Backtest Analyse der "Rule of 40" Strategie

Im vorliegenden Research Notebook wollen wir die Backtest Resultate der Rule-of-40 Strategie näher unter die Lupe nehmen. Dabei wollen wir u.a. die folgenden Fragen beantworten:

1. Welche Aktien waren die größten Gewinner/Verlierer?
2. Gibt es einen statistischen Zusammenhang zwischen dem Efficiency Score und der zukünftigen Rendite?
3. Wie entwickelten sich einzelne Positionen im zeitlichen Verlauf?

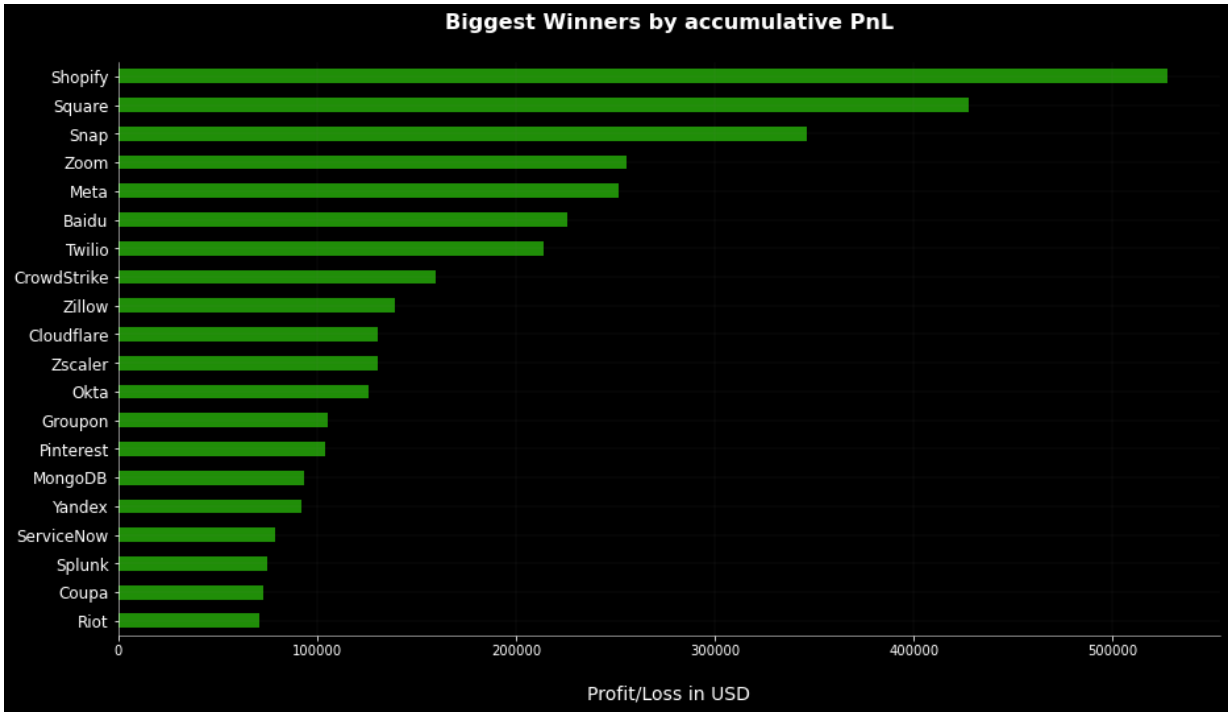
In [1]:
```
from research_tools import BacktestAnalyzer

qb = QuantBook()
backtest = api.ReadBacktest(9842406, 'c6230dcf97951faeee63daa6ec1c26c4')
bta = BacktestAnalyzer(backtest)
```

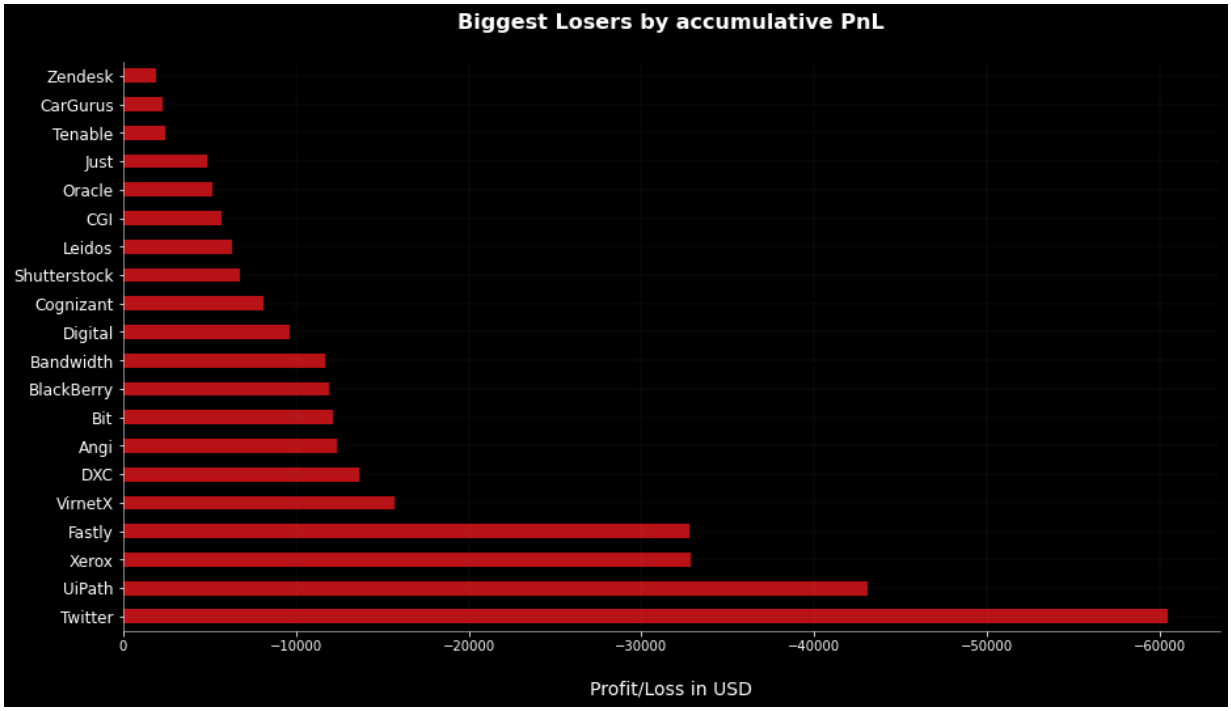
1. Welche Aktien waren die größten Gewinner und Verlierer und hatten damit den größten Einfluss auf die Performance?

Die Liste der Gewinner dürfte die Wenigsten überraschen. Unternehmen wie Shopify, Square und Zoom führen die Liste an. Das deckt sich auch mit der Tatsache, dass die Post COVID-Phase in 2020/21 die mit Abstand stärksten Jahre für die Strategie waren (siehe Backtest-Report). Das ist genau der Zeitraum in dem diese Unternehmen besonders gut performten.

Twitter führt die Liste der Verlierer-Positionen an. Dies hängt vor allem auch mit dem Zeitfenster zusammen in der die Strategie in Twitter investiert war. Mehr dazu weiter unten.

In [2]:
```
bta.PlotBiggestWinnersAndLosers()
```





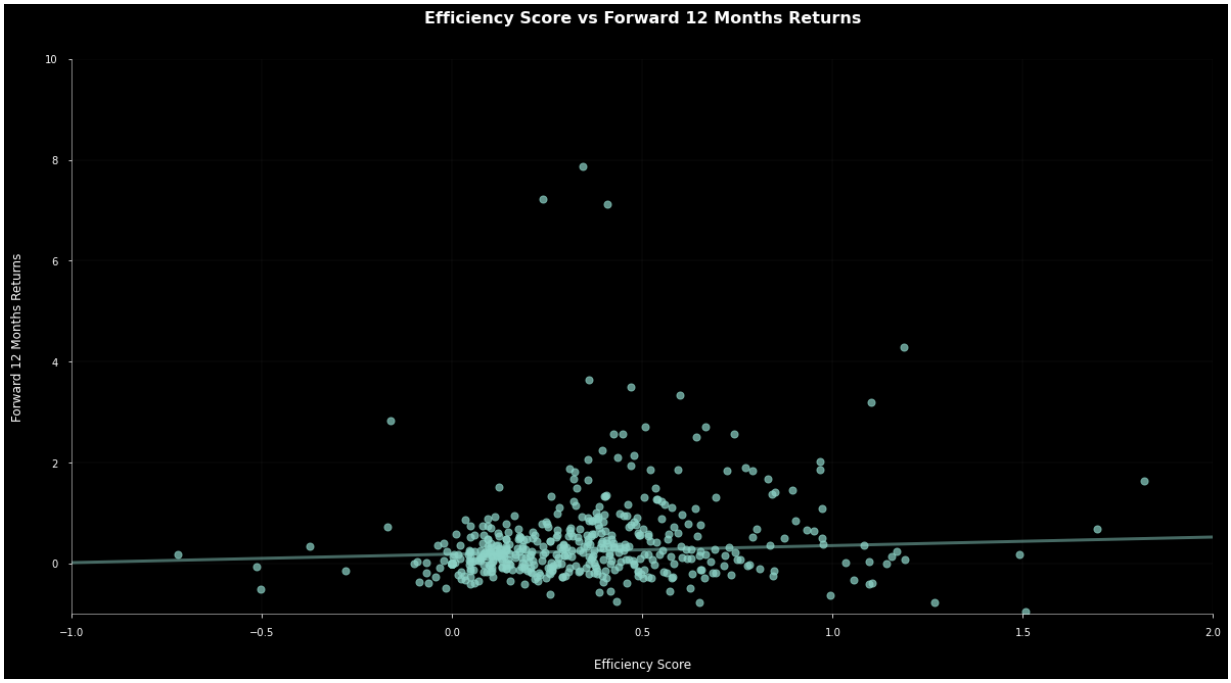
2. Gibt es einen statistischen Zusammenhang zwischen Efficiency Score und der zu erwartenden Rendite?

Die Frage kann mit einem *vorsichtigen* Ja beantwortet werden wie die Grafik unten veranschaulicht. Es ist eine leichte positive Korrelation erkennbar. Allerdings mit großen Ausreißern (einige sind zum Zwecke einer besseren Darstellung nicht sichtbar). Zudem ist der Zusammenhang gewiss nicht linearer Natur, d.h. die Aussage "Je größer der Efficiency Score, desto größer die Rendite" wäre so definitiv nicht korrekt. Die meisten Datenpunkte häufen sich um die Nulllinie.

Folgenden Punkt sollten man beachten und ist auch der Grund dafür warum man mit Schlussfolgerungen vorsichtig sein sollte:

Der Backtest fand über den Zeitraum 2010-heute stand, also in einem Bullenmarkt, wo die meisten Aktien gestiegen sind. Daher befinden sich mehr Datenpunkte im positiven Renditebereich und tragen so zu einer Verzerrung der Daten bei. Streng genommen müsste man also bei allen Punkten eine geeignete Benchmark-Rendite abziehen. Das würde sicherlich dazu führen, dass die Regressionsgerade ihre ohnehin schon schwache Steigung vollständig verliert.

In [3]:
```
bta.PlotEfficiencyScoreVsFwd12MonthsReturns()
```



In [ ]:

Wie entwickelten sich die Positionen im zeitlichen Verlauf?

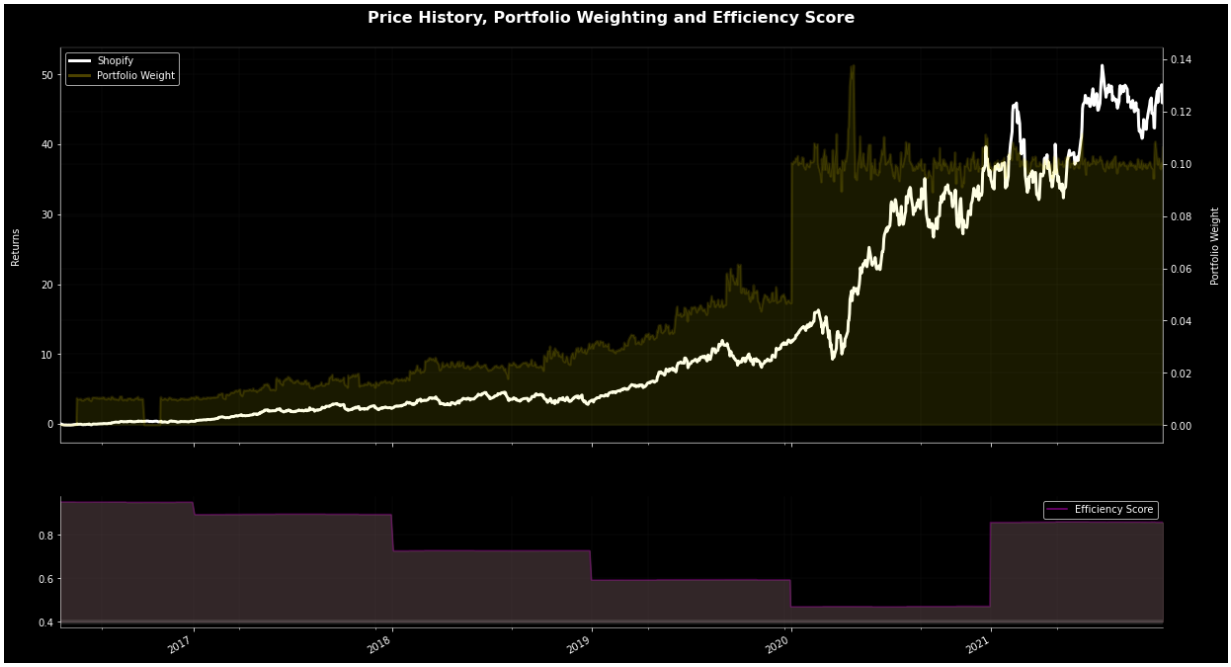
Wir wollen nun für einzelne Aktien genauer untersuchen wie die Strategie gearbeitet hat. Dazu betrachten wir neben dem Preisverlauf auch den Verlauf von Positionsgröße (Portfoliogewicht) und Efficiency Score.
Die Analyse wird hier jeweils für den größten Gewinner (Shopify) und den größten Verlierer (Twitter) dargestellt. Die Grafiken sind auf den relevanten Zeitraum zugeschnitten in dem auch tatsächlich Positionen gehalten wurden.

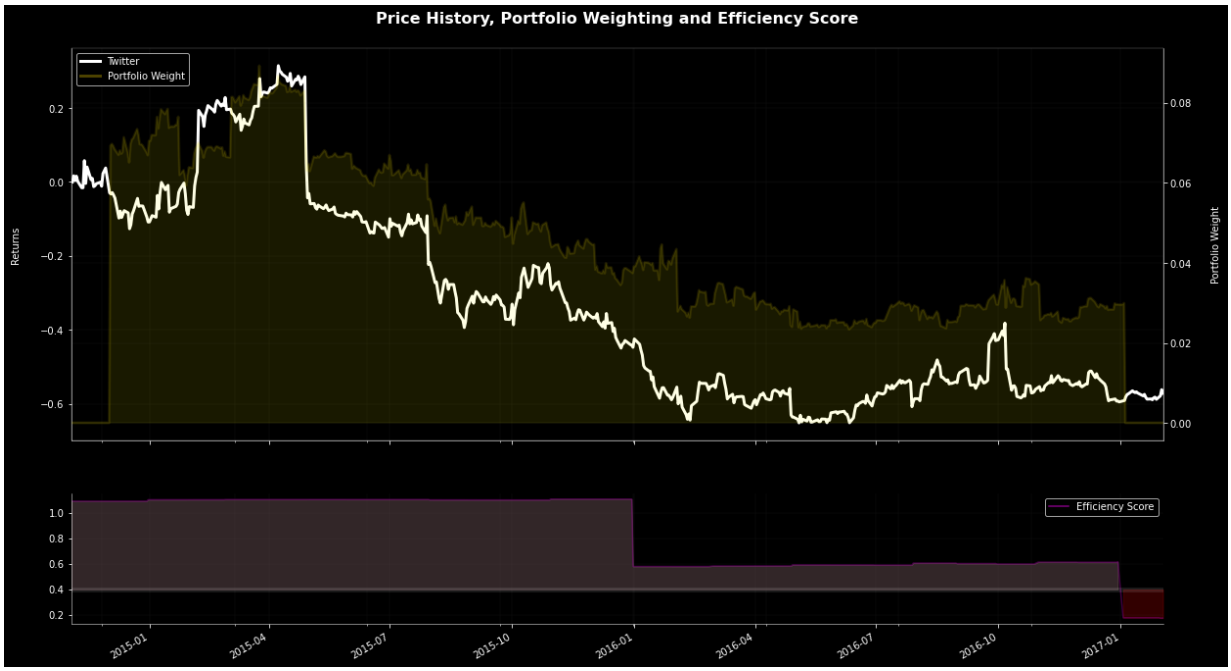
Bei Shopify fällt auf, dass der Efficiency Score im gesamten Zeitraum von 2016 bis Ende 2021 oberhalb der 40%-Marke lag. Daher war die Strategie auch fast durchgehend in Shopify investiert. Das Portfoliogewicht steigt zusammen mit der Aktie. Das liegt ganz einfach daran, dass die Gewichtung nach Marktkapitalisierung erfolgt. Der sprunghafte Anstieg der Gewichtung Anfang 2020 von ~5% auf ~10% erfolgte, weil andere Positionen aus dem Portfolio geflogen sind und das frei gewordene Kapital zu einem guten Teil in Shopify ging.

Zu Twitter (weiter unten) sei auf den Zeitraum verwiesen. Twitter hatte nur in der Zeit bis Anfang 2017 einen Efficiency Score > 40%, sodass die Strategie auch nur in dieser Zeit in Twitter investiert war. Als Timing-Instrument eignet sich der Efficiency Score damit schon mal nicht, denn man hat auf diese Weise fast den kompletten Absturz der Aktie mitgenommen, aber den danach folgenden Aufschwung zu neuen Hochs vollständig verpasst.

In [4]:

```
winner, loser = bta.GetBiggestWinner(), bta.GetBiggestLoser()
bta.PlotSummaryBySymbol(winner)
bta.PlotSummaryBySymbol(loser)
```


Price History, Portfolio Weighting and Efficiency Score
Shopify
Portfolio Weight
Returns
Portfolio Weight
Efficiency Score
2017
2018
2019
2020
2021


Price History, Portfolio Weighting and Efficiency Score
Twitter
Portfolio Weight
Returns
Portfolio Weight
Efficiency Score
2015-01
2015-04
2015-07
2015-10
2016-01
2016-04
2016-07
2016-10
2017-01

In [ ]:

In [ ]:

In [ ]:

In [ ]: