Financial analysis - Group Paper

University of St. Gallen

Skills: Programming - Introduction Level, Mario Silic

24.05.2021

Authors:	**Student IDs:**
Giuseppe Baccaro	19-615-996
Leonard Bätz	19-615-885
Nicolas Fort	19-608-660
Elyor Mukhtorov	19-610-005
Matias Santesso	19-609-825

Contents

1 Introduction and Code

This program lets the user choose a number of stock tickers directly from Yahoo Finance and to also input a timeframe for which the analysis should take place. It utilises this data to compare the portfolio of stocks chosen based on a number of financial and operational KPIs. With this program, the user can enter any number of tickers he wants, and the program returns key ratios such as the EBITDA or Price to Book Value ratio. The second phase of the program allows the user to analyse the returns of his selected tickers compared to the market itself, i.e. a comparison of the daily returns in the chosen timeframe. These returns are normalised based on the base year chosen by the user so that the returns of the selected stocks are comparable with each other. Finally, in the last step of the analysis, the sensitivity of the chosen stocks is analysed in relation to the sensitivity of the market. The model used in this last step is the Capital Asset Pricing Model, which describes the relationship between systematic risk and expected return for assets and is used for pricing securities relative to market returns.

```
pip install yfinance
pip install pandas_datareader
pip install plotly

import pandas as pd
from pandas_datareader import data as pdr
import seaborn as sns
from copy import copy
import matplotlib.pyplot as plt
import plotly.graph_objects as go
import plotly.express as px
import plotly.figure_factory as ff
import numpy as np
import datetime as dt
import yfinance as yf
%matplotlib inline

# define all the functions necessary to perform the CAPM analysis

# normalize function to compare S&P 500 and selected stock(s) from base = 1
def normalize(df):
    x = df.copy()
    for i in x:
        x[i] = x[i] / x[i][0]
    return x
```

```
# line chart showing adjusted close
# we consider the adjusted closing price rather the closing price to account
# for stock splits, dividend and stock issuances
# the adjusted closing price amends a stock's closing price to reflect
# that stock's value after accounting for any corporate actions.

def interactive_plot(df):
    line_chart = px.line(df.copy(), title = 'Adjusted Close Price'
    + 'of Selected Stocks and S&P500')

    line_chart.update_xaxes(title_text = 'Date', dtick="M1")
    line_chart.update_yaxes(title_text = 'Adjusted Close Price')
    line_chart.update_layout(showlegend = True)

    line_chart.show()

# develop a function to calculate the shares' daily and index's returns
# which are later used in the CAPM mode

def daily_return(df):
    df_daily_return = df.copy()

    for i in df:
        for j in range(1, len(df)):
            df_daily_return[i][j] = ((df[i][j] - df[i][j-1]) / df[i][j-1]) * 100

        df_daily_return[i][0] = 0
    return df_daily_return

# function plot a scatter plot between the selected stock
# and the S&P500 (Market)

def capm_scatter(df):
    for i in df:
        if i != 'Date' and i != '^GSPC':

            fig = px.scatter(portfolio_dailyreturn, x = '^GSPC',
                             y = i, title = i)

            b, a = np.polyfit(portfolio_dailyreturn['^GSPC'],
                              portfolio_dailyreturn[i], 1)
```

```
            fig.add_scatter(x = portfolio_dailyreturn['^GSPC'],
                            y = b*portfolio_dailyreturn['^GSPC'] + a)

            fig.show()

# function to calculate beta and alpha for all stocks in chosen portfolio

def beta_in_portfolio(df):
    for i in portfolio_dailyreturn:

        if i != 'Date' and i != '^GSPC':

            b, a = np.polyfit(portfolio_dailyreturn['^GSPC'],
                              portfolio_dailyreturn[i], 1)

            beta[i] = b

            alpha[i] = a

# enter yes to start the program and enter timeframe which has
# to retrieved out from yahoo finance

restart = ('yes')
while restart == ('yes'):
    start = input("Do you want to start the program? (yes/no) ")

    if 'yes' in start:
    print("Now please define the timeframe which you want to analyse.")
    start_date = input("What's the start date? (YYYY-MM-DD) ")
    end_date = input("What's the end date? (YYYY-MM-DD) ")
    break

#the tickers entered are directly retrieved from yahoo finance
#enter the tickers all caps without commas between them, and a space e.g: "JNJ MRK"
    portfolio = [str(i) for i in input('Please enter the tickers of the stocks you'
+' want to analyze: ').split() ]

    infos = []
```

```
    for i in portfolio:
        infos.append(yf.Ticker(i).info)

    df = pd.DataFrame(infos)
    df = df.set_index('symbol')

    fundamentals = ['trailingPE', 'pegRatio', 'priceToSalesTrailing12Months',
                'priceToBook', 'enterpriseToEbitda']

    df[df.columns[df.columns.isin(fundamentals)]]

    df['enterpriseToEbitda'].nlargest()

    df['priceToBook'].nlargest()

    plt.figure(figsize = (10, 5))
    plt.bar(df.index, df.enterpriseToEbitda)
    plt.title('Comparison of EV/EBITDA ratios')
    plt.show()

    plt.figure(figsize = (10,5))
    plt.bar(df.index, df.priceToBook)
    plt.title('Comparison of Price to Book Value ratios')
    plt.show()

# use market indexes to compare the individual stock to: S&P 500

    portfolio.append('^GSPC')


# next step is to import the data the user entered from yahoo finance,
# taking into account dividends and stocks splits --> adj close

    user_stocks = pd.DataFrame(pdr.DataReader(portfolio, 'yahoo', start_date,
                                          end_date)['Adj Close'])

# not necessary to print --> only a sanity check

    user_stocks
```

```
# interactive plot using normalized data from base year

    interactive_plot(normalize(user_stocks))

    portfolio_dailyreturn = daily_return(user_stocks)

    portfolio_dailyreturn['^GSPC']

# rm represent the market return: it averages out the daily return per the
# number of tradings days per year (252)
    rm = portfolio_dailyreturn.mean()['^GSPC'] * 252

# capm scatter using px

    capm_scatter(portfolio_dailyreturn)

    print("""This part of the program lets you perform a Capital Asset Pricing Model
    analysis of your selected stocks from Yahoo Finance, and evaluates the relative
    performance and volatility of your selected stocks compared to the
    performance of the S&P500 market index. You can select which stocks you would
    like to include as part of the CAPM analysis.""")

# getting beta and alpha for portfolio of selected stocks
# empty dictionaries to hold values, need to be cleared after first
# run through entire program

    beta = {}
    alpha = {}


    for i in portfolio_dailyreturn:

        if i != 'Date' and i != '^GSPC':

         b, a = np.polyfit(portfolio_dailyreturn['^GSPC'],
                           portfolio_dailyreturn[i], 1)

            beta[i] = b

            alpha[i] = a
```

```
# dictionary for user to see entire portfolio beta
    beta

# the risk free rate is taken from statista
# The 10 year U.S treasury bond yield is considered as the risk free rate
# for the CAPM
    rf = {
        "2021": 1.61,
        "2020": 1.01,
        "2019": 1.92,
        "2018": 2.69,
        "2017": 2.41,
        "2016": 2.45,
        "2015": 2.25,
        "2014": 2.17,
        "2013": 3.03,
        "2012": 1.76,
        "2011": 1.88,
        "2010": 3.29,
        "2009": 3.84,
        "2008": 2.22,
        "2007": 4.04,
        "2006": 4.70,
        "2005": 4.40,
        "2004": 4.22,
        "2003": 4.25,
        "2002": 3.82,
        "2001": 5.03,
        "2000": 5.11,
    }

# asks user for ticker of selected portfolio and calculates CAPM one ticker
# at a time

    rf_inputted = rf.get(input("Enter the ending year: "))
```

```
    for i in portfolio:

        ER_user = print("the company's cost of equity is: ", rf_inputted +
                (beta.get(input('Enter the ticker of the stock : ')) *
                 (rm - rf_inputted)))
    ER_user

restart = input("Do you want to restart the program? (yes/no)  ")


if restart == ('yes'):
    print("Let's start over!")
elif restart == ('no'):
    print("Thank you for your time! Bye!")
```

2 Example Run

The following is an example output of the code with the chosen tickers being JNJ, PFE and MRK over a period of two years, between 19.05.2019 and 19.05.2021. The first KPI used is the comparison of the enterprise value (EV) to the earnings before interest, taxes, depreciation and amortisation (EBITDA). As one can see based on the output, the stock with the highest ratio is JNJ with 16.225% compared to MRK which had a ratio of 12.077% over the observed period. Both ratios are quite high however, which may indicate a potential overvaluation of the companies so caution and due diligence should be taken. The second graph with the PV ratio shows us that MRK traded at a value 7.42 times higher than its book value, compared to a PV ratio of 3.24 for PFE. This is again a relatively high value for MRK, signalling an overvalued company.

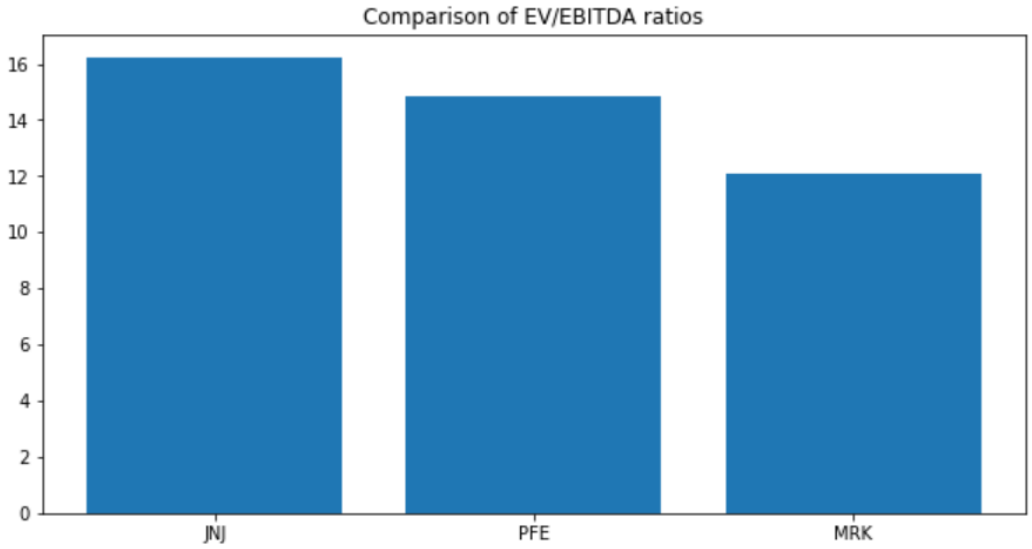
Program: "Do you want to start the program? (yes/no) "
Example User: yes

Program: "Now please define the timeframe which you want to analyse. What's the start date? (YYYY-MM-DD)"
Example User: 2019-05-19
Program: "What's the end date? (YYYY-MM-DD)"
Example User: 2021-05-19

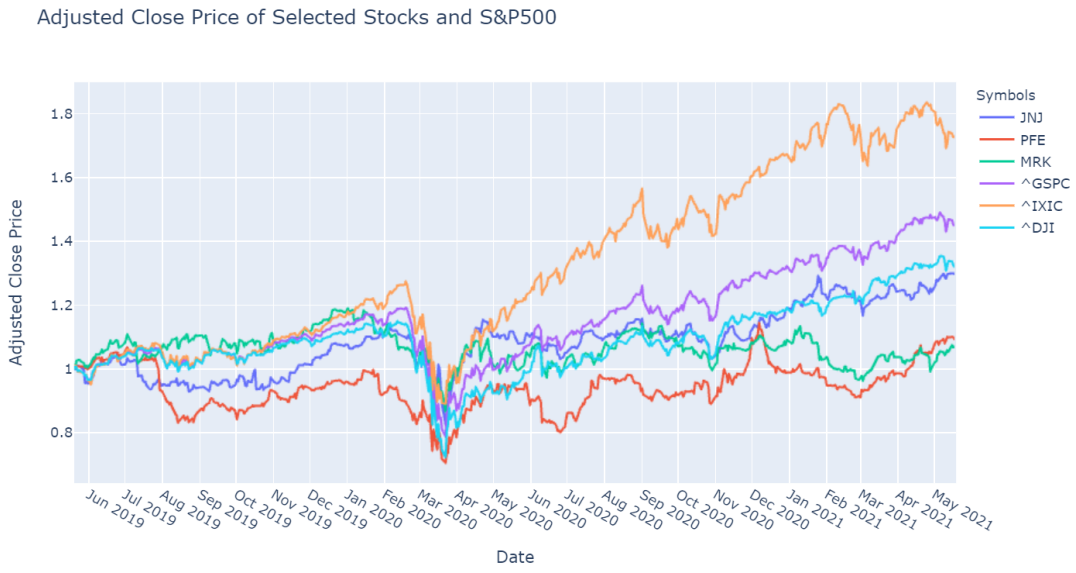
Program: "Please enter the tickers of the stocks you want to analyze:"
Example User: "JNJ PFE MRK"

Program:

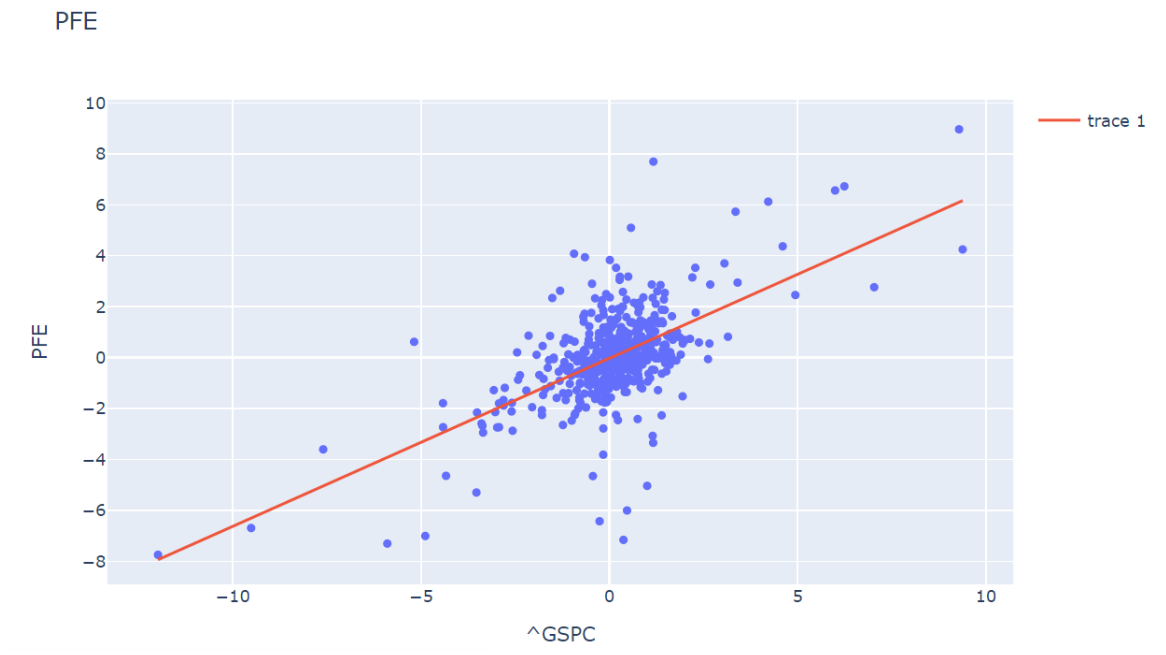
symbol	trailingPE	priceToSalesTrailing12Months	enterpriseToEbitda	priceToBook	pegRatio
JNJ	30.127188	5.333156	16.225	6.820363	2.31
PFE	20.167173	4.801949	14.818	3.245292	0.82
MRK	28.540989	4.167455	12.077	7.424142	1.30

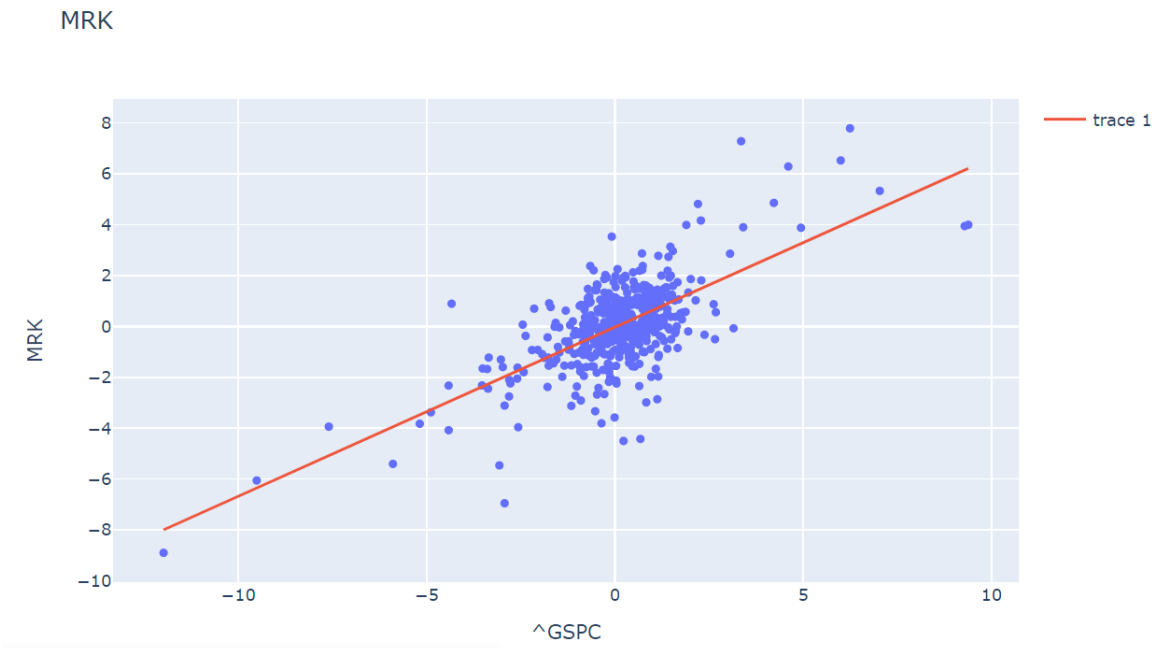

Comparison of EV/EBITDA ratios
16
14
12
10
8
6
4
2
0
JNJ
PFE
MRK


Comparison of Price to Book Value ratios
7
6
5
4
3
2
1
0
JNJ
PFE
MRK



Program: "This part of the program lets you perform a Capital Asset Pricing Model analysis of your selected stocks from Yahoo Finance, and evaluates the relative performance and volatility of your selected stocks compared to the performance of the SP500 market index. You can select which stocks you would like to include as part of the CAPM analysis."


PFE
PFE
^GSPC
trace 1
10
8
6
4
2
0
−2
−4
−6
−8
−10
−5
0
5
10


MRK
MRK
^GSPC
trace 1
8
6
4
2
0
−2
−4
−6
−8
−10
−5
0
5
10

Program: shows dictionary for user to see entire portfolio beta

'JNJ': 0.6470505991268048,
'MRK': 0.6643836130561339,
'PFE': 0.659069741239585

Program: "Enter the ending year:"
Example User: 2021
Program: "Enter the ticker of the stock:"
Example User: JNJ
Program: "the company's cost of equity is: 14.759207615641678"
Program: "Enter the ticker of the stock:"
Example User: PFE
Program: "the company's cost of equity is: 15.0053064879877198
Program: "Enter the ticker of the stock:"
Example User: MRK
Program: "the company's cost of equity is: 15.124759522459403

Program: "Do you want to restart the program? (yes/no)"
Example User: no
Program: "Thank you for your time! Bye!"

——this is the end of our paper——

Thank you for your attention!