The Big Data Newsvendor Problem in a Portfolio Optimization Context

Thierry Bazier-Matte

Summer 2015

Abstract

Following [1], we provide a portfolio optimization method based on machine learning methods.

1 Introduction

This document considers a two-asset portfolio, of which one is the risk-free asset, yielding a constant return rate R_f, and the other being a risky asset s, typically a stock, yielding a random return rate r_{st} for each period t. We suppose that each risky asset s can be decribed daily by an *information vector* x_{st} containing potentially useful information, such as technical, fundamental or news-related information. Furthermore, we assume that the allocation of each asset of the portfolio p_{st} can be fully determined using a *decision vector* q. The allocation rule is the folowing: $q^T x_{st}$ is allocated to the risky asset and $1 - q^T x_{st}$ is allocated to the risk-free asset. Over the period t, the portfolio p_{st} consisting of asset s will therefore yield a return rate of:

Maybe considerations about the length of the period should be added? For example, it's not specified what's the period length of R_f.

$$p_{st}(q) = r_{st} q^T x_{st} + (1 - q^T x_{st}) R_f. \tag{1}$$

The question we now wish to ask is how the decision vector q should be chosen. We assume we have access to a training dataset S_n, comprising of s different assets over t periods, such that $n = s \times t$.

What's the difference between having n points with having $s \times t$ points? For example, what if $s \gg t$ or the reverse?

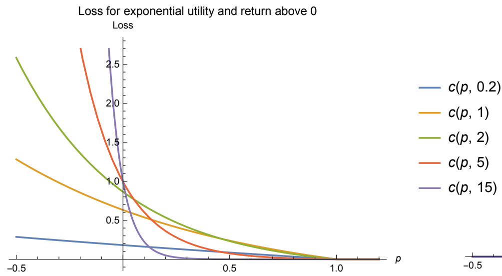
2 Definitions and Bounds

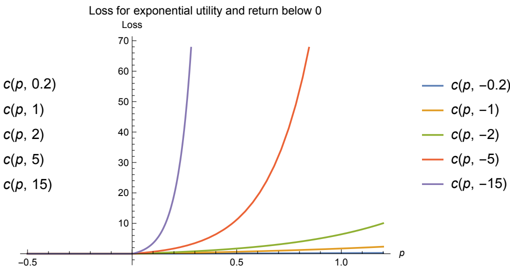
2.1 Definitions and Notation

Most of the following notation and defintions follow directly from [2].

Let S_n be a set of n vectors of $\boldsymbol{R}^p \times \boldsymbol{R}$ of the form:

$$S_n = \{(x_1, r_1), \ldots, (x_n, r_n)\}. \tag{2}$$





Each component of S_n is a tuple (x, r), where x is the information vector and r is the observed return rate.

Using S_n, we wish to create a decision vector $q_{S_n} \in \boldsymbol{R}^p$ from which we can make an investment decision when confronted with a random draw $d = (x, r)$.

Loss and Cost. We introduce the loss ℓ and the cost c of using q with a random draw $d = (x, r)$:

$$\ell(q, d) = c(q(x), r) = c(q^T x, r) \tag{3}$$

The cost must always be a non-negative quantity. Supposing an utility U, we model it as follows:

$$c(p, r) = \begin{cases} \lfloor U(r) - U(pr + (1-p)R_f) \rfloor & \text{if } r > R_f \\ \lfloor U(R_f) - U(pr + (1-p)R_f) \rfloor & \text{if } r \leq R_f \end{cases} \tag{4}$$

By $\lfloor . \rfloor$ we mean a function returning its argument if non-negative and zero otherwise. This means that we don't want to discourage taking risk (borrowing or short-selling), but it's not encouraged either.

Utility. There are two ways we can model our utility, and both are concave shaped, to represent a risk-averse approach. The first utility is the linear utility of the form

$$U(r) = r + \min(0, \beta r), \tag{5}$$

with $0 < \beta < 1$. The other utility is exponential:

$$U(r) = -\exp(-\mu r), \tag{6}$$

with $\mu > 0$.

Algorithm. We will be concerned with probabilistic confidence bounds on results produced using the following algorithm, using dataset S_n.

$$q^\star = \arg\min_{q \in \boldsymbol{R}^p} \frac{1}{n} \sum_{i=1}^{n} c(q^T x_i, r_i) + \lambda \|q\|_2^2. \tag{7}$$

Assumptions. We will assume that information vectors have been pre-processed and lie in a $X^2_{\max}$ radius ball. We also assume that the return rates observed are comprised within $[-\bar{r}, \bar{r}]$. This last assumption will be relaxed.

Include reference for definitions and theorems

Definition. A loss function ℓ is σ-admissible if the associated cost function c is convex with respect to its first argument and the following condition holds for any p_1, p_2 and r:

$$|c(p_1, r) - c(p_2, r)| \leq \sigma |p_1 - p_2| \tag{8}$$

Remark. Our loss function ℓ is σ-admissible with $\sigma = \bar{r} + R_f$ in the linear case and $\sigma = (\bar{r} + R_f) \exp(\mu \bar{r})$ in the exponential case.

Proof. First, we remark that both forms of U yield a convex function of p with r fixed.

Now we'll suppose that $c(p_1, r), c(p_2, r) > 0$. Then the expression $|c(p_1, r) - c(p_2, r)|$ reduces to

$$|U(p_1 r + (1 - p_1)R_f) - U(p_2 r + (1 - p_2)R_f|. \tag{9}$$

Now because $r \in [-\bar{r}, \bar{r}]$, U is Lipschitz continuous on its domain, and so (9) is bounded by

$$\alpha |p_1 r + (1 - p_1)R_f - (p_2 r + (1 - p_2)R_f)| = \alpha |p_1 - p_2||r - R_f| \tag{10}$$

where

$$\alpha = \sup_{r \in [-\bar{r}, \bar{r}]} |U'(r)|. \tag{11}$$

In the linear case, the derivative is piecewise constant, and is set to 1 on for returns below r_c, so that $\alpha = 1$. In the exponential case, $U'(r) = \exp \mu r$, and $\alpha = \exp \mu \bar{r}$.

The bound (10) must hold for any r. The expression $|r - R_f|$ will reach its largest value at $r = -\bar{r}$, since R_f is assumed to be non-negative.

Finally we consider the case where, without loss of generality, $c(p_2, r) = 0$. Then, if c had not been defined using $\lfloor . \rfloor$, then we would have

$$\begin{aligned} |\lfloor c(p_1, r) \rfloor - \lfloor c(p_2, r) \rfloor| &\leq |c(p_1, r) - c(p_2, r)| \quad (12) \\ &\leq \sigma |p_1 - p_2|. \end{aligned}$$

□

Theorem 1. *Let F be a reproducing kernel Hilbert space with kernel κ that $\forall x \in X$, $\kappa(x, x) \leq \kappa^2 < \infty$. If ℓ is σ-adimissible with respect to F, then the learning algorithm defined by*

$$A_S = \arg\min_{g \in F} \frac{1}{n} \sum_{i=1}^{n} \ell(g, d_i) + \lambda \|g\|_k^2 \tag{13}$$

has uniform stability α_n with respect to ℓ with

$$\alpha_n \leq \frac{\sigma^2 \kappa^2}{2 \lambda n}. \tag{14}$$

Remark. Our proposed algorithm has the form (13), and so has algorithmic stability bounded by

$$\alpha_n \leq \frac{(\bar{r} + R_f)^2 X_{\max}^2}{2\lambda n} \tag{15}$$

with linear utility and

$$\alpha_n \leq \frac{\exp(2\mu\bar{r}) X_{\max}^2}{2\lambda n} \tag{16}$$

in the case of exponential utility.

Definition. The *true risk* with respect to algorithm A and set S_n is defined as

$$R_{\text{true}}(A, S_n) = E_d[\ell(A_{S_n}, d)], \tag{17}$$

which is, in plain words, the expected loss incured when applying the algorithm created from training set S_n in the wild, ie. out of sample.

Definition. The *empirical risk* with respect to algorithm A and set S_n is defined as

$$\hat{R}(A, S_n) = \frac{1}{n} \sum_{i=1}^{n} \ell(A_{S_n}, d_i), \tag{18}$$

which is, in plain words, the average cost incured by our model over all the training set.

Remark. The maximum loss we can suffer over a single data point happens when $r_i = -\bar{r}$ and $p = 1$, ie.

$$c(1, -\bar{r}) = U(R_f) - U(\bar{r}). \tag{19}$$

We shall call this quantity γ.

Theorem 2. *Let A be an algorithm with uniform stability α_n with respect to a loss function ℓ such that $0 \leq \ell(A_{S_n}, d) \leq M$ for all $d = (x, r) \sim D$ and all sets S_n of size n. Then for any $n \geq 1$ and any $\delta \in (0, 1)$, the following bound holds with probability at least $1 - \delta$ over the random draw of the sample S_n:*

$$|R_{true}(A, S_n) - \hat{R}(A, S_n)| \leq 2\alpha_n + (4n\alpha_n + M)\sqrt{\frac{\log(2/\delta)}{2n}}. \tag{20}$$

Remark. Our alogirthm has a generalization bound of

$$|R_{\text{true}}|(A, S_n) - \hat{R}(A, S_n)| \leq 2\alpha_n + (4n\alpha_n + \gamma)\sqrt{\frac{\log(2/\delta)}{2n}}. \tag{21}$$

References

[1] Cynthia Rudin and Gah-Yi Vahn. *The Big Data Newsvendor: Pratical Insights from Machine Learning*, Operations Research, 2015.

[2] Olivier Bousquet and André Elisseeff. *Stability and Generalization*, Journal of Machine Learning Research, 2002.

[3] "Si la valeur absolue de la dérivée est majorée par k, f est k-lipschitzienne". Application lipschitzienne.

[4] Rockafellar, R. T. *Convex Analysis*, Princeton University Press, 1970.

[5] Supergradients.

[6] Reference needed!