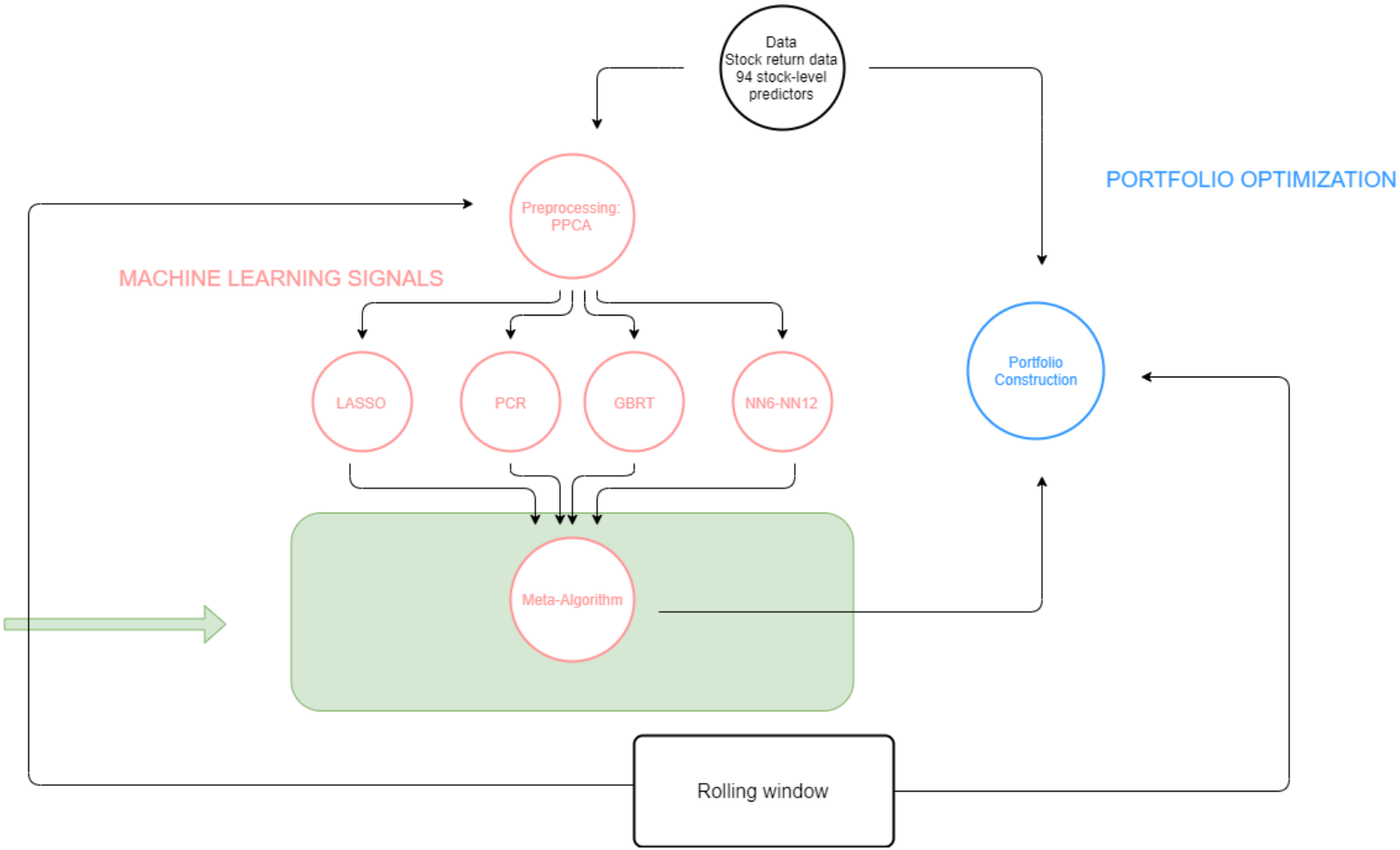
The output of the neuron in a layer is the activation function of a weighted sum of the neuron's input in the previous layer. Let us denote $K^{(l)}$ as the number of neurons in each layer and $x_k^{(l)}$ as the output of neuron k in layer $l = 1, \ldots, L$. Hence, the output formula for each neuron in layer $l > 0$ is given by

$$x_k^{(l)} = Activation(x^{(l-1)'} w_k^{(l-1)}),$$

where w_k is the weight matrix on connection from layer $l-1$ to l. Then the final output of our neural network can be written as

$$g(z; w) = x^{(L-1)'} w_k^{(L-1)}.$$

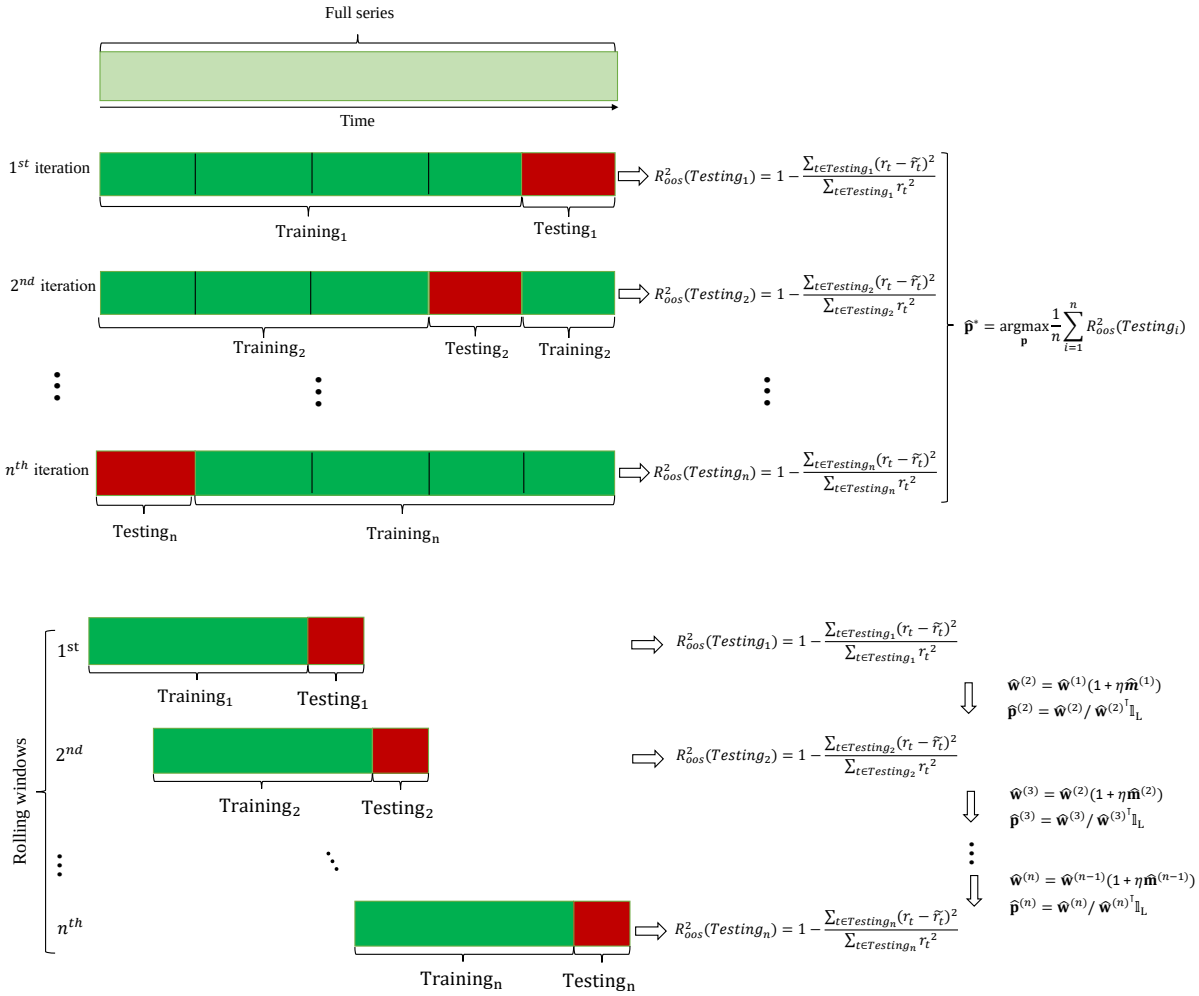
Multiplicative Weights Update Method (MWUM)



Multiplicative Weights Update Method (MWUM)

- We build an online meta-algorithm that for each sector optimally combines predictions from different ML models (experts) based on MWUM.
- We assign initial weights to the experts (we set them equal), and update these weights multiplicatively and iteratively according to the feedback of how well each model performed after each rolling window : reducing it in case of poor performance, and increasing it otherwise.
- The algorithm gradually learns what combination of ML methods to pick for which sector.

CV vs. Online Learning with MWUM



Multiplicative Weights Update Method (MWUM)

Algorithm Meta Multiplicative Weights Algorithm

Input: $\{\widehat{\mathbf{r}}_t\}_{t=1,\ldots,T}$, where $\widehat{\mathbf{r}}_t = [\widehat{r}_{t,1}, \ldots, \widehat{r}_{t,L}]^\top$ vector of the predicted returns (for a given sector) from L different prediction models at rolling window t ; η, the learning rate meta-parameter to update the gain function.

Output: $\{\widetilde{r}_t\}_{t=1,\ldots,T}$, with $\widehat{\mathbf{r}}_t^\top \mathbf{p}^{(t)}$ the weighted combination of predictions of all the models at time t, where $\mathbf{p}^{(t)} = \left[p_1^{(t)}, \ldots, p_L^{(t)}\right]^\top$; and $p_l^{(t)}$ the weight of each model $l = 1, \ldots, L$ at time t (determined only using the past performance).

Initialization : Fix the learning rate $\eta \leq 1/2$. With each decision, associate the weight at time 1 : $\mathbf{w}^{(1)} = \left[w_1^{(1)}, \ldots, w_L^{(1)}\right]$, and $w_l^{(1)} = 1$

for $t = 2, 3, \ldots, T$ **do**

1. Choose decision l with probability proportional to its weight $w_l^{(t)}$. I.e., use the distribution over decisions

$$\mathbf{p}^{(t)} = \left[w_1^{(t)}/\Phi^{(t)}, \ldots, w_L^{(t)}/\Phi^{(t)}\right]^\top \text{ where } \Phi^{(t)} = \sum_{l=1}^{L} w_l^{(t)}.$$

2. Observe the gains of the decisions $\mathbf{m}^{(t)} = \left[m_1^{(t)}, \ldots, m_L^{(t)}\right]^\top$ (see slide 23 for the optimal choice of the gain function)
3. Update the ensemble weights : $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)}(1 + \eta \mathbf{m}^{(t)})$.

end for

Multiplicative Weights Update Method (MWUM)

- The MWUM algorithm is most commonly used for decision making and prediction, and also widely deployed in game theory and algorithm design.
- It was discovered repeatedly in machine learning (AdaBoost, Winnow, Hedge), optimization (solving linear programs), theoretical computer science (devising fast algorithm for LPs and SDPs), and game theory.

Theorem (The theoretical guarantee of the algorithm)

Assume that all costs $m_l^{(t)} \in [-1, 1]$ *in* $\mathbf{m}^{(t)} = \left[m_1^{(t)}, \ldots, m_L^{(t)}\right]^\top$ *and* $\eta \leq 1/2$. *Then the MWUM guarantees that after* T *rounds, for any distribution* $\mathbf{p}$ *on the decisions, we have*

$$\sum_{t=1}^{T} \mathbf{m}^{(t)\top} \mathbf{p}^{(t)} \leq \sum_{t=1}^{T} (\mathbf{m}^{(t)} + \eta |\mathbf{m}^{(t)}|)^\top \mathbf{p} + \frac{\log L}{\eta}$$

- The vector $\mathbf{p}$ is arbitrary in this theorem. Hence, by setting it to the optimal strategy we can bound the average expected cost of the MWUM in terms of the total cost of the best strategy.
- We develop its version for the purpose of optimal Risk Premia prediction.

Multiplicative Weights Update Method (MWUM)

We define our gain function as

$$\mathbf{m}^{(t)}(\mathbf{p}^{(t)}) = \underbrace{(\mathbb{1}_L - \frac{(r_t \mathbb{1}_L - \widehat{\mathbf{r}}_t)^2}{\widehat{\sigma}_t^2})}_{\text{exploitation}} \underbrace{- \frac{\widetilde{R}_t \mathbb{1}_L}{\widehat{\sigma}_t^2}}_{\text{exploration}},$$

where

- $r_{i,t}$ is the out-of-sample return of sector i at rolling window t
- $\widehat{\mathbf{r}}_t = \left[\widehat{r}_{t,1}, \ldots, \widehat{r}_{t,L}\right]^\top$ vector of the predicted returns (for a given sector) from L different prediction models at rolling window t
- $\widetilde{R}_t = [\xi_{i,j,t}]_{i=1,\ldots,L}^{j=1,\ldots,L}$ is an $L \times L$ matrix with

$$\xi_{i,j,t} = \begin{cases} \widehat{r}_{t,i} \widehat{r}_{t,j} p_j^{(t)} & \text{if } i \neq j \\ -\widehat{r}_{t,i}^2 (1 - p_i^{(t)}) & \text{otherwise,} \end{cases}$$

- $\widehat{\sigma}_t^2 = \frac{1}{T} \sum_{s^{(t)}=1}^{T} r_{s^{(t)}}^2$ is the bootstrap estimate of the second moment of the sector returns based on the past t observations, and
- $\mathbb{1} = [1, \ldots, 1]^\top \in \mathbb{R}^L$.

Note that this gain function promotes models that start to perform well and predict differently than already well performing models. Namely,...

Multiplicative Weights Update Method (MWUM)

$$\mathbf{m}^{(t)}(\mathbf{p}^{(t)}) = \underbrace{(\mathbb{1}_L - \frac{(r_t \mathbb{1}_L - \widehat{\mathbf{r}}_t)^2}{\widehat{\sigma}_t^2})}_{\text{exploitation}} \underbrace{- \frac{\widetilde{R}_t \mathbb{1}_L}{\widehat{\sigma}_t^2}}_{\text{exploration}} ,$$

the first term **exploits** the well-performing models :

- it is responsible for increasing (penalizing) the weight of the models with the previous rolling-window forecast error smaller (larger) than the second moment of the returns ;
- i.e., if the model forecasts are better than naive $\widehat{r}_t = 0$ forecast, then the first term is positive.

Multiplicative Weights Update Method (MWUM)

$$\mathbf{m}^{(t)}(\mathbf{p}^{(t)}) = \underbrace{(\mathbb{1}_L - \frac{(r_t \mathbb{1}_L - \widehat{\mathbf{r}}_t)^2}{\widehat{\sigma}_t^2})}_{\text{exploitation}} \underbrace{- \frac{\widetilde{R}_t \mathbb{1}_L}{\widehat{\sigma}_t^2}}_{\text{exploration}}, \text{ where } \widetilde{R}_t = [\xi_{i,j,t}]_{i=1,\ldots,L}^{j=1,\ldots,L}, \text{ with}$$

$$\xi_{i,j,t} = \begin{cases} \widehat{r}_{t,i}\widehat{r}_{t,j}p_j^{(t)} & \text{if } i \neq j \\ -\widehat{r}_{t,i}^2(1 - p_i^{(t)}) & \text{otherwise,} \end{cases}$$

and the second term captures **exploration**. It can be interpreted as following :

- in case of a model that has already a high ensemble weight ($p_i^{(t)} \approx 1$), this term is negligible because $\mathbb{1}_L^\top \mathbf{p}^{(t)} = 1$ implies $p_j^{(t)} \approx 0$ for all $j \neq i$;
- in case of a model that has a low ensemble weight ($p_i^{(t)} \approx 0$), if it predicts in the same direction as some well-performing model ($\widehat{r}_{t,i}\widehat{r}_{t,j} > 0$ for j such that $p_j^{(t)} \approx 1$), it is penalized further because there is already a well-performing model that predicts the same returns. But if it predicts in the opposite direction, than a well-performing model ($\widehat{r}_{t,i}\widehat{r}_{t,j} < 0$ for j such that $p_j^{(t)} \approx 1$), then this term increases the weight of model i (as long as the prediction error from the first term is small).

Multiplicative Weights Update Method (MWUM)

The following theorem shows that the average gains based on the proposed gain function converges to $R^2_{oos}(\tau)$, i.e., the Risk Premia of the corresponding ensemble defined by a sequence of distributions $\left\{\mathbf{p}^{(t)}\right\}_t$

Lemma (1)

For the gain function defined above, when $\tau \to \infty$, for any sequence of distributions $\left\{\mathbf{p}^{(t)}\right\}_{t=1}^{\tau}$ on the decisions, we have

$$\left| R^2_{oos}(\tau) - \frac{1}{\tau}\sum_{t=1}^{\tau} \mathbf{m}^{(t)} (\mathbf{p}^{(t)})^\top \mathbf{p}^{(t)} \right| \xrightarrow{a.s.} 0 \quad as \quad \tau \to \infty$$

where $R^2_{oos}(\tau) = 1 - \frac{\sum_{t=1}^{\tau}(r_t - \widetilde{r}_t)^2}{\sum_{t=1}^{\tau} r_t^2}$ *and* $\widetilde{r}_t = \widehat{r}_t^\top \mathbf{p}^{(t)}$.

Multiplicative Weights Update Method (MWUM)

With the convergence result and the theorem for the MWUM algorithm, we get

Theorem (2)

Assume that all gains $m_l^{(t)} \in [-1, 1]$ *in* $\mathbf{m}^{(t)} = \left[m_1^{(t)}, \ldots, m_L^{(t)}\right]^\top$ *and* $\eta \leq 1/2$. *Then, for any* $\mathbf{p}^*$ *such that* $\mathbf{p}^{(t)} \to \mathbf{p}^*$, *we have*

$$\lim_{\tau\to\infty}\left\{R_{oos}^{2*}(\tau) - R_{oos}^{2}(\tau)\right\} \leq \lim_{\tau\to\infty}\left\{\eta\frac{1}{\tau}\sum_{t=1}^{\tau}\left|\mathbf{m}^{(t)}(\mathbf{p}^{(t)})\right|^\top \mathbf{p}^* + \frac{\log L}{\tau\eta} - \frac{1}{\tau}\sum_{t=1}^{\tau}(\mathbf{p}^{(t)} - \mathbf{p}^*)^\top \widehat{\mathbf{r}}_t\widehat{\mathbf{r}}_t^\top \mathbf{p}^*\right\}$$

where $R_{oos}^{2*}(\tau)$ *is a function of* $\mathbf{p}^*$.

- for sufficiently large τ, the risk premia $R_{oos}^2(\tau)$ of the resulting meta-algorithm are not much worse than the risk premia of the limiting ensemble.
- if we confirm after τ iterations that our $\mathbf{p}^{(\tau)}$ is close to the $\mathbf{p}^*$ which would have been best in all the previous iterations, then we know that our MWUM algorithm achieved close to optimal $R_{oos}^2(\tau)$.
- our algorithm allows for online updating and requires one additional verification step at the end of the simulation (check if $\mathbf{p}^{(\tau)}$ is close to $\mathbf{p}^*$).

Multiplicative Weights Update Method (MWUM)

Corollary (2.1)

Under the assumptions of Theorem (2), and for any $\mathbf{p}^*$*, we have*

$$\lim_{\tau\to\infty}\left\{R^{2*}_{oos}(\tau)-R^{2}_{oos}(\tau)\right\}\leq\lim_{\tau\to\infty}\left\{\eta\left\|\frac{1}{\tau}\sum_{t=1}^{\tau}\left|\mathbf{m}^{(t)}(\mathbf{p}^{(t)})\right|\right\|_2+\frac{\log L}{\tau\eta}-\frac{1}{\tau}\sum_{t=1}^{\tau}(\mathbf{p}^{(t)}-\mathbf{p}^*)^\top\widehat{\mathbf{r}}_t\widehat{\mathbf{r}}_t^\top\mathbf{p}^*\right\}$$

where $R^{2*}_{oos}(\tau)$ *is a function of* $\mathbf{p}^*$*.*

Corollary (2.2)

The optimal learning rate for the MWUM algorithm is given by

$$\eta^*=\sqrt{\frac{\log L}{\left\|\frac{1}{\tau}\sum_{t=1}^{\tau}\left|\mathbf{m}^{(t)}(\mathbf{p}^{(t)})\right|\right\|_2}}$$

and the resulting regret bound is

$$\lim_{\tau\to\infty}\left\{R^{2*}_{oos}(\tau)-R^{2}_{oos}(\tau)\right\}\leq\lim_{\tau\to\infty}\left\{\sqrt{\frac{\log L}{\tau}}\left\|\frac{1}{\tau}\sum_{t=1}^{\tau}\left|\mathbf{m}^{(t)}(\mathbf{p}^{(t)})\right|\right\|_2^{1/2}-\frac{1}{\tau}\sum_{t=1}^{\tau}(\mathbf{p}^{(t)}-\mathbf{p}^*)^\top\widehat{\mathbf{r}}_t\widehat{\mathbf{r}}_t^\top\mathbf{p}^*\right\}.$$

Multiplicative Weights Update Method (MWUM)

In practice we are interested in an ensemble which maximizes R^2_{oos}. We can obtain the estimate of it from the performance in the previous τ rolling windows as

$$\widehat{\mathbf{p}}^* = \arg \max_{\mathbf{p}:\mathbb{1}^\top \mathbf{p}=1} \left\{ 1 - \frac{\frac{1}{\tau}\sum_{t=1}^{\tau}(r_t - \widehat{\mathbf{r}}_t^\top \mathbf{p})^2}{\frac{1}{\tau}\sum_{t=1}^{\tau} r_t^2} \right\}. \tag{1}$$

Lemma (2.2)

The solution of (1) is given by

$$\widehat{\mathbf{p}}^* = \left(\frac{1}{\tau}\sum_{t=1}^{\tau}\widehat{\mathbf{r}}_t\widehat{\mathbf{r}}_t^\top\right)^{-1}\left(\frac{1}{\tau}\sum_{t=1}^{\tau}\widehat{\mathbf{r}}_t r_t - \frac{\mathbb{1}_L(\frac{1}{\tau}\sum_{t=1}^{\tau}\widehat{\mathbf{r}}_t\widehat{\mathbf{r}}_t^\top)^{-1}\frac{1}{\tau}\sum_{t=1}^{\tau}\widehat{\mathbf{r}}_t r_t - 1}{\mathbb{1}_L^\top(\frac{1}{\tau}\sum_{t=1}^{\tau}\widehat{\mathbf{r}}_t\widehat{\mathbf{r}}_t^\top)^{-1}\mathbb{1}_L}\mathbb{1}_L\right)$$

and

$$\left\|\widehat{\mathbf{p}}^*\right\|_2 \leq \min(1, \delta_\tau) \tag{2}$$

where

$$\delta_\tau = \frac{1}{\lambda_{min}}\left\|\frac{1}{\tau}\sum_{t=1}^{\tau}\widehat{\mathbf{r}}_t r_t - \frac{\mathbb{1}_L(\frac{1}{\tau}\sum_{t=1}^{\tau}\widehat{\mathbf{r}}_t\widehat{\mathbf{r}}_t^\top)^{-1}\frac{1}{\tau}\sum_{t=1}^{\tau}\widehat{\mathbf{r}}_t r_t - 1}{\mathbb{1}_L^\top(\frac{1}{\tau}\sum_{t=1}^{\tau}\widehat{\mathbf{r}}_t\widehat{\mathbf{r}}_t^\top)^{-1}\mathbb{1}_L}\mathbb{1}_L\right\|_2 \tag{3}$$

and λ_{min} is the smallest eigenvalue of the matrix $\frac{1}{\tau}\sum_{t=1}^{\tau}\widehat{\mathbf{r}}_t\widehat{\mathbf{r}}_t^\top$.

- The bound in (2) allows us to modify the results in Corollary (2.1) and Corollary (2.2) to derive tighter regret bounds and better learning rates that are optimal for the problem of finding the ensemble which maximizes R^2_{oos}.

Multiplicative Weights Update Method (MWUM)

In particular, using the bound (2), and Cauchy-Schwarz inequality applied to the upper bound in Theorem (2), we get

Corollary (2.3)

Under the assumptions of Theorem (2), and for $\mathbf{p}^*$ *satisfying* (2), *we have*

$$\lim_{\tau\to\infty}\left\{R^{2*}_{oos}(\tau)-R^{2}_{oos}(\tau)\right\}\leq\lim_{\tau\to\infty}\left\{\eta\min(1,\delta_\tau)\left\|\frac{1}{\tau}\sum_{t=1}^{\tau}\left|\mathbf{m}^{(t)}(\mathbf{p}^{(t)})\right|\right\|_2+\frac{\log L}{\tau\eta}\right.$$
$$\left.+\frac{1}{\tau}\sum_{t=1}^{\tau}(\mathbf{p}^*-\mathbf{p}^{(t)})^\top\widehat{\mathbf{r}}_t\widehat{\mathbf{r}}_t^\top\mathbf{p}^*\right\}$$

where δ_τ *is given in* (3), *and* $R^{2*}_{oos}(\tau)$ *is a function of the optimal* $\mathbf{p}^*$.

The important difference relative to the result in Corollary (2.1) is that the new bound is specific for the $R^{2*}_{oos}(\tau)$ of the optimal ensemble.

Multiplicative Weights Update Method (MWUM)

Corollary (2.4)

The optimal learning rate for the MWUM algorithm is given by

$$\eta^* = \sqrt{\frac{\log L}{\left\| \frac{1}{\tau} \sum_{t=1}^{\tau} \left| \mathbf{m}^{(t)}(\mathbf{p}^{(t)}) \right| \right\|_2}} \min(1, \delta_\tau)$$ *and the resulting regret bound is*

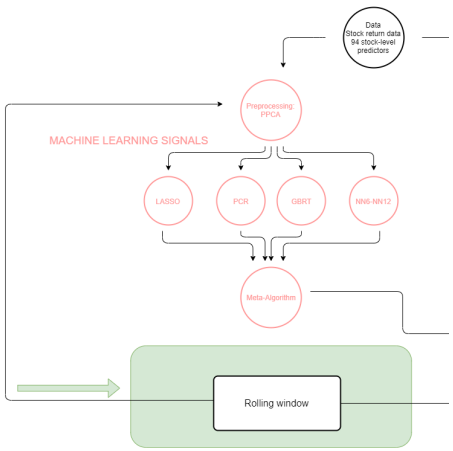
$$\lim_{\tau \to \infty} \left\{ R^{2*}_{oos}(\tau) - R^2_{oos}(\tau) \right\} \leq \lim_{\tau \to \infty} \left\{ \min(1, \delta_\tau) \frac{\log L}{\tau} \left\| \frac{1}{\tau} \sum_{t=1}^{\tau} \left| \mathbf{m}^{(t)}(\mathbf{p}^{(t)}) \right| \right\|_2^{1/2} + \frac{1}{\tau} \sum_{t=1}^{\tau} (\mathbf{p}^* - \mathbf{p}^{(t)})^\top \widehat{\mathbf{r}}_t \widehat{\mathbf{r}}_t^\top \mathbf{p}^* \right\}$$

The regret bounds in Corollary (2.3) and Corollary (2.4) show that for sufficiently large number of rolling windows τ, the risk premia measured in terms of $R^2_{oos}(\tau)$ of the resulting meta-algorithm are not much worse than the risk premia of the best performing ensemble $\mathbf{p}^*$. In fact, since $\mathbf{m}^{(t)}(\mathbf{p}^{(t)}) \in [-1, 1]$, Corollary (2.4) implies that in the limit

$$\lim_{\tau \to \infty} \left\{ R^{2*}_{oos}(\tau) - R^2_{oos}(\tau) \right\} \leq \lim_{\tau \to \infty} \frac{1}{\tau} \sum_{t=1}^{\tau} (\widehat{\mathbf{p}}^* - \mathbf{p}^{(t)})^\top \widehat{\mathbf{r}}_t \widehat{\mathbf{r}}_t^\top \widehat{\mathbf{p}}^*,$$

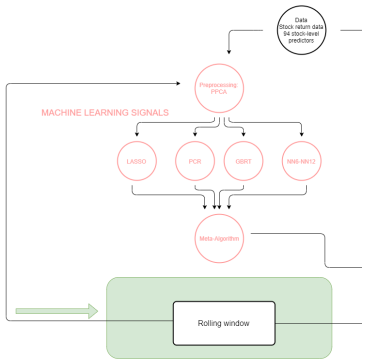
where we replace the optimal $\mathbf{p}^*$ with the consistent and unbiased estimator from Lemma (2.2).

Data



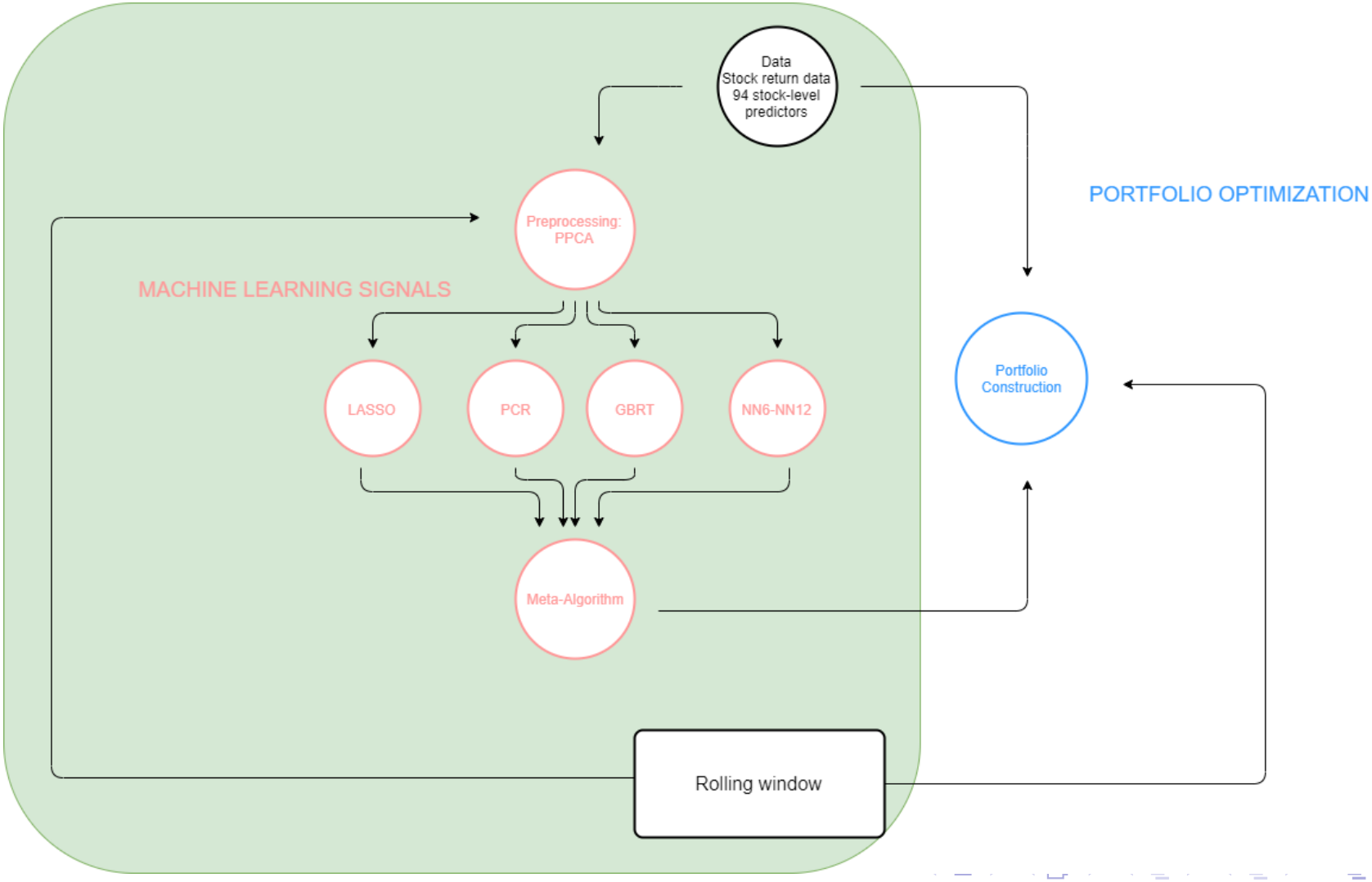
- We obtain monthly stock return data from the Center for Research in Security Prices (CRSP).
- Our sample period begins in March 1957 and ends in December 2021, totaling 65 years. Also, we obtain the 94 stock-level predictive characteristics used in Gu's paper.
- We obtain the industry dummies corresponding to the first two digits of Standard Industrial Classification (SIC) codes to classify the sectors.

Rolling Window Exercise



- We divide the 65 years of data into 30 years of training sample (1957-1986), and the remaining 35 years (1987-2021) for out-of-sample rolling window analysis.
- Each time we refit, we increase the training sample by 1 year and roll it forward to predict the most recent 12 months (with monthly updates in the predictors and MWUM algorithm weights).
- In each rolling window, we have approximately 360 observations of 30 year's data as the training set.
- We apply the cross-validation technique to the training sample to obtain meta-parameters as in the LASSO penalty strength.

Flowchart of *"Online Ensemble of Models"* (First Part)



Performance Evaluation

To assess predictive performance for sector excess stock return forecasts, we calculate the out-of-sample R^2 for each sector then average them across all the sectors. We follow the same formula as in Gu's paper and it is written as

$$R^2_{oos} = 1 - \frac{\sum (r_{i,t+1} - \hat{r}_{i,t+1})^2}{\sum r^2_{i,t+1}},$$

where $\hat{r}_{i,t+1}$ is the prediction from different ML methods and the meta-algorithm.

Comparison of R^2_{oos}

Table – Average percentage R^2_{oos} (Gu's paper - individual stocks)

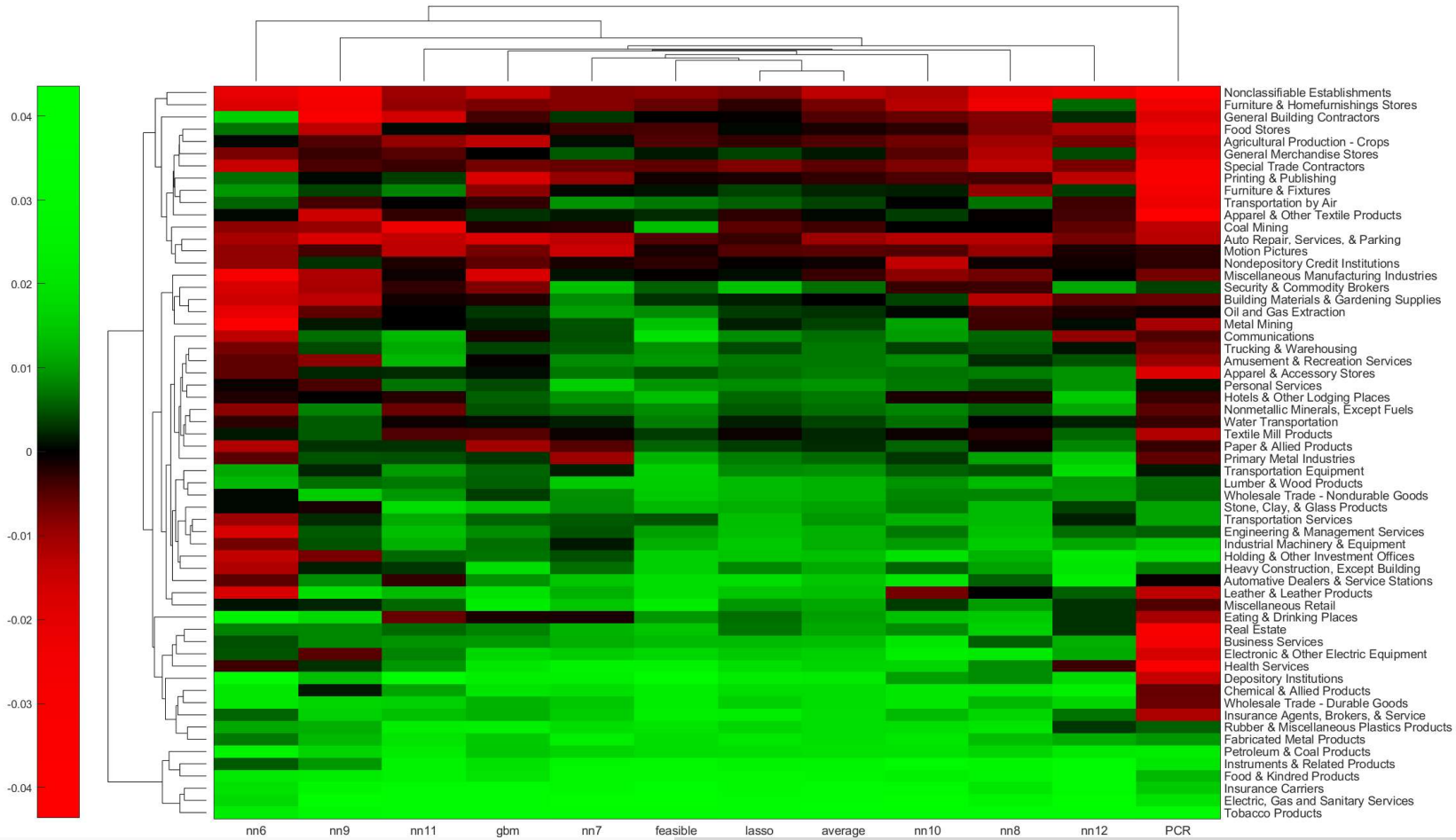
	PCR	LASSO	GBRT	NN1	NN2	NN3	NN4	NN5
All	0.26	0.11	0.34	0.33	0.39	0.40	0.39	0.36
Top 1,000	0.06	0.25	0.52	0.52	0.62	**0.70**	0.67	0.64

Table – Average percentage R^2_{oos} (our results on sector returns)

	PCR	LASSO	GBRT	NN6	NN7	NN8	NN9	NN10	NN11	NN12	Simple Averaging	η^* for $\|\mathbf{p}^*\|_2 \leq 1$	η^* for $\|\mathbf{p}^*\|_2 \leq \min(1, \delta_\tau)$	*Feasible* η^*
Average return	-0.62	0.95	0.56	-0.06	0.83	0.53	0.24	0.77	0.62	0.75	0.90	0.94	0.95	**1.19**
Weighted return	0.38	1.11	0.89	0.35	0.36	1.11	0.89	0.95	0.80	0.75	1.07	1.09	1.10	**1.12**

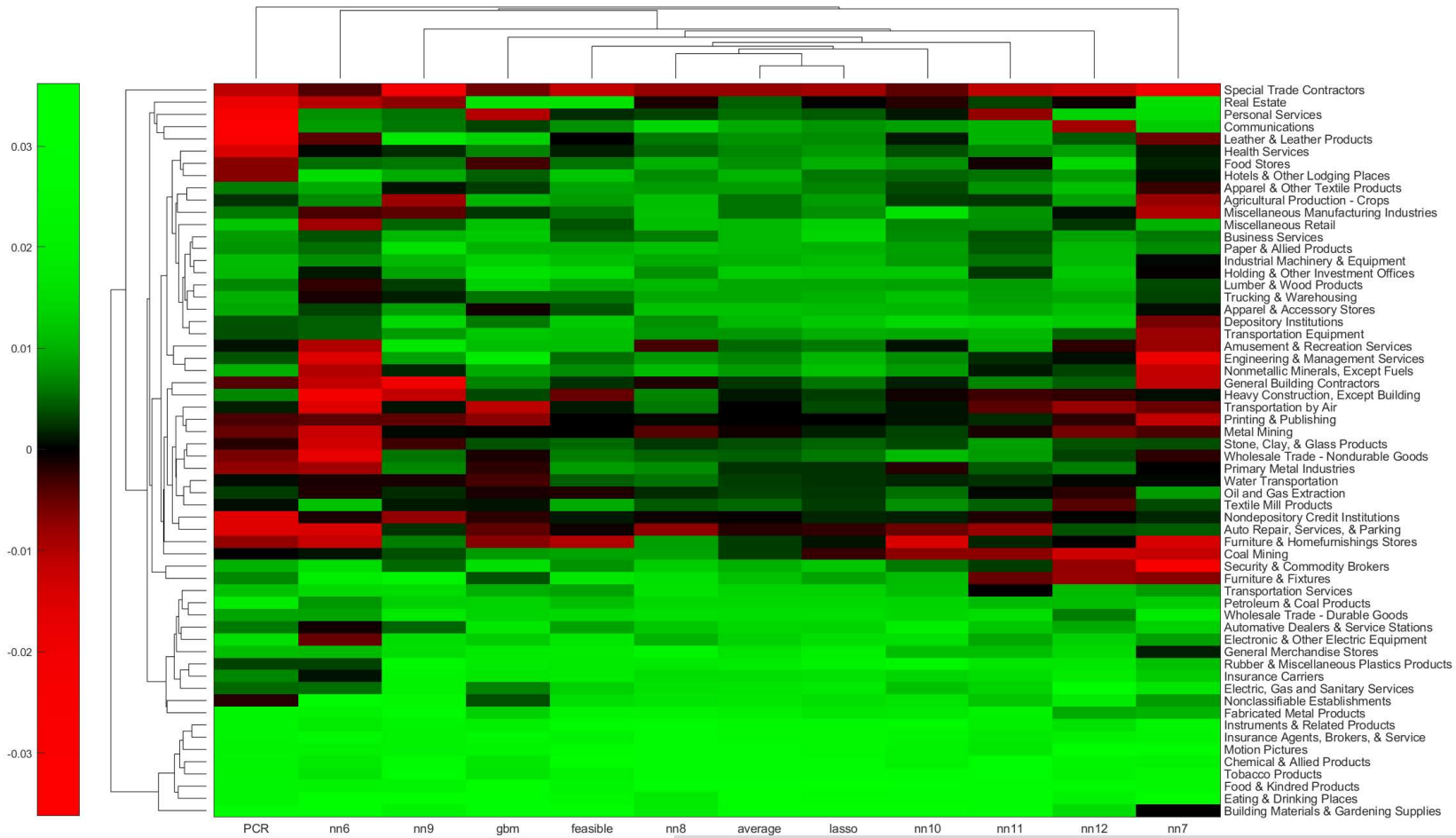
Return Prediction

Equally-weighted Scenario (ensemble the best)



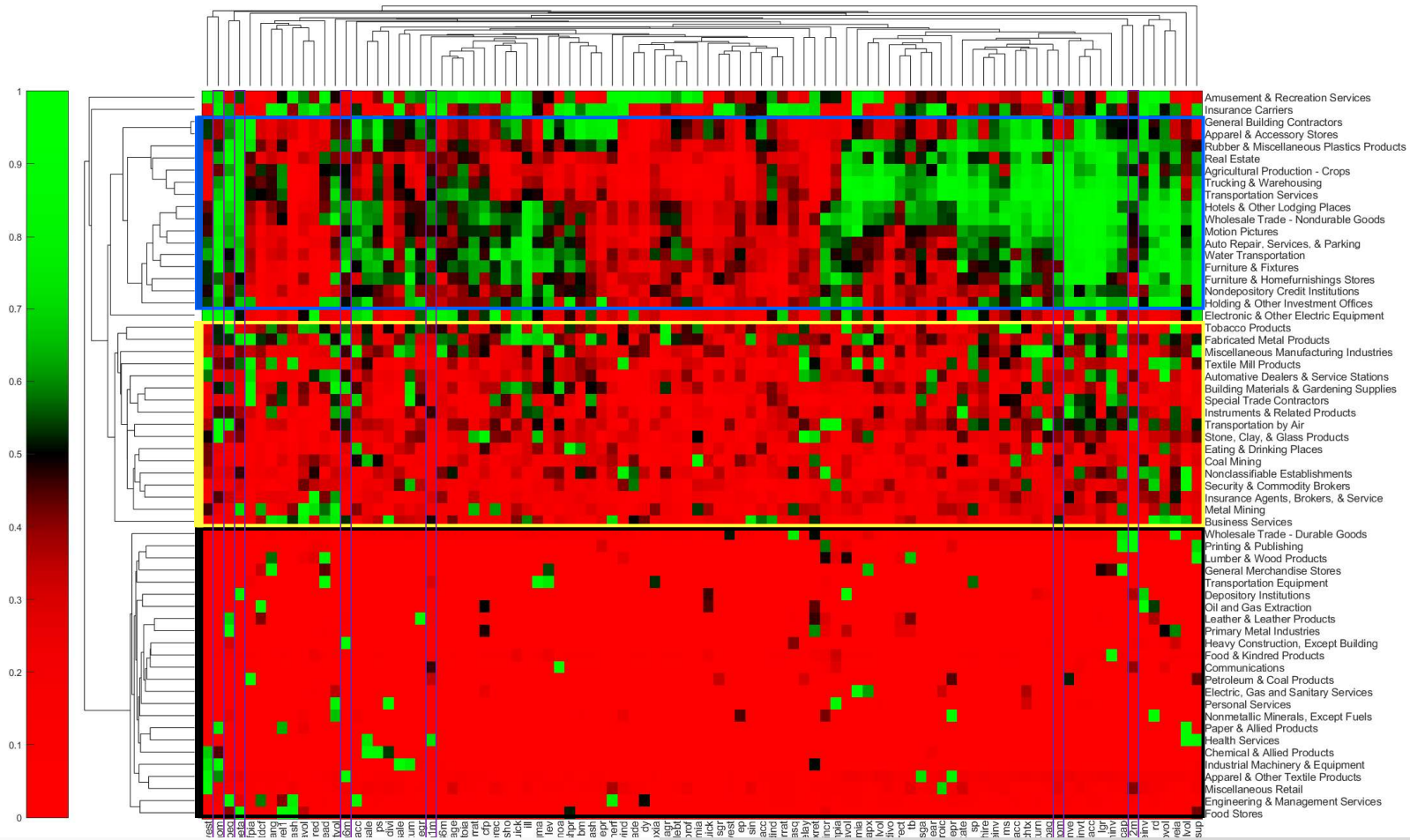
Return Prediction

Capital-weighted Scenario (ensemble the best)



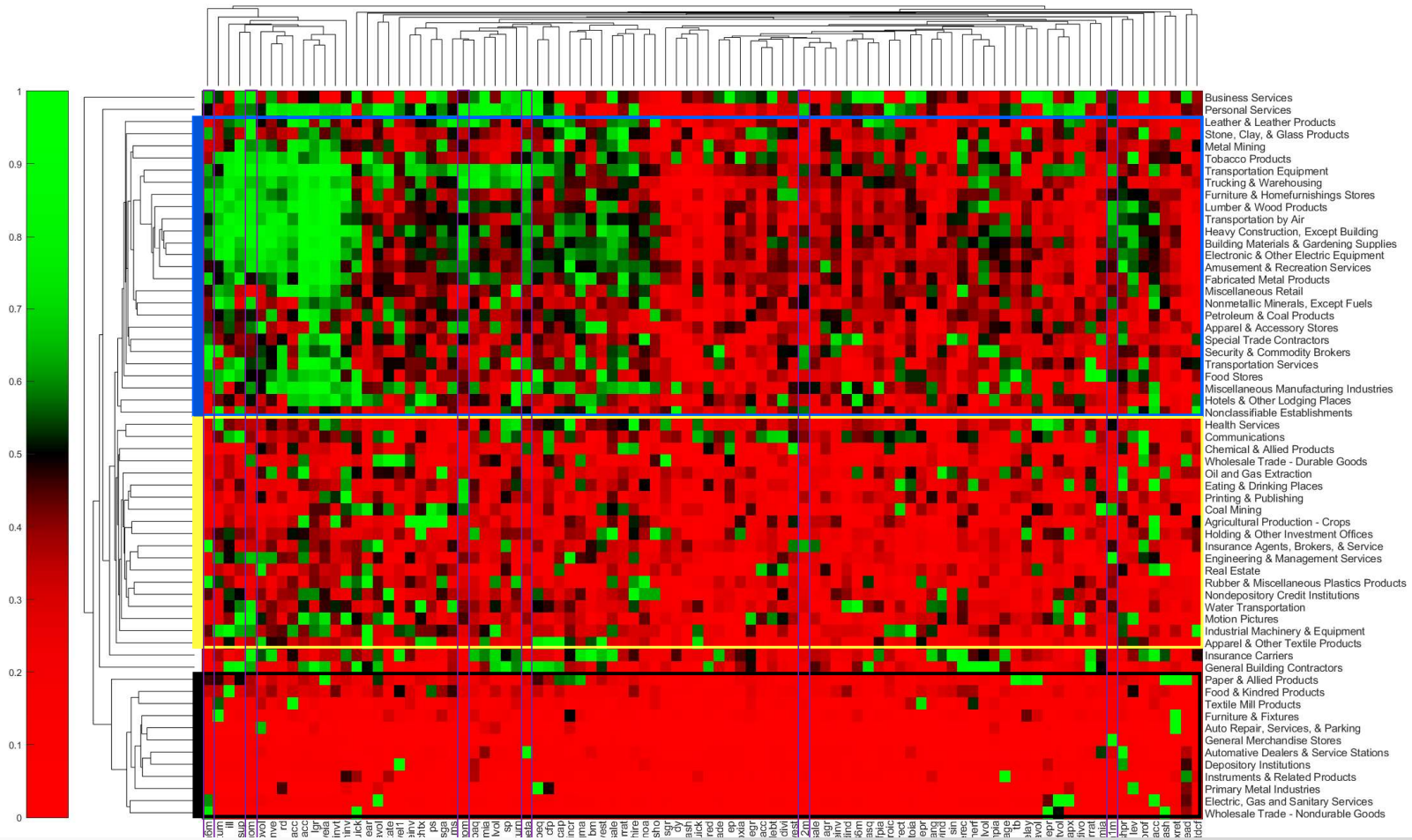
Heatmap of the Variables Importance Factors (VIF)

Equally-weighted Scenario (three main clusters of sectors)

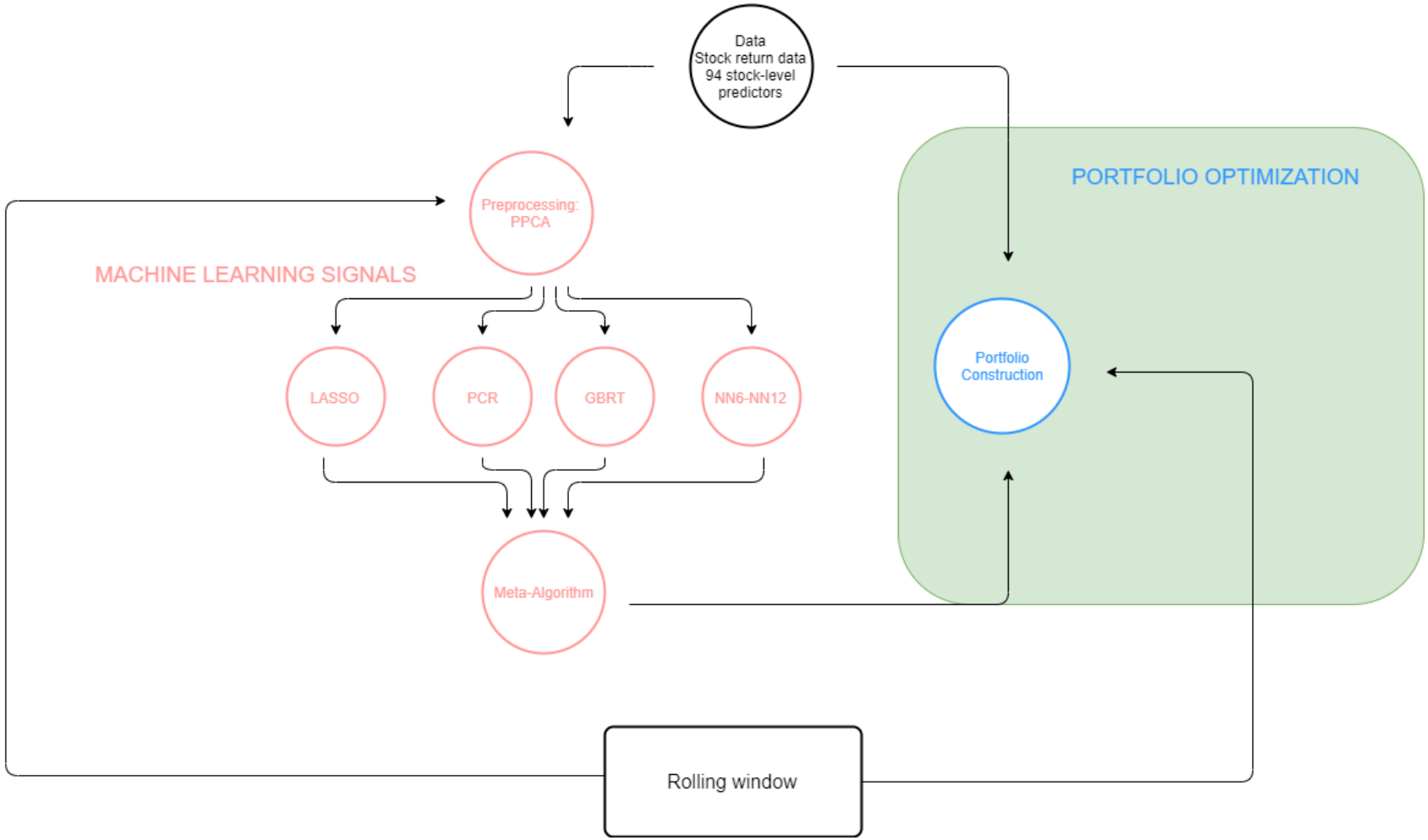


Heatmap of the Variables Importance Factors (VIF)

Capital-weighted Scenario (three main clusters of sectors)



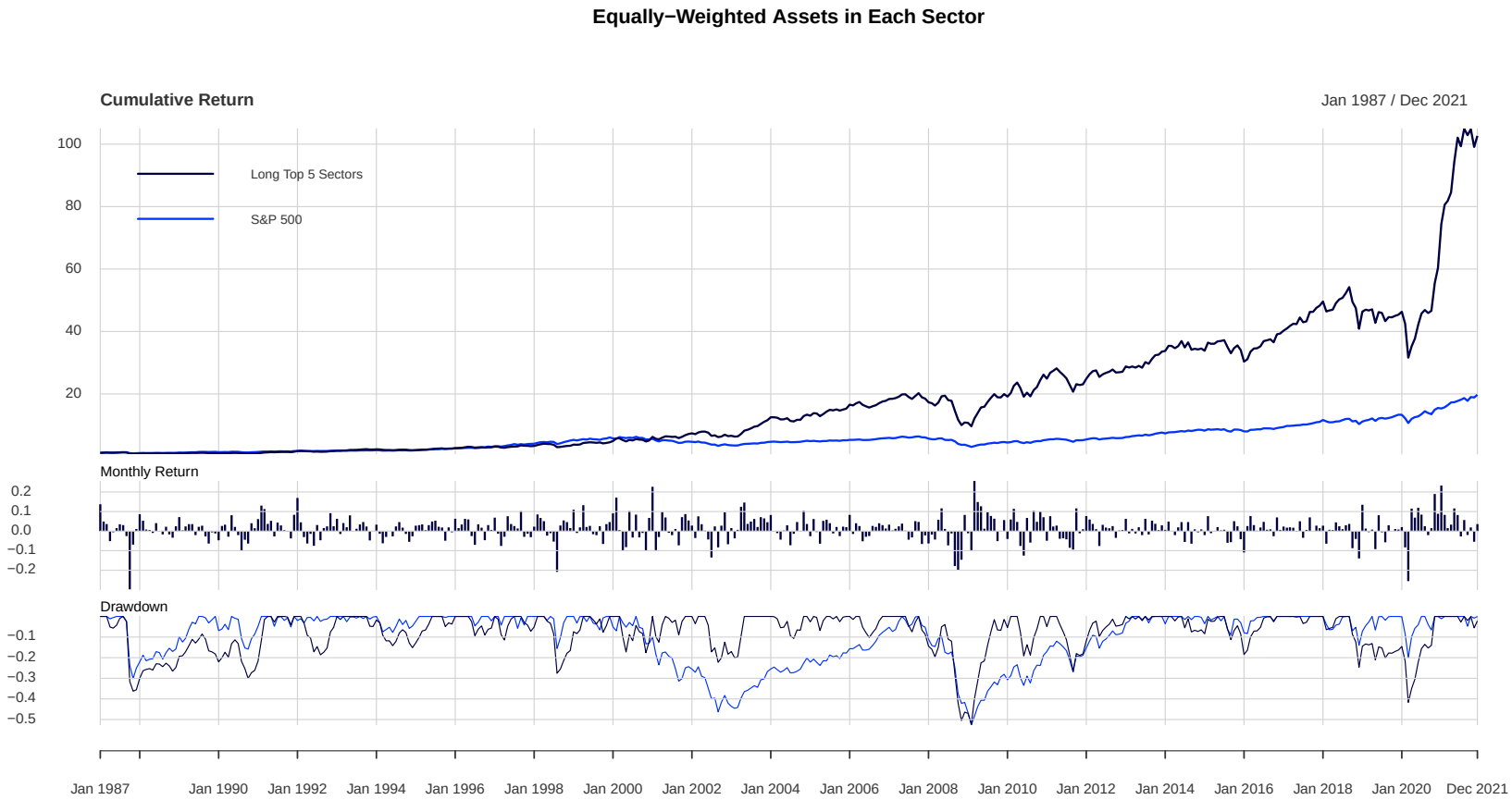
Flowchart of *"Online Ensemble of Models"* (Second Part)



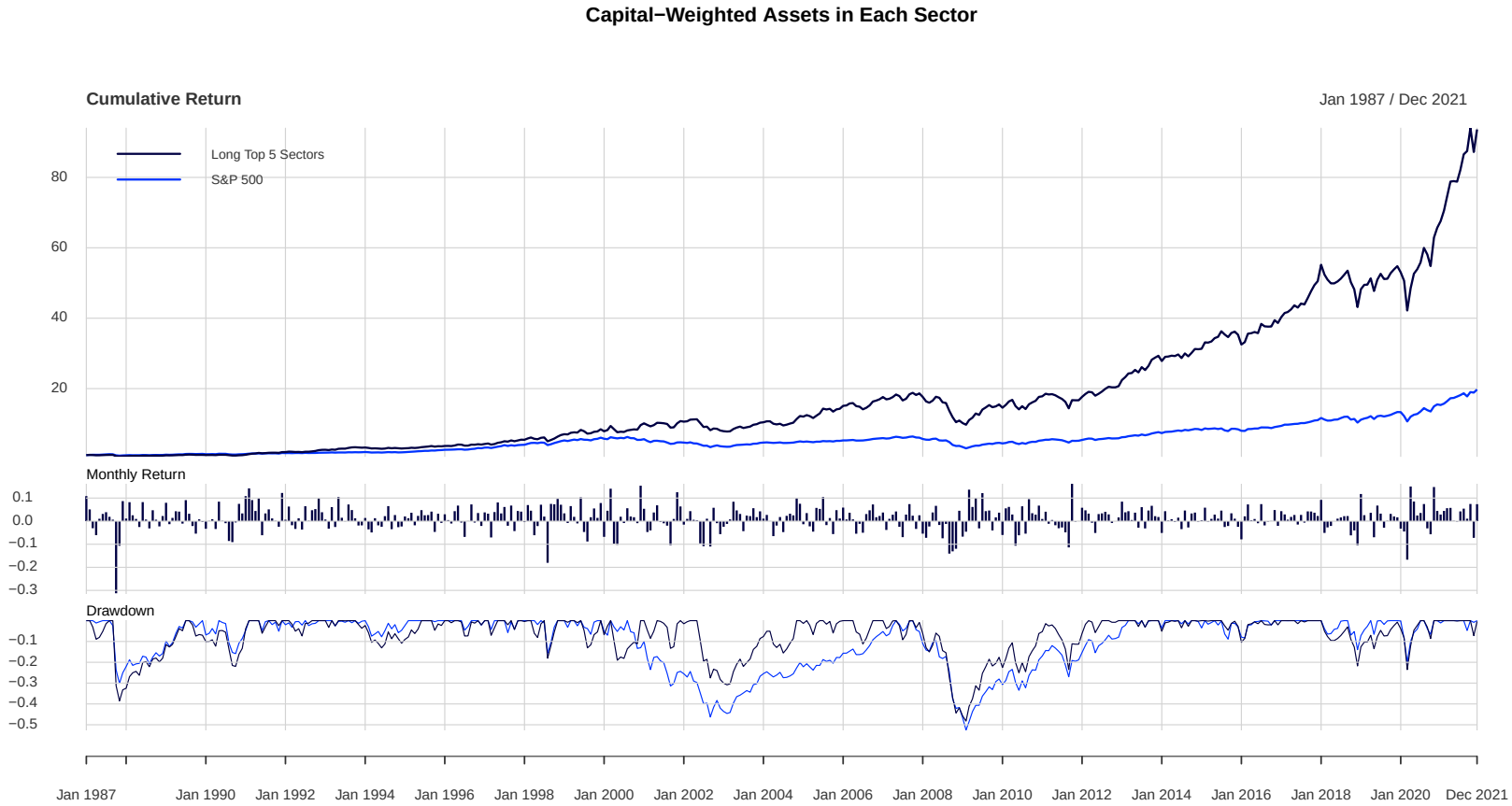
Sector Rotation Portfolio Strategy

- We construct simple portfolios of top 5 sectors according to the prediction of our meta-strategy.
- We provide backtesting results of the strategy for different definitions of sector returns.
- We show that the performance of the strategy cannot be explained by standard factor models.

Performance Analysis



Performance Analysis



Out-of-sample Years (1987-2021)

Table – Portfolio Statistics

Statistics (1987-2021)	S&P 500	1/N	Bottom 5	41-55	21-40	6-20	Top 5	5BPs	Top 5 Net 10BPs	15BPs
Panel I.A.										
Equally-Weighted Returns										
Annual Return	0.0889	0.1048	0.0661	0.0921	0.0983	0.1228	0.1415	0.1368	0.1322	0.1275
Annual Volatility	0.1504	0.1971	0.2304	0.2041	0.1951	0.1973	0.2153	0.2152	0.2151	0.2150
Annual Sharpe	0.5912	0.5317	0.2869	0.4511	0.5037	0.6225	0.6570	0.6357	0.6143	0.5931
Max. Drawdown	0.5256	0.6219	0.6639	0.6716	0.6204	0.5937	0.5233	0.5263	0.5292	0.5321
Annual Sortino	0.9487	0.9114	0.5903	0.8105	0.8640	1.0452	1.1465	1.1122	1.0782	1.0443
Panel I.B.										
Capital-Weighted Returns										
Annual Return	0.0889	0.1094	0.0950	0.0930	0.1117	0.1126	0.1385	0.1342	0.1300	0.1258
Annual Volatility	0.1504	0.1703	0.2064	0.1826	0.1686	0.1686	0.1883	0.1883	0.1883	0.1884
Annual Sharpe	0.5912	0.6422	0.4602	0.5093	0.6625	0.6676	0.7355	0.7129	0.6904	0.6679
Max. Drawdown	0.5256	0.5464	0.5995	0.5966	0.5617	0.4967	0.4825	0.4862	0.4899	0.4935
Annual Sortino	0.9487	1.0245	0.8132	0.8327	1.0657	1.0693	1.2190	1.1832	1.1477	1.1124

Recent COVID-19 Period 2020-2021

Table – Portfolio Statistics

Statistics (2020-2021)	S&P 500	1/N	Bottom 5	41-55	21-40	6-20	Top 5	Top 5 Net 5BPs	Top 5 Net 10BPs	Top 5 Net 15BPs
Panel II.A.										
Equally-Weighted Returns										
Annual Return	0.2162	0.2847	0.1915	0.3138	0.1925	0.3350	0.5040	0.4957	0.4875	0.4794
Annual Volatility	0.1954	0.3147	0.3623	0.3300	0.3336	0.2837	0.3324	0.3322	0.3320	0.3319
Annual Sharpe	1.1067	0.9049	0.5287	0.9507	0.5769	1.1807	1.5163	1.4922	1.4683	1.4444
Max. Drawdown	0.2000	0.3524	0.4569	0.3568	0.3896	0.2750	0.3173	0.3181	0.3190	0.3198
Annual Sortino	1.8400	1.5121	1.0014	1.6386	1.0437	2.0379	2.4006	2.3646	2.3287	2.2931
Panel II.B.										
Capital-Weighted Returns										
Annual Return	0.2162	0.2099	0.1955	0.1335	0.2396	0.2126	0.3073	0.3021	0.2970	0.2920
Annual Volatility	0.1954	0.2457	0.3273	0.2792	0.2361	0.2240	0.2410	0.2410	0.2410	0.2409
Annual Sharpe	1.1067	0.8545	0.5973	0.4782	1.0147	0.9495	1.2749	1.2538	1.2327	1.2117
Max. Drawdown	0.2000	0.2996	0.4178	0.3489	0.2835	0.2581	0.2300	0.2313	0.2327	0.2340
Annual Sortino	1.8400	1.3506	1.0135	0.8278	1.6604	1.4684	2.1066	2.0694	2.0325	1.9959

Returns on Portfolios Selected by Meta-strategy

Table – Alphas of the Strategy

Statistics	Bottom 5	41-55	21-40	6-20	Top 5	Top-Bottom
Panel A. Equally-Weighted Returns						
Predicted return	0.0019	0.0067	0.0096	0.0125	0.0172	0.0177
Excess return	0.0076 **	0.0091 ***	0.0095 ***	0.0114 ***	0.0130 ***	0.0079 ***
	(2.0564)	(2.8666)	(3.0509)	(3.6262)	(3.9595)	(4.2205)
CAPM alpha	0.0014	0.0032	0.0036 *	0.0054 ***	0.0071 ***	0.0081 ***
	(0.5288)	(1.4026)	(1.8481)	(2.7691)	(3.0285)	(4.1263)
3F alpha	-0.0016	0.0004	0.0010	0.0028 ***	0.0047 ***	0.0088 ***
	(-0.9944)	(0.2670)	(0.9254)	(2.6207)	(2.9736)	(4.5865)
4F alpha	0.0010	0.0022	0.0026 ***	0.0039 ***	0.0053 ***	0.0068 ***
	(0.5971)	(1.6141)	(2.6925)	(3.4015)	(3.0470)	(4.0487)
Panel B. Capital-Weighted Returns						
Predicted return	0.0008	0.0055	0.0084	0.0116	0.0178	0.0195
Excess return	0.0094 ***	0.0089 ***	0.0101 ***	0.0101 ***	0.0124 ***	0.0054 **
	(3.0388)	(3.6618)	(4.1568)	(4.5633)	(4.8145)	(2.4713)
CAPM alpha	0.0032	0.0027 *	0.0043 ***	0.0043 ***	0.0064 ***	0.0056 **
	(1.3212	(1.8506)	(3.2495)	(4.6785)	(3.7073)	(2.5342)
3F alpha	0.0003	0.0001	0.0019 **	0.0022 ***	0.0042 ***	0.0064 ***
	(0.1955)	(0.1330)	(2.3733)	(2.7900)	(2.9251)	(3.3200)
4F alpha	0.0020	0.0010	0.0023 ***	0.0023 ***	0.0035 **	0.0039 **
	(1.3503)	(1.0504)	(2.6683)	(3.0013)	(2.2045)	(2.0953)

Conclusions

- We perform a comparative analysis of machine learning methods for measuring sector risk premia, and demonstrate large economic gains from using machine learning for sector forecasting.
- We develop a framework to assemble different forecasting models and explain how to use the resulting meta-strategy in portfolio optimization.
- The developed meta strategy delivers systemically high R^2_{oos} across all the sectors.

Conclusions

- We perform a comparative analysis of machine learning methods for measuring sector risk premia, and demonstrate large economic gains from using machine learning for sector forecasting.
- We develop a framework to assemble different forecasting models and explain how to use the resulting meta-strategy in portfolio optimization.
- The developed meta strategy delivers systemically high R^2_{oos} across all the sectors.

Conclusions

- We perform a comparative analysis of machine learning methods for measuring sector risk premia, and demonstrate large economic gains from using machine learning for sector forecasting.
- We develop a framework to assemble different forecasting models and explain how to use the resulting meta-strategy in portfolio optimization.
- The developed meta strategy delivers systemically high R^2_{oos} across all the sectors.

Thank You !

Paweł Polak
pawel.polak@stonybrook.edu