Danske Bank

Danske Bank Analysis

Dividends Task

Manuel Montoya Catalá
manuwhs@gmail.com

December 2016

Contents

1 Task 1 – Estimated dividend next year

In the following subsections, the proposed task will be solved. The Python programming language has been used for this purpose. In this report, I have included several extensions and graphs generated with code that I had previously made for personal purposes, only the code I have been asked for is included in the report. The report is more technical than necesary for a client, after all, everything a client wants is to feel like they have control over his/her money and that the managers are good at investing. The aim of the extensions is to perform a more realistic analysis of the assets in the portfolio.

1.1 Task 1.a

Prepare an estimate of how much dividend Anna Jensen will receive in 2017, in DKK, given that she does not make any changes to her current portfolio.

Solution

Before making the estimation we must state a few assumptions. In a simple first approach we assume that:

1. All the assets will yield the same ammount of money as they did the last time.
2. The frequency of the assets General Electric, HP Inc, Exxon Mobil Corp will be 4 times per year and the frequency of Danske Bank is once every 10 months.
3. Inflation is not taken into account.
4. No TAX is applied.

Under this assumptions, the total amount of money obtained from dividends can be calculated simply as:

```
USDDKK = 7.12  # Exchange rate
estimation = 12900 * 0.23 * 4 * USDDKK + \
        3700 *0.12*4* USDDKK  + \
        1300* 0.75*4*  USDDKK + \
        9733 * 8 * 2 # We expect it twice because of casuality
print "Checking %s" % estimation
```

The result is $280641.28 DKK$. Notice that, out of luck, we get dividends from Danske Bank twice in 2017, both in January and November.

Now we complicate things a little to make the calculator more automated. First we create a function that, given the parameter of the dividends of the asset, it computes the dividends obtained up until a certain date specified. The function uses python libraries that deal with time structures which makes the calculations easier and more accuarate. There are other ways of computing it that are commented that I have not removed because it is always useful for possible future changes. The function also takes into account the inflation if any, it is set to 2% if not specified.

```
import datetime as dt
from calendar import monthrange
from dateutil.relativedelta import relativedelta
import numpy as np
def get_dividends(Nshares, dividendValue,
                  edate, lastDiv, # Dates of start and end
                  m_period,
                  inflation = 0.02):  # Inflation
    # Nshares: The number of shares we hold
    # dividendValue: Estimated value of the dividend of a share
    # m_period: Each how many months we get dividends
    # edate: The date up to which we calculate the dividends
    # lastDiv: Last time we obtained dividends.
    # inflation: Yearly inflation rate.

    freq = 12/m_period  # Frequency
#    time_delta = dt.timedelta(weeks=4, days=2, hours=0,
```

```
#                           minutes=0, seconds=0)  # adds up to 365 days
#    lastDiv = lastDiv + dt.timedelta(m_period*(365.0/12))
    lastDiv = lastDiv + relativedelta(months=+m_period)

    ## Initilize it !
    total_dividends = 0
    t = 1 # Number of periods past
    while (lastDiv <= edate):
        div_i = dividendValue*Nshares
         # We apply the inflation
#        months_past = (sdate - lastDiv).total_seconds()/60*60*24*30
        inflation_div = np.power(1 + inflation/freq, freq*t)
        div_i = div_i/inflation_div
#        print inflation_div

        # While the time we get the dividend is less than the current price.
        total_dividends += div_i
        lastDiv = lastDiv + relativedelta(months=+m_period)
        t += 1
    return total_dividends
```

To make things more automatic, the program will read all the data directly from the exel document, which has been previously transformed to a csv separate document. We use the pandas library to load the csv and perform operations on the data, this library is used to manage relational databases, like an Exel document.

```
import pandas as pd
port_path = "./Danske Bank task/Anne_Portfolio.csv"
dataCSV = pd.read_csv(port_path,
                  sep = ',',  # Separation of the columns
                  index_col = 0, # First column is the index
                  header = 0) # The header is the first row

# Print index and columns
print dataCSV.index.values
#['General Electric' 'HP Inc' 'Exxon Mobil Corp' 'Danske Bank']
print dataCSV.columns.values
#['ISIN' 'Amount (No.)' 'Recent dividend payed (per share)' 'Dividend currency']
Companies = dataCSV.index.values
```

Once the data is loaded, we set the time variables, and the rest of the parameters. We loop over the different assets automatically, appling taxes and changing currency if needed. Then we add up the dividends from all the companies.

```
##############################################
########## Temporal Variables  ###############
##############################################
sdate = dt.date(2016,12,31) # Start date
edate = dt.date(2017,12,31) # End date
# List of the last dates we got dividends
lastDivs = [dt.date(2016,10,25), dt.date(2016,10,5),
            dt.date(2016,12,9),dt.date(2016,3,22)]

# Period  in months of the dividends
m_periods = [3, 3, 3, 10]  # Periods of the dividends
TAX = 0.25   # TAXes on profit
USDDKK = 7.12  # Exchange rate
inflation = 0.00  # yearly inflation rate

##############################################
########## Compute Dividends   ###############
```

```
##############################################

total_div = 0
# For every company
for i in range(len(Companies)):
    comp = Companies[i]
    dividends = get_dividends(dataCSV['Amount (No.)'][comp],
                              dataCSV['Recent dividend payed (per share)'][comp],
                               edate,lastDivs[i], m_periods[i],
                               inflation = inflation)

   # Check the currency
    if (dataCSV['Dividend currency'][comp] == "USD"):
        dividends = dividends * USDDKK
        print "%s changed currency" % comp

    # Apply the TAXes
    dividends = dividends * (1-TAX)

    total_div += dividends

print "The total divididends are: %0.2f" % (total_div)
```

It has been checked that the output of this approach is the same as the basic initial one. We can do some plotting to dynamically see how we obtain the dividends for each month. The next figure shows the cumulative dividends for the year 2017.

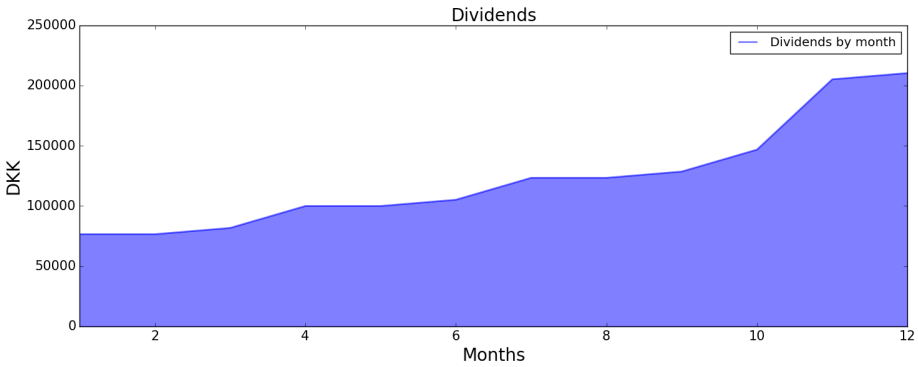

Figure 1: Dividends of 2017 by month

As we can see, we obtain dividends every month due to the different yield timings of the assets. We can observe 2 big increases in January and November, when we obtain the dividends from Danske Bank.

1.2 Task 1.b

Anna Jensen is considering buying an additional amount of 1300 Exxon Mobil Corp shares, and selling 1000 HP Inc. shares. Develop a simple calculator, in which the user can see how the mentioned transactions will affect the estimated dividend in 2017. The calculator should enable the user to change the amount of shares held for each company, and hereby get a new estimated dividend for 2017.

Solution

From the previous work, this is quite easy to implement, we just modify the values of our database:

```
dataCSV.loc['HP Inc' , 'Amount (No.)'] += 1300
dataCSV.loc['Exxon Mobil Corp' ,'Amount (No.)'] -= 1000
```

And we simply loop again over the dataset obtaining a total dividend of $263724.16 DKK$. The change does not seem to be very positive. We can plot the new portfolio againt the previous one to check it.

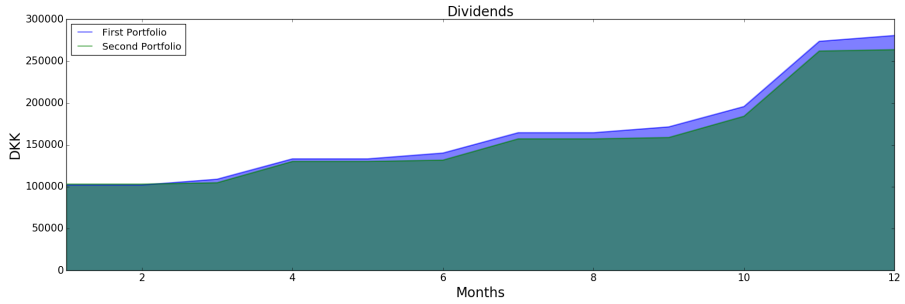

Figure 2: Dividends of 2017 by month for both portfolios

As we can see, the first month we have a better value in the new portfolio because HP yields dividends in January but when we receive the dividends from Exxon we get less money and overall we do not get as many dividends. Also, there should be transaction costs associated to this operation, and these are not taken into account.

It is also clear that we could have achieved the same result by modifing the csv document and loading it again.

1.3 Task 1.c

Before your interview, please prepare to briefly discuss the following, taking possible different needs of the customer into account. What should you take into account if the forecast is to be used to analyse the customer's:

1. Cash flow
2. Tax payments
3. Expected dividends for a longer period (For instance 10 years)

Solution

About TAX payments, they are already included in the previous model. Using the same assumptions, we can easily calculate the expected returns in 10 years. Inflation will really play a role in such a long time, the next image shows the obtained dividends by month for 10 years when we have inflation of $0\% and 2\%$.

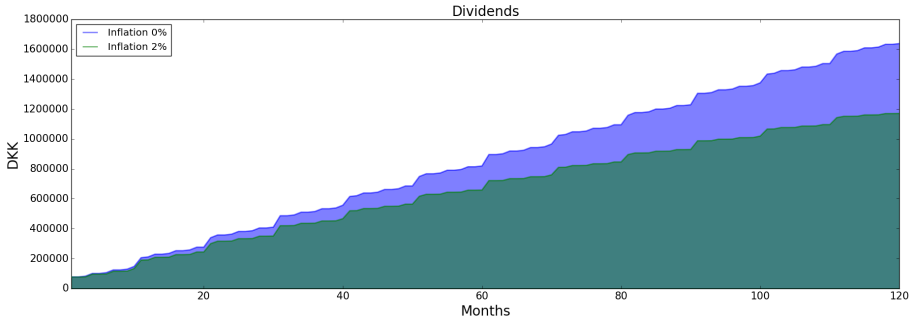

Figure 3: Dividends for 10 years time

As we can see, the inflation of 2% causes a big loss of money. Of course, once again, I do not know if this is something to tell a client, after all, it is just an assumption and all the client wants is to feel like they have control over their money and that they are making a good investing.

1.4 Appendix 1

A little more data is given about the dividends, the dividend value changes for some of the dates. This is expected since it depends on how good the company performs in that given period of time. There is a fundamental indicator of how good a company is doing, named the Dividend Yield. This is the annual dividend value divided by the price of the shares of the asset. Using Yahoo finance website, we can see the current price and calculate the Dividend Yield of each of the assets. The Yahoo names for the

given symbols Electric, HP Inc, Exxon Mobil Corp, Danske Bank are GE, HPQ, XOM, DANSKE.CO respectively.

The dividends yield respectively are $0.0288, 0.0317, 0.0330, 0.0446$. So the Danske Bank is the asset with the highest current yield dividend. Of course, if a company yields a dividend, then the price of its shares decreases that same ammount of money.

It is also important to know how well the company will do, since the price of our portfolio mainly depends on the price of the shares. I had already worked a little about this in my free time and I have been able to reuse most of my Python code.

I have downloaded monthly and daily data from Yahoo finance about the assets. The next graphs shows the monthly candlestick graph along with the volume of the companies since 2010.

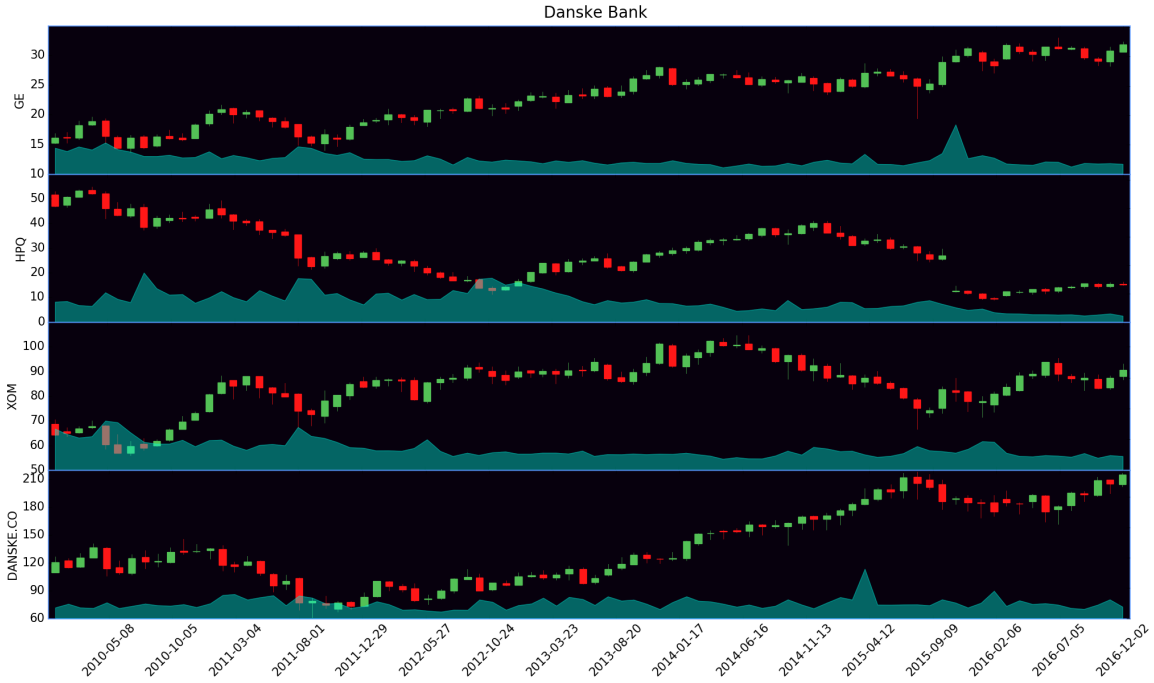

Figure 4: Monthly Price and Volume of the assets since 2010

We can see that HP is not doing great, but we should make a study on why it dropped so heavily since 2015, if it is going to recover, it would be a good investing, but the volume seems to dry up which is not a good indicator. Danske Bank seems to be reaching a resistance point so we should have caution and validate a possible outbreak.

We can make a more trading-like analysis of Danske Bank for example, using daily data and the indicators MACD and RSI. The next image shows such a graph generated in python (as all the others).

Figure 5: Daily trading indicators for Danske Bank

The MACD incator is about to become negative, which is a "sell" indicator and the RSI is leaving the "overbought" zone so the safest thing to do is to sell. Of course if suddently there is a high volume and the resistance is broken, then we should buy, always taking into account a possible fading. There is a weird high volume in the middle of 2015 that I do not know what it is about but this point could be a support line.

1.5 Appendix 2

Regarding the diversificaiton of risk in the portfolio, we could make a simple estimation using the Capital Asset Pricing Model. In this regard, we use monthly data from the past 7 years to estimate the returns and risk (standard deviation) of the assets and calculate the efficient frontier and market line for a risk-free rate of 0%. In the first interview I was told that most of the clients do not go short on assets, the possible portfolios have been constrained to only long positions.

The next graph shows the efficient frontier for the given constraints along with 10000 random portfolios that meet the restrictions. In this kind of graph, we can plot any univariate ramdom variable that we wish (usually gaussian because they have mean and std as suficient statistics) with a single point where the x-axis is its standard deviation (risk), and the y-axis is the mean (return). Each point in the graph corresponds to a portfolio (characterized by its mean return and risk), and of course, each portfolio has a set of allocation weights w associated to it.

In the image we can appreciate the different assets individually and the efficient frontier. We can also appreciate the market line for a risk free rate equal to 0, the final portfolio of the client should lie in this line, depending on its risk adversion.

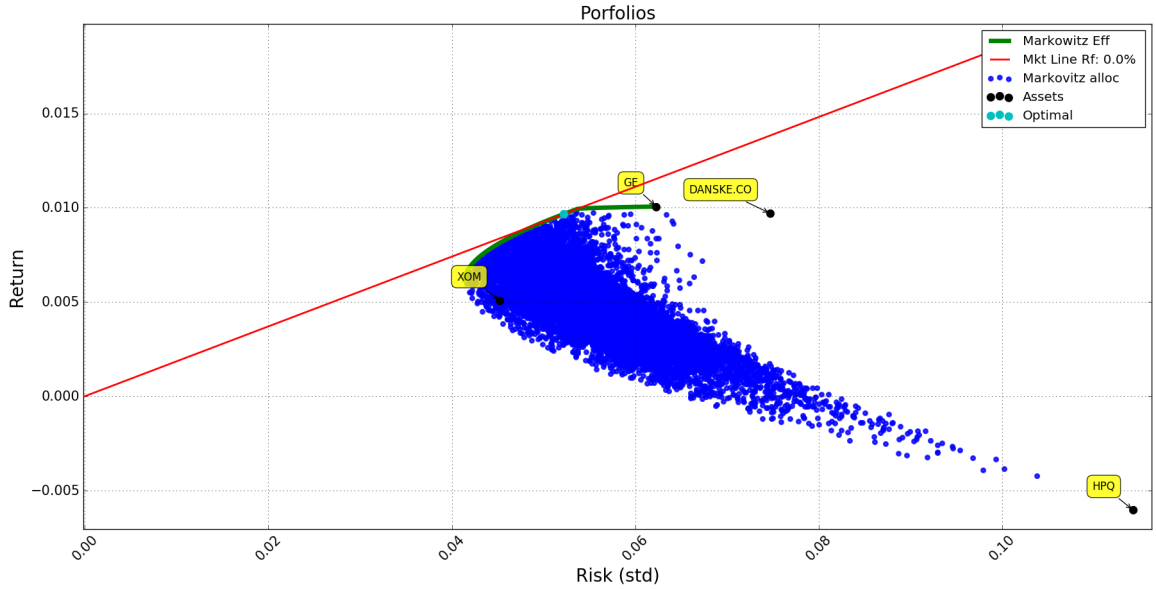

Figure 6: Daily trading indicators for Danske Bank

The weights of the optimal portfolio are:

1. GE: 5.92e-01
2. HPQ: 1.01e-08
3. XOM: 5.43e-02
4. DANSKE.CO: 3.53e-01

So General Electrics and Danske Bank are the most recommended and HP inc is not recommended, since it has had a huge drop since 2010.

Of course, this only tells us what the optimal porfolio would have been from 2010 to 2016, but as any other model, we make the assumption that the statistics will more or less hold in the future.