data_cleaning

October 1, 2018

1 Data Cleaning

George Ho

```
In [1]: import numpy as np
        import pandas as pd
```

1.1 Fama-French Industrial Portfolio Data

```
In [2]: # Only read from the 24278th line onward, skipping the last row
        # (a.k.a. only the data for the equal-weighted portfolios)
        ff = pd.read_csv('48_Industry_Portfolios_daily.csv',
                         skiprows=24278, skipfooter=1, index_col=0,
                         parse_dates=True, engine='python')

        # From the years 2000 to 2016. Had to skip some weekends.
        ff = ff.loc['2000-01-03':'2016-12-30']

        # Check for missing data
        ff = ff.replace(-99.99, np.nan)
        ff = ff.replace(-999, np.nan)
        msg = ('Data missing' if ff.isna().any().any()
               else 'No missing data!')
        print(msg)
```

```
No missing data!
```

```
In [3]: ff.head()
```

```
Out[3]:             Agric  Food   Soda   Beer   Smoke  Toys   Fun    Books  Hshld  \
        2000-01-03  -2.13  0.06   0.00   0.37   0.61   1.42   0.79   0.31   -0.62
        2000-01-04  -1.79  -1.30  -1.63  0.21   -1.36  -0.06  -1.26  -1.88  -1.41
        2000-01-05   1.71  0.63   0.37   1.39   0.24   -0.28  0.54   0.82   1.11
        2000-01-06   1.93  0.37   2.71   1.15   0.47   0.03   -0.71  0.41   0.36
        2000-01-07   1.15  0.87   1.37   2.40   1.80   2.17   0.96   0.49   1.84
```

```
            Clths  ...   Boxes  Trans  Whlsl  Rtail  Meals  Banks  Insur  \
2000-01-03   0.59  ...   -0.51  -0.45   1.96  -0.54  -0.95  -1.26  -0.92
2000-01-04  -1.59  ...   -0.90  -1.46  -2.00  -1.68  -1.26  -1.92  -1.76
2000-01-05   0.93  ...   -0.93   1.02   1.54   0.62   2.10   0.27   0.40
2000-01-06   0.22  ...   -0.18   0.93   1.22  -0.32   0.07   0.59   0.69
2000-01-07   0.55  ...   -2.03   1.85   1.91   1.36   0.84   0.58   2.10

            RlEst  Fin    Other
2000-01-03  -0.22  -0.85   2.06
2000-01-04  -0.80  -1.40  -1.78
2000-01-05   0.89   0.14   0.62
2000-01-06   0.25  -0.08  -0.24
2000-01-07   0.33   0.82   3.81

[5 rows x 48 columns]
```

1.2 LIBOR Data

Data downloaded from https://fred.stlouisfed.org/series/USD3MTD156N, from the relevant time period.

```
In [4]: libor = (pd.read_csv('libor.csv', index_col=0, parse_dates=True)
                 .squeeze())

        # Coerce data to numeric values. Non-numeric data becomes NaNs.
        libor = pd.to_numeric(libor, errors='coerce')

        # Forward-fill missing values
        libor = libor.fillna(method='ffill')

        # Check for missing data
        msg = ('Data missing!' if libor.isna().any()
               else 'No missing data!')
        print(msg)
```

```
No missing data!
```

```
In [5]: def three_month_to_daily(x, N=63):
            '''
            Convert three-month (i.e. quarterly) interest rates to
            daily interest rates. A quarter is roughly 63 days.
            '''
            return 100*((1 + x/100)**(1/N) - 1)
```

```
In [6]: libor = three_month_to_daily(libor)
```

```
In [7]: libor.head()
```

```
Out[7]: DATE
        2000-01-04    0.093170
        2000-01-05    0.092983
        2000-01-06    0.092983
        2000-01-07    0.092983
        2000-01-10    0.092927
        Name: USD3MTD156N, dtype: float64
```

1.3 S&P 500 Data

Data downloaded from https://finance.yahoo.com/quote/%5EGSPC/

```
In [8]: sp500 = pd.read_csv('sp500.csv', index_col=0, parse_dates=True)

        # For returns, take the percent change of the opening prices.
        sp500 = sp500['Open'].pct_change()

        # Coerce data to numeric values. Non-numeric data becomes NaNs.
        sp500 = pd.to_numeric(sp500, errors='coerce')

        # Forward-fill missing values
        sp500 = sp500.fillna(method='ffill')

        # We still have one last NaN: the first day (which is NaN
        # because we computed percent change). Backfill for this one.
        sp500 = sp500.fillna(method='bfill')

        # Check for missing data
        msg = ('Data missing' if sp500.isna().any()
               else 'No missing data!')
        print(msg)
```

```
No missing data!
```

```
In [9]: sp500.head()

Out[9]: Date
        2000-01-03   -0.009549
        2000-01-04   -0.009549
        2000-01-05   -0.038345
        2000-01-06    0.001922
        2000-01-07    0.000956
        Name: Open, dtype: float64
```

1.4 Aggregate Data into a Single DataFrame

```
In [10]: df = ff
         df['LIBOR'] = libor
```

```
df['SP500'] = sp500

# Putting all our data together, we see that we are missing LIBOR
# data for the very first day. This is the only missing datapoint.
# Backfill for this.
df = df.fillna(method='bfill')
```

In [11]:
```
msg = ('Data missing!' if df.isna().any().any()
       else 'No missing data!')
print(msg)
```

```
No missing data!
```

In [12]:
```
df.head()
```

Out[12]:
```
                Agric  Food  Soda  Beer  Smoke  Toys   Fun  Books  Hshld  \
2000-01-03  -2.13  0.06  0.00  0.37   0.61  1.42  0.79   0.31  -0.62
2000-01-04  -1.79 -1.30 -1.63  0.21  -1.36 -0.06 -1.26  -1.88  -1.41
2000-01-05   1.71  0.63  0.37  1.39   0.24 -0.28  0.54   0.82   1.11
2000-01-06   1.93  0.37  2.71  1.15   0.47  0.03 -0.71   0.41   0.36
2000-01-07   1.15  0.87  1.37  2.40   1.80  2.17  0.96   0.49   1.84

            Clths    ...     Whlsl  Rtail  Meals  Banks  Insur  RlEst   Fin   \
2000-01-03   0.59    ...      1.96  -0.54  -0.95  -1.26  -0.92  -0.22  -0.85
2000-01-04  -1.59    ...     -2.00  -1.68  -1.26  -1.92  -1.76  -0.80  -1.40
2000-01-05   0.93    ...      1.54   0.62   2.10   0.27   0.40   0.89   0.14
2000-01-06   0.22    ...      1.22  -0.32   0.07   0.59   0.69   0.25  -0.08
2000-01-07   0.55    ...      1.91   1.36   0.84   0.58   2.10   0.33   0.82

            Other     LIBOR     SP500
2000-01-03   2.06  0.093170 -0.009549
2000-01-04  -1.78  0.093170 -0.009549
2000-01-05   0.62  0.092983 -0.038345
2000-01-06  -0.24  0.092983  0.001922
2000-01-07   3.81  0.092983  0.000956

[5 rows x 50 columns]
```

1.5 Save Data by Year

In [13]:
```
for yr in range(2000, 2017):
    df.loc[df.index.year == yr].to_csv('{}_data.csv'.format(yr))
```