

An empirical study of electronics industry based on smart beta using Python:
Taiwan stock market as an example

Stan Chiang
National Chengchi University

Author Note

Stan Chiang, Department of Money and Banking, National Chengchi University
This research was supported in part by a grant from the English Taught Program.
Correspondence concerning this article should be addressed to Stan Chiang,
Department of Money and Banking, National Chengchi University, Taipei, Taiwan 11605.
Contact: 106302039@g.nccu.edu.tw

Abstract

This paper analyzed the effectiveness of selected factors and the assumption of beating the Taiwan stock market. Taiwan electronics industry was selected as a benchmark because of its large market share in the stock market. The industry is deeply affected by the business cycle. Companies that can maintain long-term stable profits at lower fixed costs are considered to have higher investment value. Smart beta is recently one of the most popular investment strategies. EBIT / net sales was selected as factors to measure how much a company makes a profit from a dollar of sales after paying variable costs and filter the abnormal performance of small companies. To answer this question, the author compared the performance of the constructed portfolio with the performance of Taiwan Electronics Index (TELI) using Python as **the main** analysis tool. His results **proved** that according to the historical data, the portfolio could literally **achieve** three times higher return and ten times higher sharpe ratio than TELI. His analysis was also accord with the effectiveness and the assumption.

Keywords: Smart Beta, Portfolio, Python, Taiwan Electronic Index, Factor

Introduction:

According to the statistic of Directorate General of Budget, Accounting and Statistics, 65% of GDP in Taiwan is composed of export and about 30% of export are electronics components. In addition, the electronics industry accounts for more than 50% of the total market value of listed companies. This shows that stocks of the electronics industry not only have a huge impact on the performance of the capital market, but also the focus of Taiwan's stock investment. That's why the author chose the electronics industry as his research object.

In fund investments, Smart Beta is currently a very popular investment strategy. It refers to an enhanced indexing strategy that seeks to exploit certain performance factors in an attempt to outperform a benchmark index. It also seeks to enhance returns, improve diversification, and reduce risk by investing in customized indexes or ETFs based on one or more predetermined "factors". Smart beta Strategies combine the benefits of passive investing and the advantages of active investing strategies (Fidelity learning center). Hence, world-known asset management firms like Vanguard and Blackrock, have issued various of ETF based on smart beta such as Vanguard Growth ETF, iShares Russell 1000 Growth ETF, Vanguard Dividend Appreciation ETF and iShares Edge MSCI Min Vol USA ETF.

Although some professionals claim that smart beta cannot beat the market and even underperforms than the benchmark index, the author believed that the existence of the effectiveness of smart beta and excess return based on the empirical study on the portfolio which is simulated using Python.

Initially, before constructing the portfolio, missing values, factors, duration, rebalance frequency and the number of constituent stocks per quarter should be determined which can have a significant impact on the return of the portfolio. Further, by comparing industry characteristics and the effect of factors among weighting methods, factor-weighting method is utilized to strengthen the representation of the portfolio. Finally, from the comparison of respective run chart, return (ER), Standard deviation (SD), Sharpe ratio (Sharpe), Max drawdown (MDD), and Odds ratio (odd) of the portfolio and the benchmark index, the effectiveness and the assumption of beating the market can be verified.

Literature Review:

Smart Beta is one of the investing strategies in funds investment. It is also called factor investing. Factor investing is based on the existence of factors that have earned a premium over long periods, reflect exposure to systematic risk, and are grounded in the academic literature (Bender, Briand, Melas, Subramanian, 2013). To understand factors, it is necessary to know the evolution history of modern finance theory. The earliest financial model that combined expected returns and risk is Capital Asset Pricing Model (CAPM) which became a foundation of modern financial theory in the 1960s (Lintner, 1965; Mossin, 1966; Sharpe, 1964 and Treynor, 1961) (Bender et al.). In CAPM, there are only systematic risk, expected return and risk-free rate. In 1976, Stephen Ross proposed another financial theory that described expected securities' return from a different perspective "Arbitrage pricing theory" (APT) (Ross, 1976). APT holds that the expected return of a financial asset can be modeled as a function of various macroeconomic factors or theoretical market indexes. Importantly, APT, unlike the CAPM, did not explicitly state what these factors should be (Bender et al.). Nowadays, there are three main categories of factors, macroeconomic, statistical and fundamental. The widely used factors are fundamental factors and the most popular factors today are Value, Low Size, Momentum, Low Volatility, Dividend Yield and Quality. Rosenberg and Marathe were among the first to describe the importance of these stock traits in explaining stock returns, leading to the creation of the multi-factor Barra risk models (Rosenberg and Marathe, 1976). Later, one of the best known efforts in this space came from Eugene Fama and Kenneth French in the early 1990s. Fama and French put forward a model explaining US equity market returns with three factors: the "market" (based on the traditional CAPM model), the size factor (large vs. small capitalization stocks) and the value factor (low vs. high book to market) (Fama and French, 1992, 1993). The "Fama-French" model, which today includes Carhart's momentum factor, has become a canon within the finance literature (Carhart, 2012). The above is partly from the abstract of Bender, Briand, Melas, Subramanian, 2013, Foundations of Factor investing, MSCI.

Although smart beta seems to have a rosy future, according to the research of Malkei (Malkei, 2014), he was convinced that smart beta portfolios didn't consistently outperform and when they do produced appealing results, they flunked the risk test.

The author believed the assumption that smart beta can outperform the market and also the effectiveness of factors exists because many researches around the world made the same conclusion as they had assumed and the effectiveness of factors has been verified by numerous well-known experts. The author refuted Malkei's conclusion because the author was convinced that according to the historical data, it can prove the same result as the assumption.

Discussion

Preparation of portfolio

After literature reviews of smart beta, next step was the construction of portfolio. Below is the basic information. There were still other elements that needed to be firstly determined and the raw data needed to be pre-processed. Furthermore, all the following content will be also displayed in Python.

Database: TEJ+

Market: Taiwan stock market

Industry: Electronics industry

Benchmark Index: Taiwan Electronics Index (TELI)

Listing: Taiwan Stock Exchange (TWSE) Listed (TSE) & Taipei Exchange (TPEX) Listed (OTC)

Duration: 2013010~20191231

Rebalance Frequency: Every quarter

Weighting Method: Factor-weighting

Factor: EBT Growth/Sales Growth

Programming Language: Python 3.7.4

Source Code Editor: Anaconda Jupyter Notebook in Visual Studio Code

Missing values. Missing values in the data can have a significant impact on the results.

The main reason that missing values occurred was because there was no data. There were a lot of methods to handle the data such as Listwise deletion, Pairwise deletion, Dummy variable adjustment and imputation (Acock, 2005). Here the samples per quarter are sufficient so that the author chose Listwise deletion to handle the data. The following is the implementation in Python.

```
df_F['EBT Growth'] = df_F['EBT Growth'].replace('-', np.NaN).astype('float')
df_F['Sales Growth'] = df_F['Sales Growth'].replace('-', np.NaN).astype('float')
```

```
df_F = df_F.dropna()
df_F
```


Factor. The selected factor is EBIT growth/Sales growth. This factor can make up for the deficiency in the classification of income statement and can capture the changes in fixed costs


of a company. For example, when EBIT growth is greater than sales growth and both are positive, It means that during this period, fixed costs have decreased, and fixed costs have always been a hidden concern in the electronics industry. If the firm can reduces fixed costs and retains profit growth at the same time, it is really a company worth investing in.


However, there was some problems when the author looked for the selected factor. First, TEJ+ database didn't have EBIT change rate. Therefore, it was replaced by EBT growth. The impact was that the interest cost would also be captured, and the factor couldn't purely measured operating efficiency of the company.

Second, growth rate could be positive or negative. Thus, there are four combinations, ++, +, +-, --. It got tricky when it came to determining which filtering conditions to set. The author decided to use the principle of subtraction, only adopting companies with positive EBT growth and Sales Growth.

Last, when the author looked at the data, he found that some companies with extreme values should not be included in the portfolio. After checking the quartiles and the median, the author decided to exclude the top 5% outliers which would significantly impact the result. The following were codes and the filtered results.

```
MI 
df_F = df_F[df_F['EBT Growth']>0]
df_F = df_F[df_F['Sales Growth']>0]
df_F = df_F.reset_index()
df_F
```

```
MI 
a = df_F.groupby('Date')['E/S'].quantile(0.95)
b = a.tolist()
b = b[::-1]
c = df_F['Date'].tolist()
date_list = []
for i in range(len(c)):
    if c[i] not in date_list:
        date_list.append(c[i])
    else:
        pass
True_list = []
for i in range(len(c)):
    count = 0
    for j in range(len(date_list)):
        if c[i] == date_list[j]:
            count = count + 1
            True_list.append(b[count])
            break
    else:
        count += 1
print(len(True_list))
```

```
MI 
count_del_list = []
ES_list = df_F['E/S'].tolist()
for i in range(len(True_list)):
    if ES_list[i] > True_list[i]:
        count_del_list.append(i)
    else:
        pass
print(count_del_list)
```

```

> MI 8+8
len(count_del_list)

> MI 8+8
df_g = df_F.copy()
print(df_g)

> MI 8+8
df_g = df_F.copy()
print(df_g)

> MI 8+8
df_g = df_g.drop(count_del_list)
df_g

```

	index	Code	Company	Date	Class	EBT Growth	Sales Growth	E/S
0	2	1569	濱川	201912	OTC	69.72	24.39	2.858549
1	5	1785	光洋科	201912	OTC	165.70	5.26	31.501901
2	12	2308	台達電	201912	TSE	27.86	13.13	2.121858
3	14	2313	華通	201912	TSE	42.06	10.52	3.998099
4	23	2329	華泰	201912	TSE	303.18	15.32	19.789817
...
10155	28616	8210	勤誠	201103	TSE	19.20	1.95	9.846154
10156	28619	8234	新漢	201103	OTC	213.80	39.00	5.482051
10157	28620	8240	華宏	201103	OTC	51.28	16.18	3.169345
10158	28629	8383	千耐	201103	OTC	193.42	74.08	2.610961
10159	28632	9912	偉聯	201103	TSE	126.49	15.68	8.066964

9637 rows x 8 columns

Rebalance frequency. TSE and OTC firms were required to hand in their financial statements (Statement of Comprehensive Income, Statement of Financial Position, Statement of Cash Flows and Statement of Changes in Equity) per quarter. The author could only acquire the selected factor from their quarterly financial statements so the author set rebalance frequency to “every quarter” and the constituent stocks of the portfolio would be rebalanced every quarter.

Duration. TSE and OTC firms were required to adopt IFRS accounting principles since 2013. Thus, the author choose the duration from 20130101 to 20191231 to collect precise figures. But EBT Growth and Sales Growth was the “percentage change” between last quarter and current quarter. The author needed to trace back two quarters before 2013Q1. The following were codes and the filtered results.

```

> MI 8+8
df_g = df_g[df_g['Date'] >= '201209']
df_g

```


	index	Code	Company	Date	Class	EBT	Growth	Sales	Growth	E/S
0	2	1569	濱川	201912	OTC	69.72		24.39	2.858549	
1	5	1785	光洋科	201912	OTC	165.70		5.26	31.501901	
2	12	2308	台達電	201912	TSE	27.86		13.13	2.121858	
3	14	2313	華通	201912	TSE	42.06		10.52	3.998099	
4	23	2329	華泰	201912	TSE	303.18		15.32	19.789817	
...
8981	24200	8114	振樺電	201209	TSE	7.99		5.93	1.347386	
8982	24203	8147	正凌	201209	OTC	2.80		2.51	1.115538	
8983	24205	8155	博智	201209	OTC	11.39		3.43	3.320700	
8984	24208	8176	智捷	201209	OTC	16.91		14.11	1.198441	
8985	24213	8213	志超	201209	TSE	252.70		38.65	6.538163	

8524 rows × 11 columns

Number of constituent stocks per quarter. There were almost 850 firms in the electronic industry. It was sufficient to choose 50 constituent stocks for the portfolio per quarter. Furthermore, practically, asset management and index companies mostly chose 50 constituent stocks as their ETF portfolios. After deciding the number of constituent stocks, the author ranked factors from highest to lowest per quarter. He also checked the number of constituent stocks in each quarter. The following were codes and the filtered results.

```

> MI 8-8
df_g['E/S_rank'] = df_g.groupby('Date')['E/S'].rank(ascending=False)
df_g

```

```

> MI 8+
df_N = df_g.dropna()
df_N = df_N[df_N['E/S_rank']<=50.0]
df_N

```

index	Code	Company	Date	Class	EBT	Growth	Sales	Growth	E/S	E/S_rank
1	5	1785	光洋科	201912	OTC	165.70		5.26	31.501901	9.0
4	23	2329	華泰	201912	TSE	303.18		15.32	19.789817	22.0
9	48	2367	耀華	201912	TSE	211.26		14.73	14.342159	34.0
13	60	2385	群光	201912	TSE	62.62		6.06	10.333333	39.0
16	64	2392	正崙	201912	TSE	170.07		11.27	15.090506	32.0
...
8968	24137	6282	康舒	201209	TSE	82.15		9.20	8.929348	40.0
8972	24165	8047	星雲	201209	OTC	87.54		2.47	35.441296	10.0
8973	24166	8048	德勝	201209	OTC	83.26		9.23	9.020585	39.0
8976	24179	8076	伍寶	201209	OTC	352.19		5.93	59.391231	5.0
8985	24213	8213	志超	201209	TSE	252.70		38.65	6.538163	44.0

1500 rows x 9 columns

```

> MI 8+
df_N.groupby('Date')['Code'].count()

```

```

> MI 8+
df_N.groupby('Date')['Code'].count()

```

```

Date
201209    50
201212    50
201303    50
201306    50
201309    50
201312    50
201403    50
201406    50
201409    50
201412    50
201503    50
201506    50
201509    50
201512    50
201603    50
201606    50
201609    50
201612    50
201703    50
201706    50
201709    50
201712    50
201803    50
201806    50
201809    50
201812    50
201903    50
201906    50
201909    50
201912    50
Name: Code, dtype: int64

```

Selection of weighting method

There were many weighting methods such as price-weighting, market capitalization-weighting, dividend-weighting, factor-weighting, equal-weighting (Nwogugu, 2019). Here the author only focused on factor-weighting, market capitalization-weighting and equal-weighting. After selecting weighting method, the author would use it to construct the portfolio by multiplying the return to weighting.

Equal-weighting. Equal-weighting method endowed constituent stocks in the portfolio with equal weighting. With this method, portfolio failed to show the characteristics of the factor.

Market capitalization-weighting. Market capitalization-weighting method was a common method to weight the portfolio. However, considering that there were companies with very large market value in the electronic industry like Taiwan semiconductor Manufacturing (TSMC) or Foxconn, if market capitalization-weighting method was used, they were included in the portfolio and it would result in too much weight in them.

Factor-weighting. Compared to other weighting methods, factor-weighting method could capture the characteristic of the factor and also wouldn't be influenced by the extreme market capitalization. The following were codes and the results after the weighting.

```

> M1 8+8
df_N['Weight'] = df_N['E/S']/(df_N.groupby('Date')['E/S'].transform('sum'))
df_N['Date'] = df_N['Date'].replace('-', '').astype('str')
df_N

```

	index	Code	Company	Date	Class	EBT	Growth	Sales	Growth	E/S	E/S_rank	Weight
1	5	1785	光洋科	201912	OTC	165.70	5.26	31.50	1901	9.0	0.032967	
4	23	2329	華泰	201912	TSE	303.18	15.32	19.78	9817	22.0	0.020710	
9	48	2367	耀華	201912	TSE	211.26	14.73	14.34	2159	34.0	0.015009	
13	60	2385	群光	201912	TSE	62.62	6.06	10.33	3333	39.0	0.010814	
16	64	2392	正崙	201912	TSE	170.07	11.27	15.09	0506	32.0	0.015792	
...	
8968	24137	6282	康舒	201209	TSE	82.15	9.20	8.92	9348	40.0	0.007755	
8972	24165	8047	星雲	201209	OTC	87.54	2.47	35.44	1296	10.0	0.030779	
8973	24166	8048	德勝	201209	OTC	83.26	9.23	9.02	0585	39.0	0.007834	
8976	24179	8076	伍豐	201209	OTC	352.19	5.93	59.39	1231	5.0	0.051579	
8985	24213	8213	志超	201209	TSE	252.70	38.65	6.53	8163	44.0	0.005678	

1500 rows x 10 columns

Portfolio. The author acquired daily unadjusted closing price of each companies from TEJ+ and calculated the daily return. Before the author merged the daily return with above weighted portfolio, it was necessary to think over the release date of financial statements. Q4 financial statements were required to release before March. Q1 financial statements were required to release before May. Q2 financial statements were required to release before August. Q3 financial statements were required to release before November.

January and February → last year Q3 financial statements

March and April → last year Q4 financial statements

May, June and July → this year Q1 financial statements

August, September and October → this year Q2 financial statements

November and December → this year Q3 financial statements

After considering the release date of financial statements, the author merged daily returns and weighted portfolio. Then, the author could obtain the daily return of the portfolio by cumulating the return per company. The following were codes and the daily return of the portfolio.

```
MI 8+8
df_P = pd.read_csv('price.txt',sep='\t')
df_P.columns = ['Code','Date_D','Price','Class']
df_P['Code'], df_P['Company'] = df_P['Code'].str.split(' ', 1).str
df_P = df_P[['Code', 'Company', 'Date_D', 'Class','Price']]
df_P['Price'] = df_P['Price'].replace(',','',regex=True).astype('float64')
df_P['Date_D'] = df_P['Date_D'].replace('/', '', regex=True).astype('int')
df_P
```

```
MI 8+8
df_P = df_P.sort_values(['Code','Date_D'])
df_P['Return_D'] = df_P.groupby('Company')['Price'].pct_change()
```

```
MI 8+8
c = df_P['Date_D'] >= 20130101
d = df_P['Date_D'] <= 20191231
df_P = df_P[ ( c & d ) ]
df_P.dropna()
df_P
```

	Code	Company	Date_D	Class	Price	Return_D
1322303	1333	恩得利	20130102	OTC	6.8667	0.014778
1321607	1333	恩得利	20130103	OTC	7.0000	0.019413
1320911	1333	恩得利	20130104	OTC	7.0167	0.002386
1320215	1333	恩得利	20130107	OTC	7.1000	0.011872
1319519	1333	恩得利	20130108	OTC	7.0333	-0.009394
...
3771	9912	億聯	20191225	TSE	8.0000	-0.011125
2932	9912	億聯	20191226	TSE	8.0000	0.000000
2093	9912	億聯	20191227	TSE	8.1000	0.012500
1254	9912	億聯	20191230	TSE	8.5500	0.055556
415	9912	億聯	20191231	TSE	8.5100	-0.004678

1322638 rows × 6 columns

```
MI 8+8
def make_key(date):
    month = str(date)[4:6]
    year = str(date)[0:4]

    if (month== '05' or month== '06' or month== '07' ):
        return year + '03'
    elif (month== '08' or month== '09' or month== '10' ):
        return year + '06'
    elif (month== '11' or month== '12' ):
        return year + '09'
    elif (month== '01' or month== '02' ):
        return str(int(year)-1) + '09'
    elif (month== '03' or month== '04' ):
        return str(int(year)-1) + '12'
```

```
df_P
df_P['Date_Key'] = df_P['Date_D'].apply(make_key)
df_P.Code = df_P.Code.astype('str')
df_P.Date_Key = df_P.Date_Key.astype('str')
df_P.Date_D = df_P.Date_D.astype('str')
df_N.Code = df_N.Code.astype('str')
df_N.Date = df_N.Date.replace("-", "", regex=True).astype('str')
```

```
df_final = pd.merge(left=df_P, right=df_N, left_on=['Code', 'Date_Key'], right_on=['Code', 'Date'], how='left')
df_final
```

	Code	Company_x	Date_D	Class_x	Price	Return_D	Date_Key	index	Company_y	Date	Class_y	EBT	Growth	Sales	Growth	E/S	E/S_rank	Weight
0	1333	恩得利	20130102	OTC	6.8667	0.014778	201209	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	1333	恩得利	20130103	OTC	7.0000	0.019413	201209	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	1333	恩得利	20130104	OTC	7.0167	0.002386	201209	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	1333	恩得利	20130107	OTC	7.1000	0.011872	201209	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	1333	恩得利	20130108	OTC	7.0333	-0.009394	201209	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
...
1322633	9912	偉聯	20191225	TSE	8.0000	-0.011125	201909	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1322634	9912	偉聯	20191226	TSE	8.0000	0.000000	201909	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1322635	9912	偉聯	20191227	TSE	8.1000	0.012500	201909	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1322636	9912	偉聯	20191230	TSE	8.5500	0.055556	201909	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1322637	9912	偉聯	20191231	TSE	8.5100	-0.004678	201909	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

1322638 rows x 16 columns

```
df_final['WeightxReturn'] = df_final['Return_D'] * df_final['Weight']
df_final
```

```
Port_return = df_final.groupby('Date_D')['WeightxReturn'].sum()
Port_return.index = pd.to_datetime(Port_return.index.astype('str'))

Port_return
```

Date_D	WeightxReturn
2013-01-02	0.014531
2013-01-03	0.003315
2013-01-04	-0.001012
2013-01-07	0.001895
2013-01-08	-0.004644
...	...
2019-12-25	0.006901
2019-12-26	0.000982
2019-12-27	0.003458
2019-12-30	-0.003225
2019-12-31	0.001847

Name: WeightxReturn, Length: 1717, dtype: float64

Comparison of portfolio and benchmark index

Benchmark index. The author acquired daily closing price of TELI from TEJ+ and calculated the daily return. Next, he calculated the cumulated return of the portfolio and benchmark index respectively, merging them into the same table.

```
df_index = pd.read_csv('index data.txt', encoding='CP950', sep='\t')
df_index.columns = ['Code', 'Name', 'Date', 'Close']
df_index.index = pd.to_datetime(df_index['Date'].astype('str'))
df_index['Return_M'] = df_index['Close'].pct_change()

{}
```

```
df_index = df_index[df_index.Date >= 20130101]
df_index = df_index[df_index.Date <= 20191231]
df_index['Return_M_Cum'] = (df_index['Return_M'] + 1).cumprod()
```

```

> MU 8+8
df_index['Return_P'] = Port_return
df_index['Return_P_Cum'] = ( df_index['Return_P']+1).cumprod()
df_index

```

	Code	Name	Date	Close	Return_M	Return_M_Cum	Return_P	Return_P_Cum
			Date					
2013-01-02	M2300	電子類指數	20130102	292.53	0.011410	1.011410	0.014531	1.014531
2013-01-03	M2300	電子類指數	20130103	294.10	0.005367	1.016838	0.003315	1.017895
2013-01-04	M2300	電子類指數	20130104	292.47	-0.005542	1.011202	-0.001012	1.016865
2013-01-07	M2300	電子類指數	20130107	289.38	-0.010565	1.000519	0.001895	1.018792
2013-01-08	M2300	電子類指數	20130108	288.13	-0.004320	0.996197	-0.004644	1.014061
...
2019-12-25	M2300	電子類指數	20191225	528.39	0.004620	1.826885	0.006901	4.896238
2019-12-26	M2300	電子類指數	20191226	528.20	-0.000360	1.826228	0.000982	4.901045
2019-12-27	M2300	電子類指數	20191227	533.39	0.009826	1.844172	0.003458	4.917992
2019-12-30	M2300	電子類指數	20191230	530.95	-0.004575	1.835736	-0.003225	4.902129
2019-12-31	M2300	電子類指數	20191231	527.82	-0.005895	1.824914	0.001847	4.911183

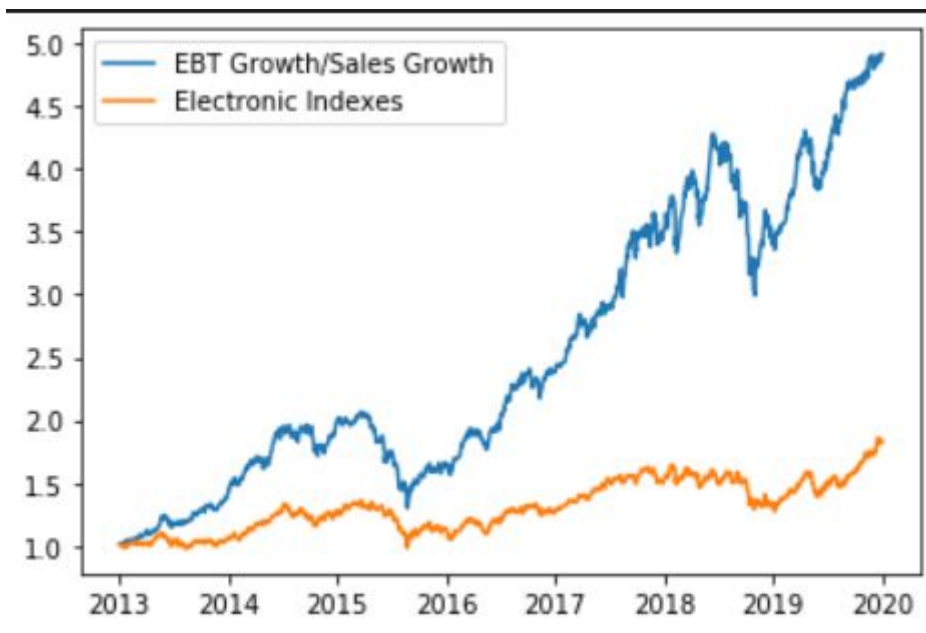
1717 rows x 8 columns

Return. The duration was from 2013 to 2020. From 2013 to 2015Q3, both followed the similar trend but the portfolio enjoyed near two times higher return. From 2015Q3 to 2020, the portfolio diverged with the benchmark index. The portfolio continued to breaking upward and reached near 500% while the benchmark index retained at the lower level which was close to 180%. In 2020, the cumulated return of the portfolio was almost three times higher than the benchmark index.

```

> MU 8+8
plt.figure(figsize=(15,7))
plt.plot(df_index['Return_P_Cum'],label='EBT Growth/Sales Growth')
plt.plot(df_index['Return_M_Cum'],label='Electronic Indexes')
plt.legend()

```



Performance. The author chose 5 ratios to compare the performance between the portfolio and the benchmark index. First, geometric mean (ER) was a better method to measure the

return than arithmetic mean. Even though the author adopted the ER, ER of the portfolio was still higher than ER of the benchmark. Second, the standard deviation (std) of the portfolio was little higher than the benchmark. Given the higher return, 0.02 was an acceptable and tolerable risk. The following is Sharpe ratio. Sharpe ratio (Sharpe) is the excess return divided by total risk. Given the same risk, the excess return of the portfolio was still greater than the benchmark. Next, maxdrawdown (mdd) describes the maximum loss that investors may face. Although mdd of the portfolio was 0.09 higher than the benchmark, it was worthwhile to tolerate the risk because of the greater return. Last, odds ratio (odd) is the win rate of the portfolio. From the portfolio, its odds was also greater than the benchmark. In brief, the portfolio was greater than the benchmark in almost every item.

```

In [8]:
Geo_a_return_M = (df_index['Return_M_Cum'] ['2019-12-31']) ** (250 / len(df_index)) - 1
Geo_a_return_P = (df_index['Return_P_Cum'] ['2019-12-31']) ** (250 / len(df_index)) - 1

STD_return_M = df_index['Return_M'].std() * np.sqrt(250)
STD_return_P = df_index['Return_P'].std() * np.sqrt(250)

Sharpe_M = Geo_a_return_M / STD_return_M
Sharpe_P = Geo_a_return_P / STD_return_P

#計算MDD
D = df_index['Return_M_Cum'].cummax() - df_index['Return_M_Cum']
MDD = D.max() * 1
d = D / (D + df_index['Return_M_Cum'])
mdd_M = d.max()

D = df_index['Return_P_Cum'].cummax() - df_index['Return_P_Cum']
MDD = D.max() * 1
d = D / (D + df_index['Return_P_Cum'])
mdd_P = d.max()

#投組勝率
Odd_M = sum(df_index['Return_M'] > 0) / len(df_index)
Odd_P = sum(df_index['Return_P'] > 0) / len(df_index)
#.cummax 每一列最大值

```

```

In [8]:
df_performance = pd.DataFrame([
    [Geo_a_return_M, STD_return_M, Sharpe_M, mdd_M, Odd_M],
    [Geo_a_return_P, STD_return_P, Sharpe_P, mdd_P, Odd_P]
])

In [8]:
df_performance.columns = ['ER', 'STD', 'Sharpe', 'Mdd', 'Odd']
df_performance.index = ['Benchmark', 'Portfolio']

```

比較兩者績效

```

In [8]:
df_performance.round(2)

```

	ER	STD	Sharpe	Mdd	Odd
Benchmark	0.09	0.15	0.61	0.28	0.53
Portfolio	0.26	0.17	1.58	0.37	0.60

Conclusion

In this paper, the author wanted to verify the doubt that smart beta couldn't beat the market and the doubt of the effectiveness of smart beta. He chose Taiwan stock market as the trading market and the electronics industry was his research object. In addition, Taiwan electronic index was its benchmark index. He then selected the EBT Growth/Sales Growth as a factor to form the portfolio and rebalanced every quarter with Python. Next, he compared the performance between the portfolio and the benchmark index. From the results, it can be proved that the effectiveness of smart beta literally existed. Moreover, smart beta strategies could indeed beat the market and enjoyed greater excess return.

References

- Acock, A. C. (2005). Working with missing values. *Journal of Marriage and Family*. doi:10.1111/j.1741-3737.2005.00191.x
- Bender, J., Briand, R., Melas, D., Subramanian, R. A (2013). *Foundations of factor investing*. MSCI.
- Carhart, M. M. (2012). On persistence in mutual fund performance. *The Journal of Finance*. doi: 10.1111/j.1540-6261.1997.tb03808.x
- Cox, J. C., Ross, S. A. (1976). The valuation of options for alternative stochastic process. *Journal of Financial Economics*.
- Fama, E. F., French, K. R. (1992). The cross-sectional of expected stock returns. *The journal of Finance*. doi:10.1111/j.1540-6261.1992.tb04398.x
- Fama, E. F., French, K. R. (1993). Common risk factors in the returns on stocks and bonds. *Journal of Financial Economics*.
- Fidelity learning Center. All about alpha, beta, and smart beta. Retrieved from <https://www.fidelity.com/learning-center/investment-products/etf/smart-beta>
- Lintner, J. (1965). Security prices, risk, and maximal gains from diversification. *The Journal of Finance*. doi:10.2307/2977249
- Malkiel, B. G. (2014). Is smart beta really smart? *The Journal of Portfolio Management*. doi:10.3905/jpm.2014.40.5.127
- Mossin, J. (1966). Equilibrium in a capital asset market. *Econometrica*. doi:10.2307/1910098
- Nwogugu, M. I. C. (2019). *Indices, index funds and etfs: exploring HCI, nonlinear risk and homomorphisms*. London, United Kingdom: Springer.
- Rosenberg, B., Marathe, V. (1976). Common factors in security returns: microeconomic determinants and macroeconomic correlates. *Research program in Finance Working Papers 44, University of California at Berkeley*
- Sharpe, W. F. (1964). Capital asset prices: a theory of market equilibrium under conditions of risk. *The Journal of Finance*. doi:10.1111/j.1540-6261.1964.tb02865.x
- Treynor, J. L. (1962). Toward a Theory of Market Value of Risky Assets. Unpublished manuscript