

## Features

- pb
- mom
- marketcap
- volume
- volatility
- roe
- accr
- agr
- interactions of all with market volatility



## Procedure

- Follow notebook 05a-fundamentals to create all features except market vol
- Compute market volatility
  - Get daily market returns from French's data library
  - Compute trailing 21 day standard deviation
  - Downsample to weekly and merge with other data
- Multiply other features by market volatility
- Save to csv file



Follow 05a-fundamentals



In [16]:

```
import pandas as pd

from sqlalchemy import create_engine
import pymssql
server = 'fs.rice.edu'
database = 'stocks'
username = 'stocks'
password = '6LAZH1'
string = "mssql+pymssql://"+ username + ":" + password + "@" + server + "/"
conn = create_engine(string).connect()
```

Exception during reset or similar

Traceback (most recent call last):

File "c:\Users\kerry\AppData\Local\Programs\Python\Python310\lib\site-packages\sqlalchemy\pool\base.py", line 753, in \_finalize\_fairy

fairy.\_reset(pool)

File "c:\Users\kerry\AppData\Local\Programs\Python\Python310\lib\site-packages\sqlalchemy\pool\base.py", line 1004, in \_reset

pool.\_dialect.do\_rollback(self)

File "c:\Users\kerry\AppData\Local\Programs\Python\Python310\lib\site-packages\sqlalchemy\dialects\mssql\base.py", line 2792, in do\_rollback

super(MSDialect, self).do\_rollback(dbapi\_connection)

File "c:\Users\kerry\AppData\Local\Programs\Python\Python310\lib\site-packages\sqlalchemy\engine\default.py", line 683, in do\_rollback

dbapi\_connection.rollback()

File "src\pymssql\\_pymssql.pyx", line 316, in pymssql.\_pymssql.Connection.rollback

File "src\pymssql\\_pymssql.pyx", line 300, in pymssql.\_pymssql.Conn



## Calculate financial ratios and growth rates

Data from SF1



```
In [17]: sf1 = pd.read_sql(
        """
        select ticker, datekey, lastupdated, netinc, ncfo, equity, assets
        from sf1
        where dimension='ARQ' and datekey>='2009-01-01' and equity>0 and assets>0
        order by ticker, datekey
        """,
        conn,
        parse_dates=["datekey"]
    )
sf1 = sf1.groupby(["ticker", "datekey", "lastupdated"]).last()
sf1 = sf1.droplevel("lastupdated")
sf1 = sf1.reset_index()
```

```
In [18]: for col in ["netinc", "ncfo"]:
          sf1[col] = sf1.groupby("ticker", group_keys=False)[col].apply(
              lambda x: x.rolling(4).sum()
          )
        for col in ["equity", "assets"]:
          sf1[col] = sf1.groupby("ticker", group_keys=False)[col].apply(
              lambda x: x.rolling(4).mean()
          )
        sf1["roe"] = sf1.netinc / sf1.equity
        sf1["accruals"] = (sf1.netinc - sf1.ncfo) / sf1.equity
        sf1["agr"] = sf1.groupby("ticker", group_keys=False)["assets"].pct_change()
        sf1 = sf1[["ticker", "datekey", "roe", "accruals", "agr"]].dropna()
```

Returns, volume, momentum, volatility

Data from sep\_weekly





```
In [19]: sep_weekly = pd.read_sql(
        """
        select ticker, date, volume, closeadj, closeunadj, lastupdated
        from sep_weekly
        where date>='2010-01-01'
        order by ticker, date, lastupdated
        """,
        conn,
        parse_dates=["date"]
    )
sep_weekly = sep_weekly.groupby(["ticker", "date", "lastupdated"]).last()
sep_weekly = sep_weekly.droplevel("lastupdated")
```

```
In [20]: sep_weekly["ret"] = sep_weekly.groupby("ticker", group_keys=False).closeadj.p
sep_weekly["annual"] = sep_weekly.groupby("ticker", group_keys=False).closeadj
sep_weekly["monthly"] = sep_weekly.groupby("ticker", group_keys=False).closeadj
sep_weekly["mom"] = sep_weekly.groupby("ticker", group_keys=False).apply(
    lambda d: (1+d.annual)/(1+d.monthly) - 1
)
sep_weekly["volatility"] = sep_weekly.groupby("ticker", group_keys=False).ret
    lambda x: x.rolling(26).std()
)
sep_weekly = sep_weekly[["ret", "mom", "volume", "volatility", "closeunadj"]]
sep_weekly = sep_weekly.reset_index()
```

Get marketcap and pb

Data from weekly



```
In [21]: weekly = pd.read_sql(
        """
        select ticker, date, marketcap, pb, lastupdated
        from weekly
        where date>='2010-01-01' and marketcap>0 and pb>0
        order by ticker, date, lastupdated
        """,
        conn,
        parse_dates=["date"]
    )
weekly = weekly.groupby(["ticker", "date", "lastupdated"]).last()
weekly = weekly.droplevel("lastupdated")
weekly = weekly.reset_index()
```

Merge



```
In [22]: df = weekly.merge(sep_weekly, on=["ticker", "date"], how="inner")
df["year"] = df.date.apply(lambda x: x.isocalendar()[0])
df["week"] = df.date.apply(lambda x: x.isocalendar()[1])
sf1["year"] = sf1.datekey.apply(lambda x: x.isocalendar()[0])
sf1["week"] = sf1.datekey.apply(lambda x: x.isocalendar()[1])
df = df.merge(sf1, on=["ticker", "year", "week"], how="left")
df = df.drop(columns=["year", "week", "datekey"])
```

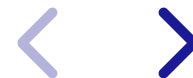
Fill ratios and growth rates forward



```
In [23]: for col in ["roe", "accruals", "agr"]:
          df[col] = df.groupby("ticker", group_keys=False)[col].apply(
              lambda x: x.ffill()
          )
```



Add sector data



```
In [24]: tickers = pd.read_sql(
        """
        select ticker, sector from tickers
        """,
        conn
    )
    df = df.merge(tickers, on="ticker")
```

Shift weekly features forward



```
In [25]: for col in ["pb", "mom", "volume", "volatility", "marketcap", "closeunadj"]:  
         df[col] = df.groupby("ticker", group_keys=False)[col].shift()
```

Calculate market volatility



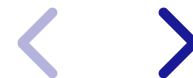
```
In [26]: import yfinance as yf
import numpy as np

price = yf.download("SPY", start="2010-01-01")["Adj Close"]
ret = price.pct_change()
vol = np.sqrt(252)*ret.rolling(21).std()
vol.name = "mktvol"
vol.index.name = "date"
vol = pd.DataFrame(vol).reset_index()
vol["year"] = vol.date.apply(lambda x: x.isocalendar()[0])
vol["week"] = vol.date.apply(lambda x: x.isocalendar()[1])
vol = vol.groupby(["year", "week"]).last()
vol = vol[["date", "mktvol"]].set_index("date")
vol["mktvol"] = vol.mktvol.shift()
vol = vol.dropna()
vol.head(3)
```

[\*\*\*\*\*100%\*\*\*\*\*] 1 of 1 completed

Out[26]:

mktvol	
date	
2010-02-12	0.192778
2010-02-19	0.198034
2010-02-26	0.199578



Merge



```
In [27]: df = df.merge(vol, on="date", how="left")
```



Filter to small caps and exclude penny stocks



```
In [28]: df = df[df.closeunadj>5]
df = df.dropna()
df["rnk"] = df.groupby("date", group_keys=False).marketcap.rank(
    ascending=False,
    method="first"
)
df = df[(df.rnk>1000) & (df.rnk<=3000)]
df = df.drop(columns=["closeunadj", "rnk"])
df = df.sort_values(by=["date", "ticker"])
```

Save data



```
In [29]: df.to_csv("../..data-2023-11-15.csv", index=False)
```

In [30]:

```
df.head()
```

Out[30]:

	ticker	date	marketcap	pb	ret	mom	volume	volatility
<b>1183</b>	AACC	2011-01-14	188.3	1.4	-0.014634	-0.184615	2.078000e+04	0.07149
<b>2047</b>	AAI	2011-01-14	1012.1	2.0	0.002677	0.438224	2.775580e+06	0.12845
<b>2117</b>	AAIC	2011-01-14	189.3	1.0	-0.010119	0.684547	3.466000e+04	0.04850
<b>4543</b>	AAON	2011-01-14	479.4	4.2	0.007778	0.528685	2.817291e+05	0.04491
<b>7543</b>	AATC	2011-01-14	63.3	1.4	-0.013960	0.008216	6.800000e+03	0.04975

