MIRC Sector Momentum Sample Algo (Part 1)

This notebook was created as a demonstration for **McMaster's MIRC** quant club. For this project, we hypothesize that a **company's earning reports** contains information about the **performance of rest of the sector**, and will attempt to **exploit any alpha** if such a relationship exists.

As an example, if we believe that Apple will post good earnings numbers (earnings per share, growth in revenue, etc.) then other tech companies also will post good earnings. Generally good earnings correlate with a jump in the stock price the next day (and vice versa with poor earnings numbers), so we will use Apple as an indicator for the rest of the sector (assuming Apple is the first in the sector to report).

Importing Relevant Packages

Here I will place any important packages that are neccessary for the functions in the notebook to run.

In [11]:
```
from quantopian.pipeline import Pipeline # Used to import from Quantop
ian
from quantopian.research import run_pipeline #
from quantopian.pipeline.factors import Returns, MarketCap # Want stoc
k performance and mkt cap (SP500)
from quantopian.pipeline.data import USEquityPricing, Fundamentals # W
ant stock pricing and earnings

from quantopian.pipeline.filters import QTradableStocksUS, StaticAsset
s # Want to know if the stock

# is tradeable

from quantopian.pipeline.classifiers.fundamentals import Sector # Need
the sector of each company

import numpy as np # Numeric calculations
import pandas as pd # Dataframe calculations
import matplotlib.pyplot as plt # Plots and charts
```

Importing the Data

We will use **Quantopian's API** to store the constituents of the **S&P500** and save the data of each date into a separate dictionary entry. Since the S&P500 is an index that tracks the performance of the 500 largest companies by market cap, it is easy to create such a pull.

In [12]:
```
def make_pipeline():

    # Pipeline factors
    close_price = USEquityPricing.close.latest # Close price of each s
tock
    market_cap = MarketCap() # Market cap of each stock

    # Pipeline Filters
    QTU = QTradableStocksUS()
    top_500_market_cap = market_cap.top(500)

    QTU_top_500 = QTU & top_500_market_cap

    # Want to make sure that the stock is listed and tradeable on that
day
    has_pricing_data = close_price.notnull()

    return Pipeline(
        # columns takes a dict of {'name of factor goes here':  factor
_variable} arguments (usually)
        columns={'close_price': close_price},
        # screen takes filter arguments
        screen=QTU_top_500 & has_pricing_data
    )
```

Test the Data

Let us take a look at the output and **make sure we have everything we need** before we dedicate the time to pull a much larger dataset.

In [13]:
```
# Want to have this in its own cell so it only needs to be run once
pipeline_test = run_pipeline(make_pipeline(), '2017-1-1', '2017-1-5')
```

Pipeline Execution Time: 0.83 Seconds

In [14]:
```
pipeline_test.keys()
```

Out[14]:
```
Index(['close_price'], dtype='object')
```

So it looks like the **pipeline_test dict** stores everything in the key ["close_price"]. We can **change/add other outputs** (which we will need later) in the columns arg in the **return Pipeline() part of the function above.** Now we will look further into the dict to see how we can access more of the data.

```
In [15]: pipeline_test["close_price"].keys()[0]
```

```
Out[15]: (Timestamp('2017-01-03 00:00:00+0000', tz='UTC', offset='C'),
          Equity(24 [AAPL]))
```

Since I picked a short time frame at the beginning of the year, there was only one day where the market was open, which is why we see only one key here. We will explore further:

```
In [18]: pipeline_test["close_price"]["2017-01-03"].head() # .head() shows just
         the first 5 entries
```

```
Out[18]: Equity(24 [AAPL])    115.84
         Equity(62 [ABT])      38.42
         Equity(64 [GOLD])     15.99
         Equity(67 [ADSK])     74.01
         Equity(76 [TAP])      97.33
         Name: close_price, dtype: float64
```

So it seems like this is what we are looking for. Above are the close prices for some of the stocks listed on the S&P500 on that day. For example, we now know that Apple's close price (AAPL) on January 3, 2017 was $115.84.

We will continue with our research project in part 2 of the MIRC tutorials.