Sequential asset ranking in nonstationary time series

Gabriel Borrageiro, Nick Firoozye, Paolo Barucca
gabriel.borrageiro.20@ucl.ac.uk
Department of Computer Science
University College London
London, United Kingdom

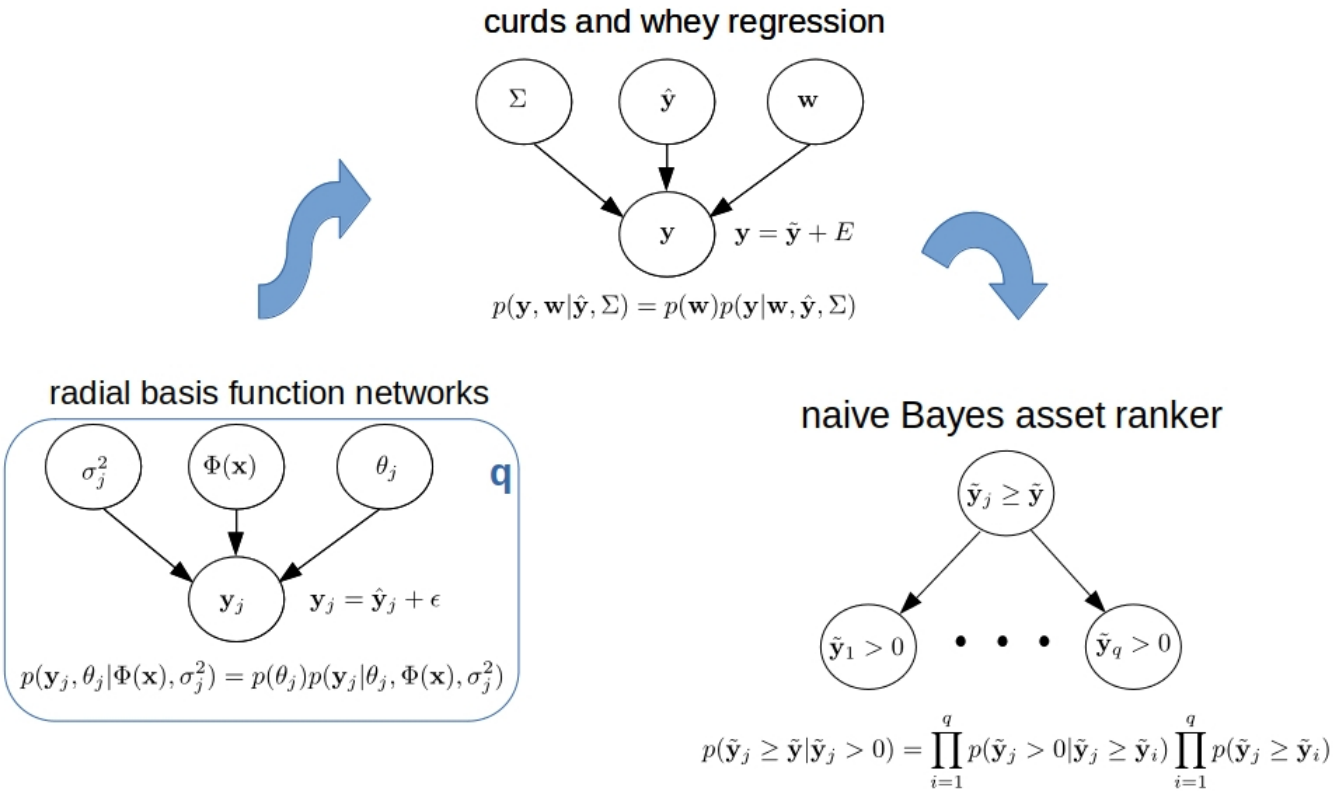

Figure 1: The radial basis function network forecasts are fed into the curds and whey multivariate regression model, whose output is ranked and selected by the naive Bayes asset ranker.

ABSTRACT

We extend the research into cross-sectional momentum trading strategies. Our main result is our novel ranking algorithm, the naive Bayes asset ranker (nbar), which we use to select subsets of assets to trade from the S&P 500 index. We perform feature representation transfer from radial basis function networks to a curds and whey (caw) multivariate regression model that takes advantage of the correlations between the response variables to improve predictive accuracy. The nbar ranks this regression output by forecasting the one-step-ahead sequential posterior probability that individual assets will be ranked higher than other portfolio constituents. Earlier algorithms, such as the weighted majority, deal with nonstationarity by ensuring the weights assigned to each expert never dip below a minimum threshold without ever increasing weights again. Our ranking algorithm allows experts who previously performed poorly to have increased weights when they start performing well. Our algorithm outperforms a strategy that would hold the long-only S&P 500 index with hindsight, despite the index appreciating by 205% during the test period. It also outperforms a regress-then-rank baseline, the caw model.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
ICAIF '22, November 2–4, 2022, New York, NY, USA
© 2022 Association for Computing Machinery.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/XXXXXXX.XXXXXXX

CCS CONCEPTS

• **Computer systems organization** → *Neural networks*; *Neural networks*; • **Mathematics of computing** → *Probabilistic algorithms*; *Time series analysis*; • **Information systems** → **Learning**

to rank; **Expert systems**; **Learning to rank**; *Expert systems*; *Clustering*; **Learning to rank**; **Learning to rank**; **Learning to rank**; • **Theory of computation** → *Regret bounds*; Kernel methods; **Online learning theory**; *Unsupervised learning and clustering*; **Online learning algorithms**; *Regret bounds*; **Online learning algorithms**; **Multi-agent learning**; **Online learning algorithms**; **Online learning algorithms**; • **Computing methodologies** → *Transfer learning*; **Online learning settings**; *Neural networks*; *Transfer learning*; *Neural networks*; *Kernel methods*; *Transfer learning*; *Supervised learning by regression.*

KEYWORDS

online learning, prediction with expert advice, learning to rank, transfer learning, radial basis function networks, multivariate regression shrinkage

ACM Reference Format:
Gabriel Borrageiro, Nick Firoozye, Paolo Barucca. 2022. Sequential asset ranking in nonstationary time series. In *3rd ACM International Conference on AI in Finance (ICAIF '22), November 2–4, 2022, New York, NY, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Our particular modelling interest is in financial time series, which are typically nonstationary. Nonstationarity implies statistical distributions that adapt over time and violates the independent and identically distributed (iid) random variables assumption of most regression and classification models. We require approaches that adopt sequential optimisation methods, preferably methods that make little or no assumptions about the data-generating process. The prediction with expert advice framework [8] is a multidisciplinary area of research suited to predicting sequences sequentially, where statistical distribution assumptions are not made. This framework minimises the regret concerning the best available expert with hindsight and is well-suited to portfolio selection problems.

The main result of this paper is our novel ranking algorithm, the naive Bayes asset ranker, which we use to select subsets of assets to trade from the S&P 500 index in either a long-only or a long/short (cross-sectional momentum) capacity. Our ranking algorithm forecasts the one-step-ahead sequential posterior probability that individual assets will be ranked higher than other constituents in the portfolio. Earlier algorithms, such as the weighted majority algorithm [20], deal with nonstationarity by ensuring the weights assigned to each expert never dip below a minimum threshold without ever increasing weights again. In contrast, our ranking algorithm allows experts who performed poorly previously to have increased weight when they start performing well. Finally, our algorithm computes the posterior ranking probabilities with exponential decay and is better suited to learning in nonstationary environments.

We achieve higher risk-adjusted and total returns than a strategy that would hold the long-only S&P 500 index with hindsight, despite the index appreciating by 205% during the test period. We also outperform a regress-then-rank baseline, a sequentially fitted curds and whey multivariate regression model.

We end this introduction by providing a roadmap for this paper's layout. In section 2, we provide background to our research and perform a literature review. We provide a rationale for online learning in finance in section 2.1. This is followed by a short introduction to transfer learning in section 2.2, radial basis function networks in section 2.3 and curds and whey multivariate regression in section 2.4. The final sections introduce the prediction with expert advice framework in section 2.5, portfolio selection with expert advice in section 2.6 and an introduction to the concept of naive Bayes ranking in section 2.7.

2 BACKGROUND AND LITERATURE REVIEW

This section introduces the various models and topics that are the foundations of our experimental work. The subjects discussed are feature representation transfer, multivariate regression, prediction with expert advice and ranking algorithms.

2.1 The rationale for online learning in finance

Financial time series exhibit serial correlation and nonstationarity characteristics. High serial correlation results in spuriously high R^2 for regression models [12]. Nonstationarity results in batch learning models generalising poorly on unseen test data. Merton [23] models the dynamics of financial assets as a jump-diffusion process. The model superimposes a diffusion component modelled by geometric Brownian motion, with a jump component driven by a Poisson process. The goal is to model abrupt changes in prices due to the arrival of new information, facilitating the increased likelihood of tail events compared to the normal distribution. Financial time series are also modelled as a random-walk process [22]. The random-walk process implies the unpredictability of economic time series returns and their time-varying random nature, which Nakamura and Small [25] find exists in equities, currencies, precious metals and energy returns.

One approach for coping with nonstationarity is to learn online continuously. One may combine sequential optimisation with states-of-nature/transitional learning approaches such as reinforcement learning [35] or continual learning approaches such as transfer learning [36]. Nonstationarity is typically identified through unit root tests such as the augmented Dickey-Fuller (adf) test [31]. In section 4, we experiment with 21 years of daily sampled S&P 500 data between 2001-01-26 and 2022-03-25. When running the adf test against the S&P 500 daily transaction prices, 90.5% of the time series are considered nonstationary and when running the test against their linear returns of the form

$$y_{j,t} = p_{j,t}/p_{j,t-1} - 1, \quad j = 1, ..., d, \tag{1}$$

where j denotes the constituent and t denotes the time-step, 100% of the time series are now considered stationary. One could now assume that performing batch learning on the daily returns is reasonable, and nonstationarity is no longer a concern. A simple experiment calls this view into question, as whilst the returns might be stationary, the models that operate on such time series require parameters that must be adapted over time. To illustrate this point succinctly, we fit ar(1) models with structural form

$$y_{j,t} = w_{j,t} y_{j,t-1} + \epsilon_{j,t},$$

to each constituent's daily returns. The models are trained daily using the past $n = 100$ days on a rolling window basis. Using one-week non-overlapping time buckets, we average the ar(1) coefficients and denote them as $\overline{w}_{j,t}$. Finally, we conduct two-sample t-tests for equal means [34] where we compare $\overline{w}_{j,t}$ with $\overline{w}_{j,t+h}$ for $h = 1, ..., 52$ weeks, setting the significance level of the test, $\alpha = 0.05$. We find that as the shift h increases, the probability that the null hypothesis $H_0 : \overline{w}_{j,t} = \overline{w}_{j,t+h}$ is rejected, increases considerably. Figure 2 averages the results across the S&P 500 constituents and time shifts; it demonstrates that the ar(1) parameters are changing over time, despite the training data being almost the same for small shifts h. Specifically, the fraction of matched training days between coefficients $\overline{w}_{j,t}$ and $\overline{w}_{j,t+h}$ is $max\left(0, \frac{n-5h}{n}\right)$.

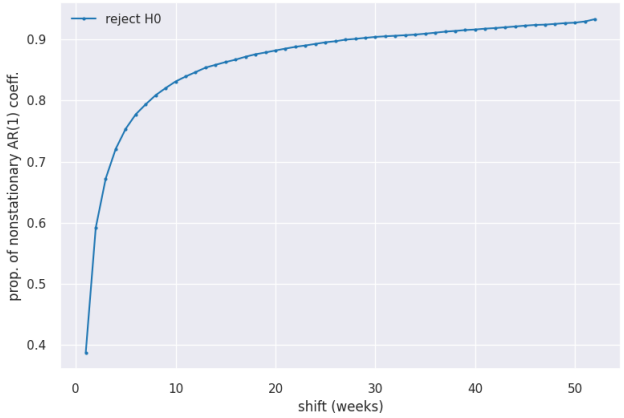

Figure 2: The probability that $H_0 : \overline{w}_{j,t} = \overline{w}_{j,t+h}$ is rejected by the two-sample t-test for equal means.

2.2 Transfer learning

Transfer learning refers to the machine learning paradigm in which an algorithm extracts knowledge from one or more application scenarios to help boost the learning performance in a target scenario [36]. Typically, traditional machine learning requires significant amounts of training data. Transfer learning copes better with data sparsity by looking at related learning domains where data is sufficient. Even with large datasets, including streaming data, transfer learning provides benefits by learning the adaptive statistical relationship of the source and target domains. Following Pan and Yang [26]:

Definition 2.1 (transfer learning). Given a source domain $\mathcal{D}_S$ and learning task $\mathcal{T}_S$, a target domain $\mathcal{D}_T$ and learning task $\mathcal{T}_T$, transfer learning aims to help improve the learning of the target predictive function $f_T(.)$ in $\mathcal{D}_T$ using the knowledge in $\mathcal{D}_S$ and $\mathcal{T}_S$, where $\mathcal{D}_S \neq \mathcal{D}_T$, or $\mathcal{T}_S \neq \mathcal{T}_T$.

Feature-based approaches transform the original features to create a new feature representation. For example, Borrageiro et al. [6] perform online feature representation transfer from radial basis function networks (rbfnets) to sequentially optimised reinforcement learning agents, who learn to risk-manage and trade currency pairs. Similarly, they apply feature representation transfer from echo state networks [15] to reinforcement learning agents who learn to trade cryptocurrency perpetual swaps, capturing a funding profit [5].

2.3 The radial basis function network

The rbfnet is a single-layer network whose hidden units are radial basis functions of the form

$$\phi_j(\mathbf{x}) = \exp\left(-\frac{1}{2}[\mathbf{x} - \boldsymbol{\mu}_j]^T \Sigma_j^{-1} [\mathbf{x} - \boldsymbol{\mu}_j]\right). \tag{2}$$

The hidden unit means and covariances are typically learnt through clustering algorithms such as k-means [21]. The hidden unit outputs are aggregated into a feature vector

$$\boldsymbol{\phi}_t = [1, \phi_1(\mathbf{x}), ..., \phi_k(\mathbf{x})],$$

and mapped to the response via regression

$$y_t = \boldsymbol{\theta}^T \boldsymbol{\phi}_t + \epsilon_t.$$

The separation of class distributions modelled by local radial basis functions is probabilistic. The predictive uncertainty increases where there is class-conditional distribution overlap. The radial basis functions can be thought of as the posterior feature probabilities, and the weights can be interpreted as the posterior probabilities of class membership, given the presence of the features [3]. It is first with Moody and Darken [24] that we see the formulation of the rbfnet as the combination of an unsupervised learning model (k-means) and a supervised learning model (linear regression). This form of feature representation transfer boosts model learning capacity and, when combined with sequential optimisation, outperforms biased baselines such as the random-walk in multi-horizon returns forecasting [4].

2.4 Curds and whey multivariate regression

The curds and whey (caw) procedure due to Breiman and Friedman [7] is a suitable way of predicting several response variables from the same set of explanatory variables. The method takes advantage of the correlations between the response variables to improve predictive accuracy compared with the usual procedure of doing individual regressions of each response variable on the shared set of predictor variables. Assume there are n training examples, d predictors $\mathbf{X} \in \mathbb{R}^{n \times d}$ and q targets $\mathbf{Y} \in \mathbb{R}^{n \times q}$. The basic version of the procedure begins with the multivariate ridge regression

$$\boldsymbol{\Theta} = (\mathbf{X}^T\mathbf{X} + \lambda \mathbf{I}_d)^{-1}\mathbf{X}^T\mathbf{Y},$$

and estimates a shrinkage matrix

$$\hat{\mathbf{Y}} = \mathbf{X}\boldsymbol{\Theta}$$
$$\mathbf{W} = (\hat{\mathbf{Y}}^T\hat{\mathbf{Y}} + \lambda \mathbf{I}_q)^{-1}\hat{\mathbf{Y}}^T\mathbf{Y}.$$

At test time, the vector of forecasts is

$$\tilde{\mathbf{y}}_t = \mathbf{W}\hat{\mathbf{y}}_t \equiv \mathbf{W}[\mathbf{x}_t^T\boldsymbol{\Theta}]^T.$$

Given the nonstationary data we model, our particular interest is in sequential optimisation. Thus we combine the caw procedure with ewrls updating; this is shown in algorithm 1.

Algorithm 1: Sequentially optimised curds and whey regression.

Require: λ, τ
// λ is a ridge penalty
// τ is an exponential decay constant
Initialise: $\Theta = \mathbf{0}_{d\times q}, \mathbf{W} = \mathbf{0}_{q\times q}, \mathbf{P} = \mathbf{I}_d/\lambda, \mathbf{Q} = \mathbf{I}_q/\lambda$
Input: $\mathbf{x}_t \in \mathbb{R}^d, \mathbf{y}_t \in \mathbb{R}^q$
Output: $\tilde{\mathbf{y}}_t \in \mathbb{R}^q$

$r_t = 1 + \mathbf{x}_{t-1}^T \mathbf{P}_{t-1} \mathbf{x}_{t-1}/\tau$
$\mathbf{k}_t = \mathbf{P}_{t-1}\mathbf{x}_{t-1}/(r_t\tau)$
$\hat{\mathbf{y}}_{t-1} = [\mathbf{x}_{t-1}^T \Theta_{t-1}]^T$
$\Theta_t = \Theta_{t-1} + \mathbf{k}_t(\mathbf{y}_t - \hat{\mathbf{y}}_{t-1})^T$
$\mathbf{P}_t = \mathbf{P}_{t-1}/\tau - \mathbf{k}_t \mathbf{k}_t^T r_t$
$\mathbf{P}_t = \mathbf{P}_{t-1}\tau$
$\hat{\mathbf{y}}_t = [\mathbf{x}_t^T \Theta_t]^T$
$s_t = 1 + \hat{\mathbf{y}}_{t-1}^T \mathbf{Q}_{t-1} \hat{\mathbf{y}}_{t-1}/\tau$
$\mathbf{m}_t = \mathbf{Q}_{t-1}\hat{\mathbf{y}}_{t-1}/(s_t\tau)$
$\mathbf{W}_t = \mathbf{W}_{t-1} + \mathbf{m}_t(\mathbf{y}_t - \mathbf{W}_{t-1}\hat{\mathbf{y}}_{t-1})^T$
$\mathbf{Q}_t = \mathbf{Q}_{t-1}/\tau - \mathbf{m}_t \mathbf{m}_t^T s_t$
$\mathbf{Q}_t = \mathbf{Q}_{t-1}\tau$
$\tilde{\mathbf{y}}_t = \mathbf{W}_t \hat{\mathbf{y}}_t$

2.5 Prediction with expert advice

Prediction with expert advice is a multidisciplinary research area that predicts individual sequences in an online learning setting. Unlike standard statistical approaches, the prediction with expert advice framework imposes no probabilistic assumption on the data-generating mechanism. Instead, it generates predictions that work well for all sequences, with performance nearly as good as the best expert with hindsight [8]. The basic structure of problems in this context is encapsulated in algorithm 2, adapted from Rakhlin and Sridharan [30]. Perhaps the most well-known algorithm within this

Algorithm 2: Prediction with expert advice

for $t \leftarrow 1$ **to** T **do**
  Learner chooses the set of predictions $\hat{\mathbf{y}}_t \in \mathcal{D}$.
  Nature simultaneously chooses an outcome $y_t \in \mathcal{A}$.
  Player suffers a loss $\ell(\hat{\mathbf{y}}_t, y_t)$ and all observe $(\hat{\mathbf{y}}_t, y_t)$.

framework is the weighted majority (wm) algorithm of Littlestone and Warmuth [20]. The authors study the construction of prediction algorithms where the learner faces a sequence of trials, and the goal is to make as few mistakes as possible with predictions made at the end of each trial. They are interested in cases where the learner believes some experts will perform well but does not know which ones. A simple method based on weighted voting is introduced to minimise the regret concerning the best expert with hindsight. Given a sequence of trials, if there is an algorithm in the pool d that makes at most m mistakes, then the wm algorithm will make at most $c(\log d + m)$ mistakes on that sequence, where c is fixed constant. However, the wm algorithm is less suited to nonstationary data, despite the authors modifying their base algorithm to ensure that the weight assigned to the individual experts never dips below η/d, which is the learning rate divided by the number of experts. This fixed, minimum threshold weight is somewhat rudimentary. In the context of financial time series, our preference is for algorithms that assign more weight to experts now performing well, irrespective of whether they performed less well previously.

2.6 Portfolio selection with expert advice

Helmbold et al. [14] present an online investment algorithm that achieves almost the same wealth as the best constant-rebalanced portfolio with hindsight, including Cover's universal portfolio selection algorithm [9]. The algorithm employs a multiplicative update rule using a framework introduced by Kivinen and Warmuth [18]. Singer [33] notes that the earlier work into online portfolio selection algorithms which are competitive with the best constant rebalanced portfolio determined in hindsight [10], employ the assumption that high yield returns can be achieved using a fixed asset allocation strategy. However, the return of a constant rebalanced portfolio is often much smaller than the return of an ad-hoc investment strategy that adapts to changes in the market. Singer presents an efficient portfolio selection algorithm that tracks a changing market and describes a simple extension of his algorithm for including transaction costs.

Poh et al. [27] apply batch (offline) learning-to-rank algorithms, originally designed for natural language processing, to cross-sectional momentum trading strategies. Cross-sectional strategies mitigate some of the risk associated with wider market moves by buying the top α-percentile of strategies with the highest expected future returns and selling the bottom α-percentile of strategies with the lowest expected future returns. Classical approaches that rely on the ranking of assets include ranking annualised returns [16], or more recent regress-then-rank approaches [13]. Poh et al. [28] use a context-aware learning-to-rank model based on the transformer architecture to encode top/bottom-ranked assets, learn the context and exploit this information to rerank the initial results.

2.7 Naive Bayes ranking

The academic literature shows that naive Bayes ranking has a broad meaning. At the core of the idea is the naive Bayes classifier

$$P(y_c|\mathbf{x}) = \frac{P(\mathbf{x}|y_c)P(y_c)}{\sum_j P(\mathbf{x}|y_j)P(y_j)},$$

which predicts that the target y takes on a label value of c if this posterior probability is highest. Independence assumptions in the inputs mean that the probabilities are modelled iteratively and inexpensively. Zhang and Su [37] study the general performance of naive Bayes in ranking, using the auc curve [29] to evaluate the quality of rankings generated by a classifier. Flach and Matsubara [11] consider binary classification tasks, where a ranker sorts a set of instances from highest to lowest expectation that the instance is positive. Using the odds ratio to rank the attribute values, they obtain a restricted version of the naive Bayes ranker.

Krawczyk and Wozniak [19] propose a modification to the naive Bayes classifier for mining streams in a nonstationary environment in the presence of the concept drift phenomenon [17]. Their algorithm works by fitting a naive Bayes classifier on samples they deem essential and removing unnecessary and outdated examples that no longer represent the present state of the analysed data stream;

this approach contrasts with our algorithm. We do not focus our learning on retaining relevant training data. Instead, we forecast one-step-ahead posterior ranking probabilities using all test data processed as a stream. Further details of our meta-algorithm are provided next.

3 THE NAIVE BAYES ASSET RANKER

Our ranking algorithm is the naive Bayes asset ranker (nbar), which is succinctly displayed in algorithm 3. The algorithm sequentially ranks a set of experts; it does so by forecasting the one-step-ahead posterior probability that individual experts will be ranked higher than the set of available experts at its disposal. In the context of the experiment described in section 4, each expert is a forecasted return for an individual portfolio constituent of the S&P 500. The forecasted returns come from the caw multivariate regression model, which utilises feature representation transfer from the constituent S&P 500 returns to rbfnets whose k-means++ [2] clusters form hidden units. Assume that the algorithm is presented with a set of q forecasts. The goal is to select a subset of experts $1 \leq k \leq q$ such that the reward of the k experts is expected to be the highest; this is achieved by estimating the sequential posterior probability that expert $j \in 1, ..., q$ is ranked higher than each of the remaining $q-1$ experts. This posterior probability is computed with exponential decay, allowing experts who performed poorly and now perform well to be selected with greater weight than previously. The inputs to the algorithm are $\alpha = k/q$, with $k \leq q$, the α-percentile subset of assets to trade, and τ, an exponential half-life. Denote as

$$p(\mathbf{r}_{j,t} \geq \mathbf{r}_t) = p(\mathbf{a}_t) = \prod_{i=1}^{q} p(\mathbf{r}_{j,t} \geq \mathbf{r}_{i,t}),$$

the probability that the forecasted returns of asset j will be ranked higher than the q assets considered, where $\mathbf{r}_t \in \mathbb{R}^q$ is the vector of forecasts at time t. Furthermore, denote as $p(\mathbf{r}_{j,t} > 0) = p(\mathbf{b}_t)$, the probability that asset j has a forecasted return at time t that is greater than zero; this condition is required so that we do not naively select k assets to go long if there are fewer assets with expected positive returns. The sequential posterior probability that algorithm 3 computes is

$$p(\mathbf{a}_t|\mathbf{b}_t) = \frac{p(\mathbf{b}_t|\mathbf{a}_t)p(\mathbf{a}_t)}{p(\mathbf{b}_t|\mathbf{a}_t)p(\mathbf{a}_t) + p(\mathbf{b}_t|\mathbf{a}_t^c)p(\mathbf{a}_t^c)}. \tag{3}$$

Finally, the algorithm returns the set of indices that would sort $p(\mathbf{a}_t|\mathbf{b}_t)$ from largest to smallest.

4 THE RESEARCH EXPERIMENT

Our research experiment aims to assess the benefits of sequentially optimised ranking algorithms to select subsets or portfolios of financial assets to hold in either a long-only or long/short (cross-sectional momentum) capacity. More concretely, we experiment with the constituents of the S&P 500 index. We use our nbar algorithm as the sequentially optimised ranker, with the posterior ranking probabilities of equation 3 estimated continuously during the test set. The nbar inputs are the one-step-ahead predicted daily returns estimated by the caw multivariate regression model. In turn, the caw model collects individual one-step-ahead predicted daily returns from rbfnets, one per S&P 500 constituent. The rbfnets have hidden units whose means are determined via k-means++; thus, we perform feature representation transfer from external inputs, the S&P 500 constituent daily returns, to hidden unit outputs determined by clustering algorithms. These hidden unit outputs are mapped to the response, individual constituent one-step-ahead daily return forecasts, using exponentially weighted recursive least-squares. A schematic in the form of graphical probability models is shown in figure 1.

Algorithm 3: The naive Bayes asset ranker

Input: $\mathbf{r}_t$, τ, α
Initialise: $\mathbf{s} = \mathbf{1}_q$, $\mathbf{S} = \mathbf{1}_{q \times q}$, $\mathbf{p} = \mathbf{0}_q$, $n = 0$, $k = \lfloor \alpha q \rfloor$
// $\mathbf{r}_t$ are the forecast one-step-ahead daily returns from the caw algorithm 1
// τ is an exponential decay constant
// α is the maximum percentile of experts that can be chosen
Output: $\mathbf{z}_t = argSort(\mathbf{p}_t)$
if $\tau = 1$ **then**
  $n = n + 1$
  $w_t = (n-1)/n$
else
  $w_t = \tau$
// $I(.)$ is an indicator function that returns 1 for a true condition, or else 0
for $j \leftarrow 1$ **to** q **do**
  $\mathbf{s}_{j,t} = w_t \mathbf{s}_{j,t} + (1 - w_t) I(\mathbf{r}_{j,t} \geq 0)$
  for $i \leftarrow 1$ **to** q **do**
    $\mathbf{S}_{ij,t} = w_t \mathbf{S}_{ij,t} + (1 - w_t) I(\mathbf{r}_{j,t} \geq \mathbf{r}_{i,t})$
Initialise zero vectors $\mathbf{a} = \mathbf{b} = \mathbf{0}_q$
for $j \leftarrow 1$ **to** q **do**
  $\mathbf{b}_j = \mathbf{s}_{j,t} / \sum_{i=1}^{q} \mathbf{s}_{i,t}$
  $\mathbf{a}_j = \frac{\sum_{i=1}^{q} \mathbf{S}_{ij,t}}{\sum_{h=1}^{q} \sum_{k=1}^{q} \mathbf{S}_{hk,t}}$
$\mathbf{a}^c = 1 - \mathbf{a}$
$\mathbf{p}_t = \frac{\mathbf{ba}}{\mathbf{ba} + \mathbf{ba}^c}$
// $argSort(.)$ returns the indices that would sort an array from largest to smallest value
$\mathbf{z}_t = argSort(\mathbf{p}_t)$
Denote $\mathbf{z}_{j,t} = j^*$, the test time return is $\sum_{j=1}^{k} \mathbf{r}_{j^*,t+1} \mathbf{p}_{j^*,t}$.

At first glance, this meta-model setup might seem overly or unnecessarily complicated. We could, for example, use the raw S&P 500 constituent daily returns as inputs to the nbar. However, we are building on the research of regress-then-rank algorithms in the context of portfolio selection [13, 16], which motivates our use of the rbfnets. We are also building on the work of Breiman and Friedman [7] to take advantage of the correlations between the response variables to improve predictive accuracy compared with the usual procedure of doing individual regressions of each response variable on the shared set of predictor variables. We use their caw procedure but combine it with exponentially weighted recursive least-squares to facilitate sequential optimisation in the test set without forward-looking bias. Finally, we utilise the research of Borrageiro et al. [4] to make use of online learning rbfnets, as these models retain more remarkable knowledge of the input feature

space. They also respond better to regime changes or concept drifts than models that do not use feature representation transfer; for example, from clustering algorithms [4], Gaussian mixture models [6] or echo state networks [5].

4.1 Baseline models

In order to assess the true value of the meta-model shown in figure 1, we adopt two baseline models. The first model intuitively is the long-only holding of the S&P 500 constituents with equal weighting and replicates a passive, index-tracking investment strategy. A second baseline is our proxy for the regress-then-rank models, the sequentially optimised caw multivariate regression model, algorithm 1. This baseline also uses the individual predicted daily returns from online learning rbfnets.

4.2 The S&P 500 dataset

We conduct this research experiment using the daily closing constituent prices for the S&P 500 index, which we extract from Refinitiv. Due to their relatively new trade history, some time series have little data. Therefore, we select a subset of the S&P 500 index, where each constituent contains a trade count greater than or equal to the 25'th percentile of trade counts; this leaves us with a subset of 378 Refinitiv information codes (rics). The dataset begins on 2001-01-26 and ends on 2022-03-25, 5326 days.

4.3 Experiment design

We use the first 25% of the data as a training set and the remaining data as a test set. In the training set, algorithms 1 and 3 are initialised and fitted. These models are also sequentially optimised without forward-looking bias in the test set. The hyperparameters that are set for this experiment are:

- Exponential decay, $\tau = 0.99$.
- Ridge penalty, $\lambda = 0.001$.
- Radial basis function networks with 500 hidden units determined by k-means++.
- Maximum percentile of assets to trade either long or short, $\alpha = 0.05$.

Once the training data are assigned to their nearest cluster centres, the cluster-conditional covariance matrices and their inverses are estimated. Cluster centres with few training data vectors assigned to them are regularised to a diagonal variance prior. Thus, we are adopting a Bayesian maximum a posteriori procedure here.

We use the forecasts of the caw model, algorithm 1, as the basis for taking risk in a subset of constituents in the S&P 500 index. Specifically, the long-only caw model buys the expected top five per cent of performing assets $\tilde{\mathbf{y}}_t$. For example, if there are k assets in this top five percentile, then a weight of $1/k$ is applied per constituent. The long/short caw model works similarly, except that it includes the short-selling of the bottom five per cent of most negative forecasts, with a weight of $-1/k$.

A second forecaster we consider is the nbar algorithm 3, applied to the one-step-ahead forecasts of the caw model. Algorithm 3 outputs $\mathbf{z}_t$, the set of indices that would sort $\mathbf{p}_t$ in descending order, which is the posterior probability of highest ranked assets. Denoting $\mathbf{z}_{j,t} = j^*$ and assuming there are k assets in the expected top five per cent of performing assets, the nbar assigns a weight to the $j'th$ constituent $(1 \leq j \leq k)$ of

$$\mathbf{p}_{j^*,t} \Big/ \sum_{i^*=1}^{k} \mathbf{p}_{i^*,t}. \tag{4}$$

Similarly, for short positions, assuming there are k assets we wish to go short, the weight assigned to the $j'th$ constituent $(q-k \leq j \leq q)$ is

$$-\left(1 - \mathbf{p}_{j^*,t}\right) \Big/ \left(\sum_{i^*=q-k}^{q} 1 - \mathbf{p}_{i^*,t} \right). \tag{5}$$

We must also consider execution costs. We force the caw and nbar models to trade as price takers, meaning that the models incur a cost equal to half the bid/ask spread times the change in absolute position. Specifically, we apply the average transaction cost per S&P 500 constituent, whose distribution of relative basis point costs

$$bpcost_j = \frac{10000}{T} \sum_{t=1}^{T} \frac{ask_{j,t} - bid_{j,t}}{mid_{j,t}},$$

is shown in figure 3. Furthermore, as these data are sampled daily,

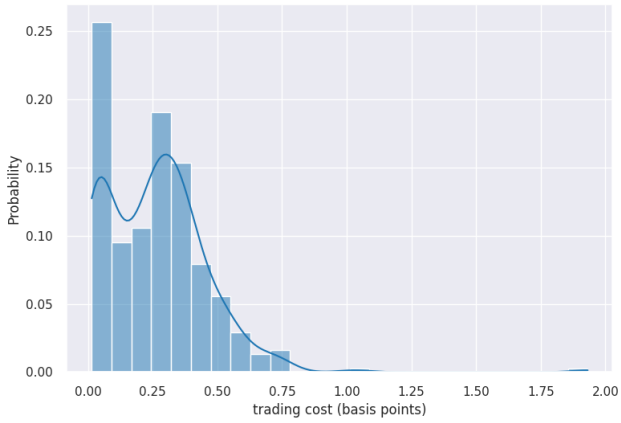

Figure 3: The distribution of transaction costs, where the distribution is taken over the average transaction cost per S&P 500 constituent.

any portfolio rebalancing is applied at most once a day, at the close of trading circa 4 pm EST.

4.4 Results

The passive index tracking baseline purchases each constituent with equal weighting at $t = 0$ and holds them till the end of the experiment. This strategy pays transaction costs once and therefore has the least fees, as shown in table 1. Table 2 and figure 4 also show that the cumulative returns generated by this strategy are 205%, the compound annual growth rate (cagr) is 7.3% and the risk-adjusted annualised Sharpe ratio (sr) is a little under 0.8. Assuming normally distributed returns, the Sharpe ratio implies a probability of positive annual returns of 71%. The largest peak-to-trough drawdown for the strategy is just under 72%, and the total return to maximum drawdown is around 2.9. Finally, by simply holding the index, the percentage of days with positive returns is 55%. The same

performance metrics are available for the caw and nbar models. Both long-only and long/short caw and nbar models outperform the passive index tracking baseline, with the long/short models showing higher risk-adjusted performance measures indicated by the Sharpe ratios. The nbar performs best, with the long/short nbar showing the highest total and risk-adjusted returns. Table 1 shows that despite the caw and nbar models being actively managed strategies that rebalance the portfolios daily, only the caw models show high transaction costs. The nbar models rebalance less often and do a better job of picking portfolio constituents.

Figure 5 shows a bird's eye view of the nbar cross-sectional momentum weights across time. Long positions show up as dark blue specks, and short positions show up as yellow specks. We see evidence of the nbar dynamically shifting weights over time to find the best candidates to hold on a cross-sectional momentum basis, given the fixed constraint that a maximum of five per cent of total assets can be held long or short. The weight range in figure 5 indicates a relatively diffuse weight choice; in other words, no single constituent appears to dominate the others regarding predicted returns performance. We can zoom into a specific portfolio constituent, namely Electronic Arts Inc. In figure 6, we see that the long/short nbar switches between long, short and flat positions as necessary, without an exponential decay of the weights permanently, as with the weighted majority algorithm.

Figure 7 displays the sensitivity of total returns and Sharpe ratios to the selection percentile, that is, the fraction of assets that are held in either a long-only or long/short manner. We draw similar conclusions with the fixed experiment that selects five per cent of the expected best-performing assets. The nbar models perform best, and the long/short models perform better than their long-only counterparts. The total return increases as fewer assets are selected, particularly for the nbar, which shows a ×5 improvement over the baseline when trading just a pair of assets in a long/short manner. However, such a strategy is not scalable if a large amount of investment capital needs to be allocated to it. Furthermore, we have not modelled the trade impact that would invariably appear if we were executing prominent positions relative to each asset's average daily volume turnover. Figure 7 also shows that the Sharpe ratios increase toward a selection percentile of around five per cent and, depending on the model, decrease or plateau after that.

Table 1: Relative transaction costs incurred by each model in the test set. A buy-and-hold strategy on the S&P 500 achieves the lowest transaction costs. However, from the perspective of a more active portfolio management standpoint, our ranking algorithm incurs far lower transaction costs than the regress-then-rank baseline.

	transaction costs
long S&P 500	**-0.003**
long caw	-0.933
long/short caw	-1.966
long nbar	-0.050
long/short nbar	-0.104

Figure 4: Total return by each model in the test set where the maximum selection percentile is set to 5% of the total number of portfolio constituents. The naive Bayes asset ranker performs best, particularly the cross-sectional momentum version.

Table 2: Summary returns statistics are shown in relation to the experiment, shown visually in figure 4. The cross-sectional momentum naive Bayes asset ranker has the highest total and risk-adjusted returns.

	long S&P 500	long caw	long nbar	long short caw	long short nbar
mean	0.0005	0.001	0.0013	0.0009	0.0015
std	0.012	0.016	0.016	0.010	0.010
total ret	2.047	4.113	5.372	3.397	**5.806**
cagr	0.073	0.108	0.124	0.098	0.128
sr	0.798	1.243	1.636	1.624	**2.879**
$pr(\text{ann. ret} > 0)$	0.71	0.816	0.895	0.893	0.994
max dd	0.717	0.64	0.646	0.942	0.202
total ret / max dd	2.853	6.423	8.311	3.607	28.7
win ratio %	0.549	0.553	0.563	0.547	0.575

5 DISCUSSION

The results show that the caw models outperform the baseline, and the nbar models have the best overall performance. The caw models apply a multivariate regression technique, and the nbar models rank assets sequentially using posterior probabilities. How might we rationalise this performance ordering? Several academic papers discuss the shortcomings of regression models compared to classification models in the financial time series prediction setting. For example, Satchell and Timmermann [32] show that regression models that typically minimise prediction mean-square error (mse) obtain worse performance than a random-walk model when forecasting daily foreign exchange (fx) returns. Furthermore, they show that the probability of correctly predicting the sign of the change in daily fx rates is higher for the regression models than the random-walk baseline, even though the mse of the regression

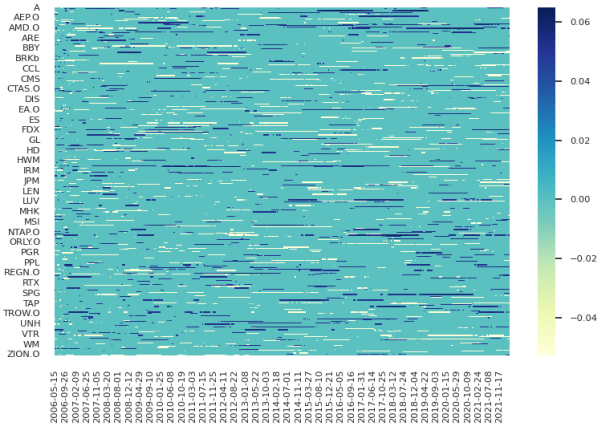

Figure 5: The naive Bayes asset ranker cross-sectional momentum weights across time. We find visual evidence that the portfolio selection is dynamic and changing over time.

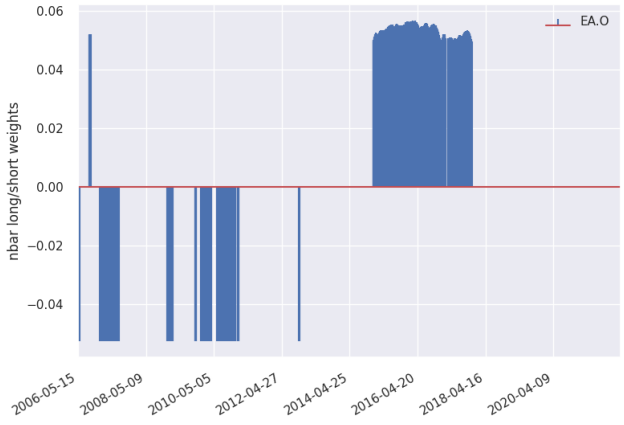

Figure 6: Naive Bayes asset ranker cross-sectional momentum weights across time for Electronic Arts Inc. Unlike the weighted majority algorithm, which assigns an expert maximum weight, which then erodes as the expert makes mistakes, our algorithm allows the expert to be selected with greater absolute weight in future, irrespective of how poor it performed previously.

models exceeds that of the random-walk model. They conclude that mse is not always an appropriate performance measure for evaluating predictive performance. More recently, Amjad and Shah [1] find that classical time series regression algorithms, such as arima models, have poor performance when forecasting Bitcoin returns. However, they find that the probability distribution of the sign of future price changes is adequately approximated from finite data, specifically classification algorithms that estimate this conditional probability distribution.

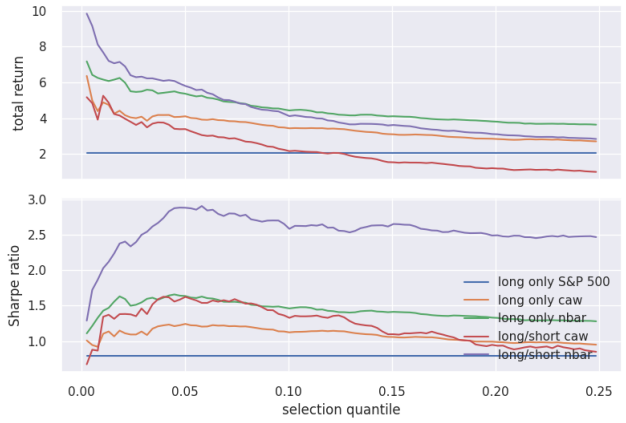

Figure 7: Test returns by model and selection percentile. Restricting the maximum selection percentile results in the highest total returns but is not particularly useful for portfolio managers that need to allocate substantial investment capital. The risk-adjusted returns for this test set peak near an upper-bound selection percentile of 5% of total constituents.

6 CONCLUSIONS

We extend the research into cross-sectional momentum trading strategies. Our main result is our novel ranking algorithm, the naive Bayes asset ranker (nbar), which we use to select subsets of assets to trade from the S&P 500 index. We perform feature representation transfer from radial basis function networks to a curds and whey (caw) multivariate regression model that takes advantage of the correlations between the response variables to improve predictive accuracy. The nbar ranks this regression output by forecasting the one-step-ahead sequential posterior probability that individual assets will be ranked higher than other portfolio constituents. Earlier algorithms, such as the weighted majority, deal with nonstationarity by ensuring the weights assigned to each expert never dip below a minimum threshold without ever increasing weights again. Our ranking algorithm allows experts who previously performed poorly to have increased weights when they start performing well. Our algorithm outperforms a strategy that would hold the long-only S&P 500 index with hindsight, despite the index appreciating by 205% during the test period. It also outperforms a regress-then-rank baseline, the caw model.

REFERENCES

[1] Muhammad Amjad and Devavrat Shah. 2017. Trading Bitcoin and Online Time Series Prediction. In *Proceedings of the Time Series Workshop at NIPS 2016*, Vol. 55. PMLR, Barcelona, Spain, 1–15.

[2] David Arthur and Sergei Vassilvitskii. 2007. K-Means++: The Advantages of Careful Seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics, New Orleans, Louisiana, 1027–1035.

[3] Chris M. Bishop. 1995. *Neural networks for pattern recognition*. Oxford University Press, Oxford.

[4] Gabriel Borrageiro, Nick Firoozye, and Paolo Barucca. 2021. Online Learning with Radial Basis Function Networks.

[5] Gabriel Borrageiro, Nick Firoozye, and Paolo Barucca. 2022. The Recurrent Reinforcement Learning Crypto Agent. *IEEE Access* 10 (2022), 38590–38599.

[6] Gabriel Borrageiro, Nick Firoozye, and Paolo Barucca. 2022. Reinforcement Learning for Systematic FX Trading. *IEEE Access* 10 (2022), 5024–5036.

[7] Leo Breiman and Jerome H. Friedman. 1997. Predicting Multivariate Responses in Multiple Linear Regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 59, 1 (1997), 3–54.

[8] Nicolo Cesa-Bianchi and Gábor Lugosi. 2006. *Prediction, Learning, and Games*. Cambridge University Press, USA.

[9] Thomas M. Cover. 1991. Universal Portfolios. *Mathematical Finance* 1, 1 (1991), 1–29.

[10] Thomas M. Cover and Erik Ordentlich. 1996. Universal portfolios with side information. *IEEE Transactions on Information Theory* 42, 2 (1996), 348–363.

[11] Peter Flach and Edson Takashi Matsubara. 2007. *A Simple Lexicographic Ranker and Probability Estimator*. Springer Berlin Heidelberg, Berlin, Heidelberg. 575–582 pages.

[12] Clive W. Granger and Paul Newbold. 1974. Spurious regressions in econometrics. *Journal of econometrics* 2 (1974), 111–120. Issue 2.

[13] Shihao Gu, Bryan Kelly, and Dacheng Xiu. 2020. Empirical Asset Pricing via Machine Learning. *The Review of Financial Studies* 33, 5 (02 2020), 2223–2273.

[14] David P. Helmbold, Robert E. Schapire, Yoram Singer, and Manfred K. Warmuth. 1998. On-Line Portfolio Selection Using Multiplicative Updates. *Mathematical Finance* 8, 4 (1998), 325–347.

[15] Herbert Jaeger. 2002. Adaptive Nonlinear System Identification with Echo State Networks. In *Advances in neural information processing systems*, Vol. 15. Advances in neural information processing systems, Vancouver, British Columbia, 1000–1008.

[16] Narasimhan Jegadeesh and Sheridan Titman. 1993. Returns to Buying Winners and Selling Losers: Implications for Stock Market Efficiency. *The Journal of Finance* 48 (1993), 65–91. Issue 1.

[17] Gama Joao. 2012. A survey on learning from data streams: current and future trends. *Progress in AI* 1, 1 (2012), 45–55.

[18] Jyrki Kivinen and Manfred K. Warmuth. 1995. Additive versus Exponentiated Gradient Updates for Linear Prediction. In *Proceedings of the Twenty-Seventh Annual ACM Symposium on Theory of Computing* (Las Vegas, Nevada, USA). ACM, New York, NY, USA, 209–218.

[19] Bartosz Krawczyk and Michal Wozniak. 2015. Weighted Naïve Bayes Classifier with Forgetting for Drifting Data Streams. In *2015 IEEE International Conference on Systems, Man, and Cybernetics*. IEEE, Hong Kong, 2147–2152.

[20] Nick Littlestone and Manfred K. Warmuth. 1994. The Weighted Majority Algorithm. *Information and Computation* 108 (1994), 212–261.

[21] Stuart Lloyd. 1982. Least squares quantization in PCM. *IEEE transactions on information theory* 28, 2 (1982), 129–137.

[22] Harry M. Markowitz and Paul H. Cootner. 1965. The Random Character of Stock Market Prices. *J. Amer. Statist. Assoc.* 60, 309 (1965), 381–381.

[23] Robert C. Merton. 1976. Option pricing when underlying stock returns are discontinuous. *Journal of Financial Economics* 3, 1 (1976), 125 – 144.

[24] John Moody and Christian J. Darken. 1989. Fast Learning in Networks of Locally-Tuned Processing Units. *Neural computation* 1 (1989), 281–294. Issue 2.

[25] Tomomichi Nakamura and Michael Small. 2007. Tests of the random walk hypothesis for financial data. *Physica A: Statistical Mechanics and its Applications* 377, 2 (2007), 599–615.

[26] Sinno J. Pan and Qiang Yang. 2010. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering* 22 (2010), 1345–1359.

[27] Daniel Poh, Bryan Lim, Stefan Zohren, and Stephen Roberts. 2021. Building Cross-Sectional Systematic Strategies by Learning to Rank. *The Journal of Financial Data Science* 3, 2 (2021), 70–86.

[28] Daniel Poh, Bryan Lim, Stefan Zohren, and Stephen Roberts. 2022. Enhancing Cross-Sectional Currency Strategies by Context-Aware Learning to Rank with Self-Attention. *The Journal of Financial Data Science* 4, 3 (2022), 89–107.

[29] Foster Provost and Tom Fawcett. 1997. Analysis and Visualization of Classifier Performance: Comparison under Imprecise Class and Cost Distributions. In *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining*. AAAI Press, Newport Beach, CA, 43–48.

[30] Alexander Rakhlin and Karthik Sridharan. 2014. *Statistical Learning Theory and Sequential Prediction.* Technical Report. MIT. STAT928.

[31] Said E. Said and David A. Dickey. 1984. Testing for unit roots in autoregressive-moving average models of unknown order. *Biometrika* 71, 3 (12 1984), 599–607.

[32] Steve Satchell and Allan Timmermann. 1995. An assessment of the economic value of non-linear foreign exchange rate forecasts. *Journal of Forecasting* 14, 6 (1995), 477–497.

[33] Yoram Singer. 1998. Switching Portfolios. In *International Journal of Neural Systems*. Morgan Kaufmann, Burlington, Massachusetts, 488–495.

[34] George W. Snedecor and William G. Cochran. 1989. *Statistical Methods, eight edition*. Vol. 1191. Iowa state University press, Ames, Iowa.

[35] R. S. Sutton and A. G. Barto. 2018. *Reinforcement Learning: An Introduction.* A Bradford Book, Cambridge, MA, USA.

[36] Qiang Yang, Yu Zhang, Wenyuan Dai, and Sinno J. Pan. 2020. *Transfer Learning*. Cambridge University Press, Cambridge.

[37] Harry Zhang and Jiang Su. 2004. Naive Bayesian Classifiers for Ranking. In *Machine Learning: ECML 2004*, Jean-François Boulicaut, F. Esposito, F. Giannotti, and D. Pedreschi (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 501–512.