```python
# -*- coding: utf-8 -*-
"""
Created on Mon May 28 15:03:44 2018

@author: Tawanda Vera
"""

from __future__  import print_function

"""
Part 1
Consider a mutual fund with beta of 0.8 which has an expected rate of return of
 14%. If risk-free rate of return is rf = 5%, and you expect the rate of
 return on market portfolio to be 15%.

Use Python to address the following questions
a) Would you be interested in investing in the fund? If so, what is the Alpha
of the fund.
b) What passive portfolio comprised of a market-index portfolio and a
money-market account would have the same beta as the fund?
"""
import numpy as np
import matplotlib.pyplot as mp
import pandas_datareader.data as web
import datetime

#Part 1a)
rf = 0.05  # Risk free rate
bp = 0.8   # Beta
rm = 0.15  # expected rate of return on market portfolio
rp = 0.14   # Expected rate of return of mutual fund

# alpha
E_rp = rf + bp *(rm - rf)  #

alpha = rp - E_rp

print("alpha:", np.round(100*alpha,2),'%')

# Alpha = 1%
# The fund has a positive alpha of 1% so i will invest the mutual fund

# Part 1b) What passive portfolio comprised of a market-index portfolio and a
# money-market account would have the same beta as the fund?

w1 = 0.8  # 80% in the market index portfolio
w2 = 0.2  # 20% in the money market

# Passive Portfolio
passive =  w1 * rm + w2* rf

alpha_b = rp - passive

print('Passive alpha:', np.round(100*alpha_b,2),'%')

# Note: show that the difference between the expected rate of return on this
# passive portfolio and that of the fund equals the alpha from question 1.
```

```
print('Same beta as the fund:', alpha == alpha_b)

# THe same beta portfolio will have 80% market return and 20% money market at
# risk free rate

################################################################################
"""
Part 2
Consider the following data for a one-factor economy. All portfolios are
assumed to be well diversified.
"""

# Input
ret_A = np.array([0.12, 0.13, 0.14, 0.15, 0.16, 0.17, 0.18, 0.19, 0.2])
beta_A = np.array([1.2, 1.3, 1.4, 1.5, 1.6, 1.7])
ret_F = np.array([0.06, 0.07, 0.08, 0.09])
beta_F =np.array([0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9])
ret_E = 0.08
beta_E = 0.6

#Expected returns
r_a = ret_A.mean()    # 0.16
r_f = ret_F.mean()    # 0.075

#Betas
b_a = beta_A.mean()    # 1.45
b_f = beta_F.mean()    # 0.45


# risk free rate and return of return on the market
# r_a = r + b_a * (r_m - r)   # return for A
# 0.16 = r + 1.45 * r_m - 1.45 * r
#0.16 = 1.45 * r_m - 0.45 * r    # Equation 1
# Equation 2
# r_f = r + b_f * (r_m - r)   # return for F
 # 0.075 = r + 0.45 * r_m - 0.45 * r
# 0.075 = 0.45 * r_m - 0.55 * r  # Equation 2

# Solve the linear equation

A = np.array([[1.45,0.45], [0.45,0.55]])

B = np.array([0.16, 0.075])

C = np.linalg.solve(A, B)

print ('r_m , r:', np.round(C, 2))

r_m = 0.09   #  market return
r  = 0.06    #  risk free rate

# Then we can use this to values to figure out whether the a for portfolio E
#  is zero.  The formula is:

alpha_E = (ret_E - r) - beta_E * (r_m - r)

print('alpha_E:', np.round(100*alpha_E, 2), '%')
```

```
print('Portfolio E is equal to zero:', np.round(100* alpha_E, 2) == 0.00)

# Portfolio E is not equal to zero,
#So the alpha is positive which implies that the portfolio is undervalued.
# So there is an arbitrage opportunity and we can exploit this by buying this
# portfolio.

# We can construct the zero-beta portfolio, using portfolios A and E.

w_a = np.round(-beta_E /(b_a - beta_E))       # w_a = 1

w_e = np.round(b_a / (b_a - beta_E))  # w_e = 2

# So, we should sell short 1 units of A, and buy 2 units of E.
# It is easy to verify that the beta for this new portfolio is zero:

b_new = np.round(b_a * w_a + w_e * beta_E)

print('b_new is equal to zero:', b_new == 0)


# The return on this portfolio is:

αA = (r_a - r) - b_a * (r_m - r)    # alpha of A
αE = (ret_E - r) - beta_E * (r_m - r)    # alpha of E

r_new = r + αA * w_a + αE * w_e  # the new return on the portfolio

print('new returns:', np.round(100* r_new), '%')


# So our arbitrage strategy is to short portfolio F (borrow at 7%) and then
# invest in this new portfolio which earns 8%.

################################################################################
"""
Part 3
Suppose the economy can be .in one of the following two states:
(i) Boom or “good” state and
(ii) Recession or “bad” state.
Each state can occur with an equal opportunity. The annual return on the market
 and a certain security X in the two states of the economy are as follows:
• Market: at the end of the year, the market is expected to yield a return of
30% in the good state
and a return of (-10%) in the bad state;
• Security X: at the end of the year, the security is expected to yield a
return of 40% in the good
state and a return of (-35%) in the bad state;

Furthermore, assume that annual risk-free rate of return is 5%.
"""
# Calculate the beta of security X relative to the market

# Market Portfolio
gd = 30        # good state return
p = 0.5      # probability
```

```
bd = -10       # bad state return

e_rm = gd * p + bd * (1-p)  # expected market returns

var_rm = p *(gd - e_rm)**2 + (1-p)*(bd - e_rm)**2 # variance of market portfolio

print ('Expected market returns:', np.round(e_rm),
       'Variance of market portfolio:', np.round(var_rm,2))

# Security
gs = 40        # good state return
bs = -35       # bad state return

e_rs = gs * p + bs * (1-p)  # expected market returns

var_rs = p *((gs - e_rs)**2) + (1-p)*((bs-e_rs)**2) # variance of market portfolio

print ('Expected market returns:', e_rs,
       'Variance of market portfolio:', np.round(var_rs,2))

# Covariance of X and M

cov_X_M = p *(gs - e_rs)*(gd - e_rm) + (1-p)*(bs-e_rs)*(bd - e_rm)

# Beta of Security X
beta_X = cov_X_M / var_rm

print( 'Beta of Security X:', beta_X)

# Security's Alpha
rfr = 5  # risk free rate

a_X = e_rs - (rfr + beta_X * (e_rm - rfr))

print( 'Alpha of Security X:', a_X)



x = np.linspace(-50, 50, 1000) # 1000 values between 0 and 100
x = np.arange(-50, 50)     # -50, -49, ... 49, 50
y = a_X + beta_X * (x - rfr)

mp.scatter(x, y, alpha=0.2)


################################################################################
# Part 4

# Download the data for the last 15 years for sp500 and DJTA
sdt = datetime.datetime.now() - datetime.timedelta(15*365)
edt = datetime.datetime.now()
sp500 = web.DataReader("SP500", "fred", sdt, edt)

sp500.tail()
```

```
ticker =('SP500')
# Construct a simple trading system that goes long when sp500 closes above its
# 200 Day Exponential Moving Average of Close Prices (200 DEMA) and closes in
# position and goes short when prices close below the 200 DEMA.

def Periods(data):
    import numpy as np
    # Create the indicator (20 day SMA)
    data['ticker'] = data
    data['200DEMA'] = data.ticker.ewm(span=200, adjust=True).mean()

    # Generate signals based on trading logic
    data['Signal'] = data.ticker - data['200DEMA']
    #
    Exit = 0   # Deviations at zero is used as the threshold
    data['Long'] = np.where(data['Signal'] > Exit, 1, 0)   # long
    data['Short'] = np.where(data['Signal'] < Exit, -1, data['Long']) # short
    bullish = data['Short'].value_counts()
    bearish = data['Long'].value_counts()

    # Calculate Beta
    cov = np.cov(data.ticker, data.SP500)
    sd_prd = data.ticker.std() * data.SP500.std()
    beta = cov/sd_prd
    return bullish, bearish, beta


print('bullish, bearish days, beta:', Periods(sp500))

"""
On 2012 days (Bullish Periods, 1)
On 616 days, (Bearish Periods, -1)

The results show more bullish periods compared to bearish periods
Bull and bear markets are key elements in analyzing and predicting financial
markets. Investors who actively manage their portfolios seek to invest in
assets with bullish prospects and stay away from assets with bearish prospects,
 or even to go short in those. To successfully implement such a strategy,
 they require accurate identification and prediction of bullish and bearish periods.
 Bull markets are commonly understood as prolonged periods of gradually
 rising prices, while bear markets are characterized by falling prices and
 higher volatility than during bull markets.
"""
################################################################################
```