Final Project:

Installation

1. Open a command prompt in the folder, and type 'pip install –r requirements.txt' to make sure that all the dependencies are installed
2. To run the file, on a command prompt type 'python final_project.py'
3. Please note: This script makes use of fix_yahoo_finance, which may bug out during execution and fetch blank dataframes. Please re-run the program
4. Also note, this program makes use of pandas version 0.19.0, since the latest version has removed pd.ols() which is required for trend analysis
5. Before running the script, please make sure that the file 'index_constituents' is at sibling level with the python script file.
6. The program takes quite a while to run due to the large amount of data it operates on.

Explanation

1. **Consider stocks constituting the S&P500 index (as of today) and download EOD data for the last 10 years for each**
   The file index_constituents consists of SnP500's constituent components that helps us download the indices constituents.

```
GSPC_current,A,AA,AAL,AAP,AAPL,ABBV,ABC,ABT,ACN,ADBE,ADI,ADM,ADP,ADS,ADSK,AEE,AEP,AES,AET,AFL,AGN,AIG,AIV,AIZ,AJG,AKAM,ALB,ALK,ALL,ALL
GSPC_current,A,AA,AAL,AAP,AAPL,ABBV,ABC,ABT,ACN,ADBE,ADI,ADM,ADP,ADS,ADSK,AEE,AEP,AES,AET,AFL,AGN,AIG,AIV,AIZ,AJG,AKAM,ALB,ALK,ALL,ALL
GSPC_current,A,AA,AAL,AAP
DJI_2015,AAPL,AXP,BA,CAT,CSCO,CVX,DD,DIS,GE,GS,HD,IBM,INTC,JNJ,JPM,KO,MCD,MMM,MRK,MSFT,NKE,PFE,PG,TRV,UNH,UTX,V,VZ,WMT,XOM
DJI_2014,AXP,BA,CAT,CSCO,CVX,DD,DIS,GE,GS,HD,IBM,INTC,JNJ,JPM,KO,MCD,MMM,MRK,MSFT,NKE,PFE,PG,T,TRV,UNH,UTX,V,VZ,WMT,XOM
DJI_2013,AXP,BA,CAT,CSCO,CVX,DD,DIS,GE,GS,HD,IBM,INTC,JNJ,JPM,KO,MCD,MMM,MRK,MSFT,NKE,PFE,PG,T,TRV,UNH,UTX,V,VZ,WMT,XOM
DJI_2012,AA,AXP,BA,BAC,CAT,CSCO,CVX,DD,DIS,GE,HD,HPQ,IBM,INTC,JNJ,JPM,KO,MCD,MMM,MRK,MSFT,PFE,PG,T,TRV,UNH,UTX,VZ,WMT,XOM
DJI_2011,AA,AXP,BA,BAC,CAT,CSCO,CVX,DD,DIS,GE,HD,HPQ,IBM,INTC,JNJ,JPM,KO,MCD,MDLZ,MMM,MRK,MSFT,PFE,PG,T,TRV,UTX,VZ,WMT,XOM
DJI_2010,AA,AXP,BA,BAC,CAT,CSCO,CVX,DD,DIS,GE,HD,HPQ,IBM,INTC,JNJ,JPM,KO,MCD,MDLZ,MMM,MRK,MSFT,PFE,PG,T,TRV,UTX,VZ,WMT,XOM
DJI_2009,AA,AXP,BA,BAC,CAT,CSCO,CVX,DD,DIS,GE,HD,HPQ,IBM,INTC,JNJ,JPM,KO,MCD,MDLZ,MMM,MRK,MSFT,PFE,PG,T,TRV,UTX,VZ,WMT,XOM
DJI_2008,AA,AXP,BA,BAC,C,CAT,CVX,DD,DIS,GE,HD,HPQ,IBM,INTC,JNJ,JPM,KO,MCD,MDLZ,MMM,MRK,MSFT,MTLQQ,PFE,PG,T,UTX,VZ,WMT,XOM
DJI_2007,AA,AIG,AXP,BA,C,CAT,DD,DIS,GE,HD,HON,HPQ,IBM,INTC,JNJ,JPM,KO,MCD,MMM,MO,MRK,MSFT,MTLQQ,PFE,PG,T,UTX,VZ,WMT,XOM
DJI_2006,AA,AIG,AXP,BA,C,CAT,DD,DIS,GE,HD,HON,HPQ,IBM,INTC,JNJ,JPM,KO,MCD,MMM,MO,MRK,MSFT,MTLQQ,PFE,PG,T,UTX,VZ,WMT,XOM
DJI_2005,AA,AIG,AXP,BA,C,CAT,DD,DIS,GE,HD,HON,HPQ,IBM,INTC,JNJ,JPM,KO,MCD,MMM,MO,MRK,MSFT,MTLQQ,PFE,PG,T,UTX,VZ,WMT,XOM
```

   Firstly, yahoo finance connection is fixed using the 'fix_yahoo_finance' library. Then, the function 'get_yahoo_data' is called in order to fetch SnP500 and DJIA index constituents data from Yahoo Finance.

2. **Starting with the first year, consider 1 Year of Stock data at a time and using simple trend analysis predict the expected movement of each of the stocks for the next year**
   Trend is found out by using pd.ols() in a function called as find_trend. This function is called within the 'portfolio' function which takes in 4 parameters, that is M, X, N, P, where M is for the M day moving average trend analysis, X is for the number of stocks to add per year till N continuous years removing bottom P% from the portfolio.

   The prediction for the next year using simple trend analysis is done within the 'portfolio' method as follows:

```
for year in trend_df.index:
    for ticker in trend_df.columns:
        data_series = ma[ma.index.year==year][ticker].dropna().values
        if data_series.size == 0:
            trend_df.loc[year, ticker] = 0
        else:
            trend_df.loc[year,ticker] = find_trend(data_series)
```

3. **Choose the top X stocks with the greatest expected movement in either direction (LONG or SHORT) and add those positions to the portfolio**
   The top X stocks are chosen with the greatest expected movement and their positions are then added to the portfolio

```
if N > trend_df.shape[0]:
    return 0
for year in range(trend_df.index[0],trend_df.index[0]+N):
    if year != trend_df.index[0]:
        port_df.loc[year+1] = port_df.loc[year]
    invest = trend_df.loc[year].abs().sort_values(ascending=False)[:X].index
    for ticker in invest:
        if(trend_df.loc[year,ticker]>0):
            port_df.loc[year+1,ticker] += 1
        else:
            port_df.loc[year+1, ticker] -= 1
```

4. **Continue adding stocks from the DJIA universe for N continuous years based on expected gains over the next year. This might lead to adding new stocks to the portfolio or adding to existing positions**

   Stocks are then added from the Dow Jones Industrial Average to the portfolio

5. **After N years, identify the worst performing stocks in the portfolio (based on annualized returns from these stocks since adding to portfolio) and exit positions (LONG or SHORT) in P% of the portfolio**

```
for year in range(trend_df.index[0]+N,2015):
    if year != trend_df.index[0]:
        port_df.loc[year + 1] = port_df.loc[year]

    worst = (return_df*port_df).cumsum().loc[year].sort_values()[:int(P*X*N)].index
    port_df.loc[year+1,worst] = 0

    invest = trend_df.loc[year].abs().sort_values(ascending=False)[:int(P*X*N)].index
    for ticker in invest:
        if (trend_df.loc[year, ticker] > 0):
            port_df.loc[year + 1, ticker] += 1
        else:
            port_df.loc[year + 1, ticker] -= 1
```

6. **Actual returns are calculated from price of the stock at the date of buying to the current price. So, you drop the stocks that was bought, but did not actually perform well.**
   Replace these positions with new stocks based on highest future expected returns
   Continue this process for 10 years

```
max = 0
for rebalance in range(10, 100, 10):
    if max < portfolio([max_period, max_initial_add, max_initial_period, rebalance]):
        max_rebalance = rebalance
```

7. **Implement the above portfolio selection strategy in Python and using a simple Machine Learning or Optimization algorithm converge out the best values of *M, X, N & P***

```
print("best values of M,X,N and P are %d,%d,%d and %f"%
      (max_period, max_initial_add,max_initial_period, max_rebalance))
```

```
best values of M,X,N and P are 190,190,9 and 90.000000
```