

Small Cap Value and Momentum

MGMT 638: Data-Driven Investments: Equity

Kerry Back, Rice University



Small Cap Value and Growth

- small cap \approx Russell 2000
- value usually measured by PB or PE
- some academic work (Fama-French) found PB is a better predictor of returns
- low PB = value, high PB = growth
- academics usually use BP instead of PB and call it book-to-market
- high BP = value, low BP = growth
- small-cap growth has historically had very poor returns



Value and Momentum Portfolios I

- get marketcap data in addition to prices
- calculate momentum
- keep stocks between 1,001 and 3,000 in market cap
- create 5x5 sort on value and momentum
- compute equally weighted portfolio returns



Value and Momentum Portfolios II

- rank each stock between 1,001 and 3,000 on value
 - low rank = best (low pb)
- rank each stock also on momentum
 - low rank = best (high momentum)
- add ranks to get a single combined rank
 - low combined rank = best
- go long best 50 and short worst 50



Value and Momentum Portfolios III

- For long only portfolio, choose best stocks in each sector and match sector weights to benchmark (e.g., Russell 2000).
- For long-short portfolio, match shorts and longs in each sector to get market-neutral and sector-neutral portfolio.



Value and Momentum Portfolios IV

- Use machine learning to find the optimal way to combine value and momentum
- And add other predictors (ROE, investment rate, short-term reversal, ...)



Data and Procedure

- Get sectors from tickers table
- Get marketcap and pb from weekly table
- Get closeadj and closeunadj from sep_weekly as before
- Calculate momentum as before
- Filter to 1,001-3,000 on marketcap each week
- Form portfolios



Create connection



Get data



Calculate momentum



Merge marketcap and pb



Save this week's data



```
In [7]: today = df[df.date==df.date.max()]
today.head(3)
```

```
Out[7]:
```

	ticker	date	ret	mom	closeunadj	marketcap	pb	sector
668	A	2023-10-27	-0.059141	-0.188863	102.77	30069.2	5.4	Healthcare
981	AA	2023-10-27	-0.020825	-0.256682	23.51	4195.9	0.9	Basic Materials
1644	AADI	2023-10-27	0.039120	-0.626255	4.25	104.2	0.8	Healthcare



Shift predictors and shift filtering variables to backtest



```
In [8]: df = df.set_index(["ticker", "date"])
variables = ["mom", "pb", "marketcap", "closeunadj"]
df[variables] = df.groupby("ticker", group_keys=False)[variables].shift()
df = df.dropna()
df.head(3)
```

```
Out[8]:
```

		ret	mom	closeunadj	marketcap	pb	sector
ticker	date						
A	2011-01-14	0.008130	0.199287	41.88	14557.7	4.5	Healthcare
	2011-01-21	0.050456	0.270914	42.22	14675.8	4.5	Healthcare
	2011-01-28	-0.075973	0.337839	44.35	15416.2	4.8	Healthcare

Filter out penny stocks and filter to small caps




```
In [9]: df = df[df.closeunadj>5]
df["rnk"] = df.groupby("date").marketcap.rank(
    ascending=False,
    method="first"
)
df = df[(df.rnk>1000) & (df.rnk<=3000)]
df.reset_index().groupby("date").ticker.count()
```

```
Out[9]: date
2011-01-14    2000
2011-01-21    2000
2011-01-28    2000
2011-02-04    2000
2011-02-11    2000
...
2023-09-29    1865
2023-10-06    1853
2023-10-13    1837
2023-10-20    1829
2023-10-27    1802
Name: ticker, Length: 668, dtype: int64
```



Value and Momentum Portfolios I



```
In [10]: df["value_group"] = df.groupby("date", group_keys=False).pb.apply(
        lambda x: pd.qcut(x, 5, labels=range(1, 6))
    )
df["mom_group"] = df.groupby("date", group_keys=False).mom.apply(
    lambda x: pd.qcut(x, 5, labels=range(1, 6))
)
rets = df.groupby(["date", "value_group", "mom_group"]).ret.mean()
rets = rets.unstack().unstack()
rets.head(3)
```

Out[10]:

mom_group		1					
value_group		1	2	3	4	5	1
date							
2011-01-14		-0.004985	-0.014070	-0.008452	-0.006321	-0.009538	-0.006124
2011-01-21		0.018622	0.018095	0.020878	0.013126	0.003709	0.013191
2011-01-28		-0.026927	-0.021369	-0.030210	-0.027047	-0.030028	-0.010046

3 rows × 25 columns



```
In [11]: (52*rets.mean()).unstack().round(3)
```

```
Out[11]:
```

	value_group	1	2	3	4	5
	mom_group					
	1	0.040	0.038	0.060	0.052	-0.003
	2	0.114	0.087	0.076	0.079	0.066
	3	0.129	0.092	0.094	0.102	0.097
	4	0.143	0.095	0.094	0.117	0.079
	5	0.178	0.125	0.112	0.104	0.138

How many stocks are in the groups?



```
In [12]: counts = df.groupby(["date", "value_group", "mom_group"]).ret.count()
counts = counts.unstack().unstack()
counts.tail(3)
```

Out[12]:

mom_group		1					2					...	4				
value_group		1	2	3	4	5	1	2	3	4	5	...	1	2	3	4	5
date																	
2023-10-13		131	74	61	57	45	103	94	57	53	60	...	50	75	87	79	76
2023-10-20		138	75	57	50	46	108	94	59	47	58	...	58	71	80	66	91
2023-10-27		144	63	54	52	48	107	94	52	57	50	...	62	80	55	79	84

3 rows × 25 columns

Value and Momentum Portfolios II



- Rank stocks on momentum each week: 1=best, 2=next best, etc. (best=high momentum)
- Rank stocks on pb each week: 1=best, 2=next best, etc. (best=low pb)
- Add momentum and pb ranks: lowest combined ranks are best stocks
- Test A: sort into deciles on combined ranks and compute equally weighted returns
- Test B: go long best 50 stocks and short worst 50 stocks and compute returns




```
In [13]: df["mom_rnk"] = df.groupby("date", group_keys=False).mom.rank(  
        ascending=False,  
        method="first"  
    )  
df["pb_rnk"] = df.groupby("date", group_keys=False).pb.rank(  
    ascending=True,  
    method="first"  
)  
df["combined_rnk"] = df.mom_rnk + df.pb_rnk
```

Test A: Deciles

```
In [14]: df["decile"] = df.groupby("date", group_keys=False).combined_rnk.apply(
        lambda x: pd.qcut(x, 10, labels=range(1, 11))
    )
    rets = df.groupby(["date", "decile"]).ret.mean()
    rets = rets.unstack()
    52*rets.mean()
```

```
Out[14]: decile
1      0.140782
2      0.111245
3      0.110181
4      0.106055
5      0.096709
6      0.103076
7      0.092262
8      0.056943
9      0.074983
10     0.034706
dtype: float64
```



Test B: Top 50 and Bottom 50

- rank at each date on combined_rnk
- put best 50 in long portfolio at each date
- put worst 50 in short portfolio at each date
- compute equally weighted returns in each portfolio
- calculate long minus short return



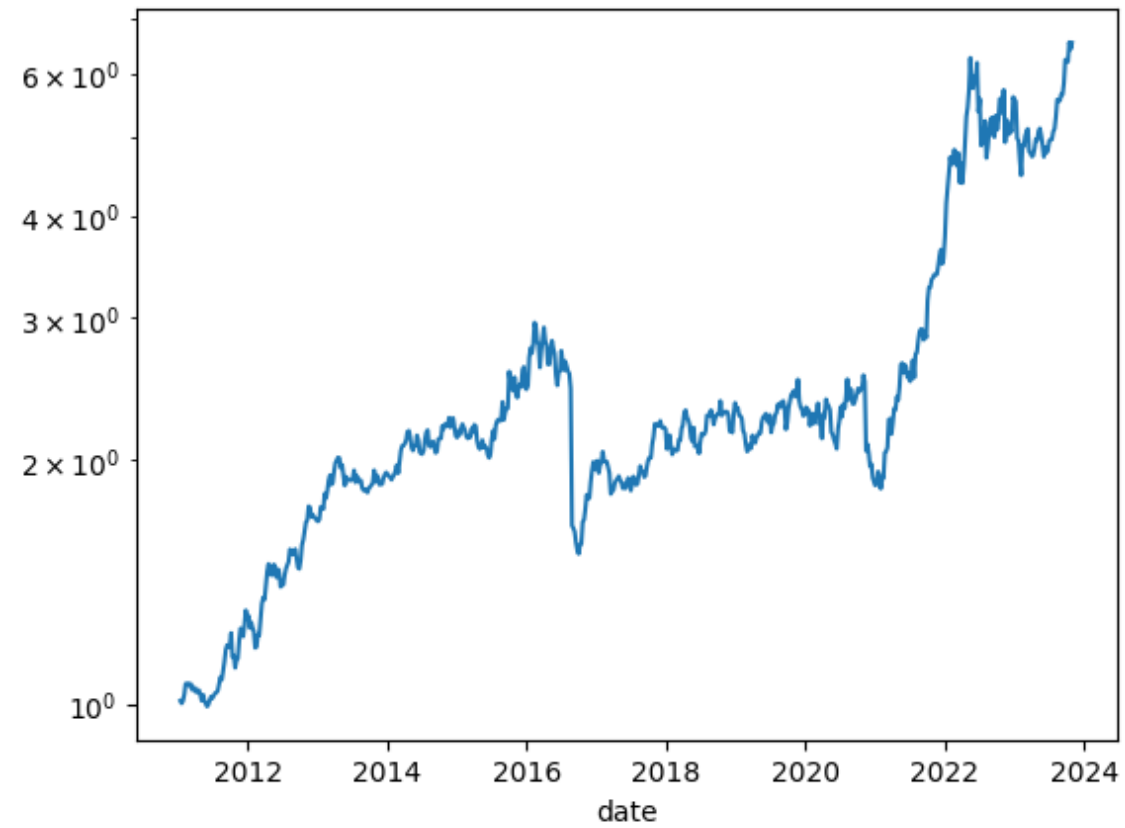
```
In [16]: print(f"annualized mean long return is {52*long_rets.mean():.2%}")  
         print(f"annualized mean short return is {52*short_rets.mean():.2%}")
```

```
annualized mean long return is 17.82%  
annualized mean short return is 0.23%
```



```
In [17]: (1+long_rets-short_rets).cumprod().plot(logy=True)
```

```
Out[17]: <Axes: xlabel='date'>
```



What are the top 50 and bottom 50 today?

- Apply penny stock and size filters to today dataframe
- Rank on momentum (low rank = high momentum = best)
- Rank on value (low rank = low pb = best)
- Add ranks
- Find best 50 and worst 50 stocks today



In [20]: long

Out[20]:

	ticker	sector	mom_rnk	pb_rnk	combined_rnk	closeunadj	
	772871	EHTH	Financial Services	26	32	58	7.860
	427948	CBUS	Healthcare	30	69	99	10.270
	2367118	TRML	Healthcare	78	53	131	14.000
	1429651	LSEA	Real Estate	97	39	136	7.240
	2197648	SPHR	Communication Services	100	51	151	33.470
	387833	BZH	Consumer Cyclical	48	221	269	23.430
	2449388	USAP	Basic Materials	102	197	299	14.020
	1761051	OPRT	Financial Services	272	44	316	5.510
	1597522	MUX	Basic Materials	99	267	366	7.090
	1488461	MDV	Real Estate	125	260	385	15.210
	1139786	HOV	Consumer Cyclical	29	362	391	66.360
	1593141	MTW	Industrials	128	266	394	12.320
	959535	FRD	Basic Materials	236	147	423	9.710



In [21]: short

Out[21]:

	ticker	sector	mom_rnk	pb_rnk	combined_rnk	closeunadj	
	2425203	UG	Consumer Defensive	1664	1382	3046	6.180
	1121494	HLIT	Technology	1596	1456	3052	9.890
	1570573	MRVI	Healthcare	1623	1431	3054	6.220
	124978	AMLX	Healthcare	1701	1354	3055	15.740
	1869626	PLMR	Financial Services	1625	1433	3058	49.360
	229881	AUID	Technology	1460	1599	3059	6.000
	291701	BE	Industrials	1409	1654	3063	9.780
	1903132	PRCT	Healthcare	1393	1678	3071	26.090
	260770	AYX	Technology	1304	1767	3071	31.460
	244448	AVXL	Healthcare	1685	1399	3084	5.200
	1447625	LYFT	Technology	1387	1701	3088	9.260
	1217100	IMXI	Technology	1566	1524	3090	16.200
	1253736	IRTC	Healthcare	1378	1716	3094	78.190
	2107199	SEMR	Technology	1463	1635	3098	8.050



2442252 UDAV Consumer 1554 1554 3122 9.770

Sector weights



```
In [22]: long.groupby("sector").ticker.count()
```

```
Out[22]: sector
Basic Materials      3
Communication Services  2
Consumer Cyclical   11
Energy              1
Financial Services  19
Healthcare          4
Industrials         5
Real Estate         5
Name: ticker, dtype: int64
```



```
In [23]: short.groupby("sector").ticker.count()
```

```
Out[23]: sector
Basic Materials      1
Communication Services  1
Consumer Cyclical    3
Consumer Defensive   2
Energy               2
Financial Services   3
Healthcare           19
Industrials          3
Technology           15
Utilities            1
Name: ticker, dtype: int64
```



Value and Momentum Portfolios III

- Rank on combined rank separately in each sector
- Do that by grouping by date and sector instead of just date
- Go long best 5 and short worst 5 in each sector to get sector neutrality
- Compute equally weighted returns for long and short portfolios
- Compute long minus short return



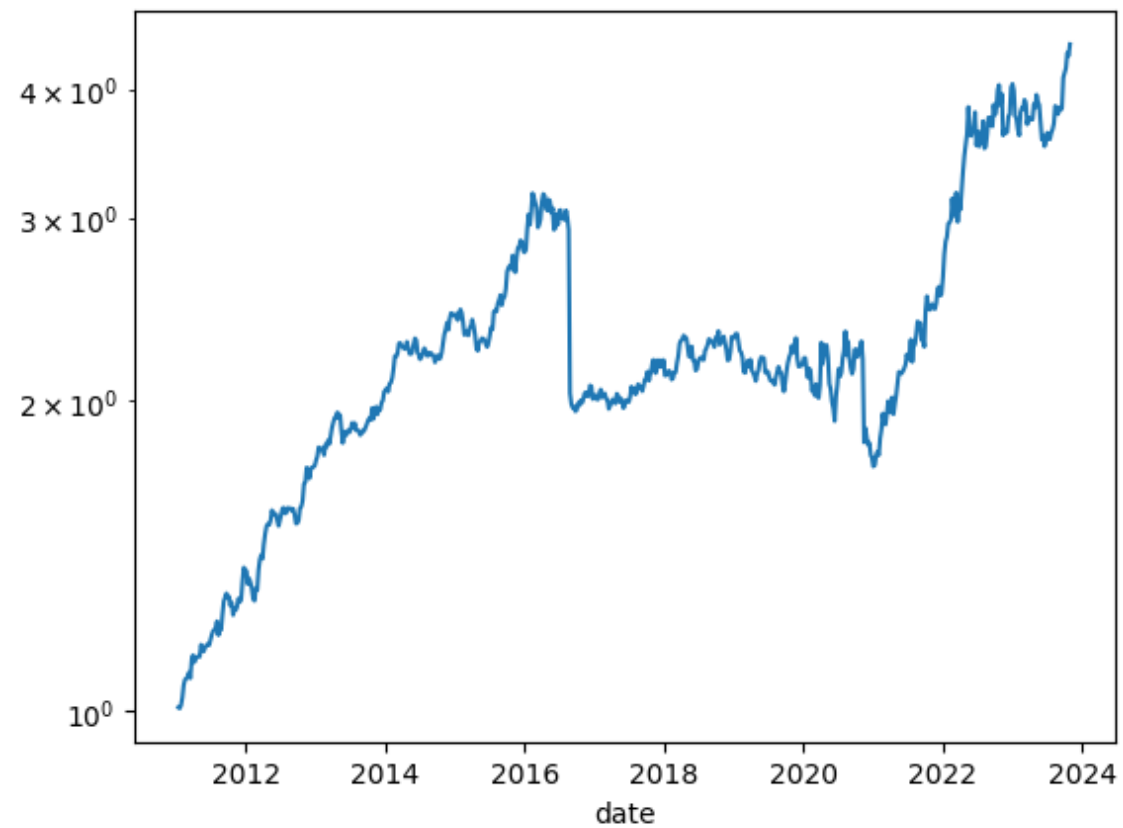
```
In [25]: print(f"annualized mean long return is {52*long_rets.mean():.2%}")  
         print(f"annualized mean short return is {52*short_rets.mean():.2%}")
```

```
annualized mean long return is 15.07%  
annualized mean short return is 1.82%
```



```
In [26]: (1+long_rets-short_rets).cumprod().plot(logy=True)
```

```
Out[26]: <Axes: xlabel='date'>
```



Best and worst stocks today in sector-neutral strategy

- Just group by sector when ranking
- Choose top 5 and bottom 5 in each sector




```
In [28]: long_neutral
```

Out[28]:

	ticker	sector	mom_rnk	pb_rnk	combined_rnk	closeunadj	
	2449388	USAP	Basic Materials	102	197	299	14.020
	1597522	MUX	Basic Materials	99	267	366	7.090
	950535	FRD	Basic Materials	276	147	423	9.710
	2638060	ZEUS	Basic Materials	47	636	683	49.430
	986128	GATO	Basic Materials	182	579	761	5.010
	2197648	SPHR	Communication Services	100	51	151	33.470
	2300744	TDS	Communication Services	504	24	528	17.800
	2455127	USM	Communication Services	274	430	704	41.390
	1177442	IAC	Communication Services	704	159	863	41.840
	2116089	SGA	Communication Services	751	195	946	19.150
	387833	BZH	Consumer Cyclical	48	221	269	23.430
	1122726	WOM	Consumer	22	222	224	22.222

```
In [29]: short_neutral
```

Out[29]:

	ticker	sector	mom_rnk	pb_rnk	combined_rnk	closeunadj	
	1433593	LTHM	Basic Materials	1633	986	2619	15.110
	1573351	MSB	Basic Materials	950	1725	2675	20.170
	2165284	SMID	Basic Materials	1242	1447	2689	19.366
	1555752	MP	Basic Materials	1572	1207	2779	16.500
	2379344	TSE	Basic Materials	1735	1708	3443	5.990
	1195703	IDT	Communication Services	1159	1501	2660	27.790
	1755960	OOMA	Communication Services	1203	1535	2738	10.740
	934187	FNGR	Communication Services	1019	1749	2768	5.650
	1042555	GOGO	Communication Services	1113	1776	2889	10.650
	2388282	TTGT	Communication Services	1732	1497	3229	25.260
	425453	CBRL	Consumer Cyclical	1611	1418	3029	64.620
	1133337	WED	Consumer	1433	1734	2844	22.530

```
In [30]: long_neutral.groupby("sector").ticker.count()
```

```
Out[30]: sector
Basic Materials      5
Communication Services  5
Consumer Cyclical    5
Consumer Defensive   5
Energy               5
Financial Services   5
Healthcare           5
Industrials          5
Real Estate          5
Technology           5
Utilities            5
Name: ticker, dtype: int64
```



```
In [31]: short_neutral.groupby("sector").ticker.count()
```

```
Out[31]: sector
Basic Materials      5
Communication Services  5
Consumer Cyclical    5
Consumer Defensive   5
Energy               5
Financial Services   5
Healthcare           5
Industrials          5
Real Estate          5
Technology           5
Utilities            5
Name: ticker, dtype: int64
```



How many shares to buy/sell?

- Can do this either for long and short or long_neutral and short_neutral
- \$1,000,000 to invest long and short
- Divide by number of stocks to get \$ per stock
- Divide by price to get shares per stock



Long side



```
In [32]: long_neutral["shares"] = (1000000/long_neutral.shape[0])/long_neutral.closeunadj
long_neutral["shares"] = long_neutral.shares.round(0).astype(int)
long_neutral
```

Out[32]:

	ticker	sector	mom_rnk	pb_rnk	combined_rnk	closeunadj	
	2449388	USAP	Basic Materials	102	197	299	14.020
	1597522	MUX	Basic Materials	99	267	366	7.090
	950535	FRD	Basic Materials	276	147	423	9.710
	2638060	ZEUS	Basic Materials	47	636	683	49.430
	986128	GATO	Basic Materials	182	579	761	5.010
	2197648	SPHR	Communication Services	100	51	151	33.470
	2300744	TDS	Communication Services	504	24	528	17.800
	2455127	USM	Communication Services	274	430	704	41.390
	1177442	IAC	Communication Services	704	159	863	41.840
	2116089	SGA	Communication Services	751	195	946	19.150
	387833	BZH	Consumer Goods	48	221	269	23.430



Short side



In [33]:

```
short_neutral["shares"] = (1000000/short_neutral.shape[0])/short_neutral.close
short_neutral["shares"] = short_neutral.shares.round(0).astype(int)
short_neutral
```

Out[33]:

	ticker	sector	mom_rnk	pb_rnk	combined_rnk	closeunadj
1433593	LTHM	Basic Materials	1633	986	2619	15.110
1573351	MSB	Basic Materials	950	1725	2675	20.170
2165284	SMID	Basic Materials	1242	1447	2689	19.366
1555752	MP	Basic Materials	1572	1207	2779	16.500
2379344	TSE	Basic Materials	1735	1708	3443	5.990
1195703	IDT	Communication Services	1159	1501	2660	27.790
1755960	OOMA	Communication Services	1203	1535	2738	10.740
934187	FNGR	Communication Services	1019	1749	2768	5.650
1042555	GOGO	Communication Services	1113	1776	2889	10.650
2388282	TTGT	Communication Services	1732	1497	3229	25.260
425453	CBRL	Consumer Goods	1611	1418	3029	64.620



```
In [35]: with pd.ExcelWriter("portfolios 2023-11-01.xlsx") as writer:
        long.to_excel(writer, "long", index=False)
        short.to_excel(writer, "short", index=False)
        long_neutral.to_excel(writer, "long neutral", index=False)
        short_neutral.to_excel(writer, "short neutral", index=False)
        today.to_excel(writer, "today", index=False)
```

