008 - Comparative Analysis Between Growth and Value ETFs

December 18, 2023

1 #008 Comparative Analysis Between Growth and Value ETFs

1.1 libs

[1]:
```
#import Libraries
import pandas as pd
from pandas_datareader import data as pdr
import numpy as np
import random
import plotly.graph_objects as go
import seaborn as sns
```

1.2 Functions

[2]:
```
# Define a function using Plotly Express
def plotly_data(df, title):

    # Create figure
    fig = go.Figure()

    # Set title
    fig.update_layout(title_text = title)

    # For loop that plots all stock prices in the pandas dataframe df

    for i in df.columns[0:]:
        # Add range slider
        #fig.update_layout(xaxis=dict(rangeselector =␣
↪dict(buttons=list([dict(count=1, label="1m", step="month",␣
↪stepmode="backward"), dict(count=6, label="6m", step="month",␣
↪stepmode="backward"), dict(count=1, label="YTD", step="year",␣
↪stepmode="todate"), dict(count=1, label="1y", step="year",␣
↪stepmode="backward"), dict(step="all")])), rangeslider=dict( visible=True),␣
↪type="date"))
        # Add line graph
        fig.add_scatter(x = df.index, y = df[i], name = i)
        # Update Layout
        fig.update_layout({'plot_bgcolor': "white"})
```

```
        #fig.update_traces(line_width = 3)
        fig.update_layout(legend=dict(orientation="h",))

    fig.show()

# Define a function using Plotly Express, changes axis y to logarithm scale
def log_plotly_data(df, title):

    # Create figure
    fig = go.Figure()

    # Set title
    fig.update_layout(title_text = title)

    # For loop that plots all stock prices in the pandas dataframe df

    for i in df.columns[0:]:
        # Add range slider
        #fig.update_layout(xaxis=dict(rangeselector =␣
 ↪dict(buttons=list([dict(count=1, label="1m", step="month",␣
 ↪stepmode="backward"), dict(count=6, label="6m", step="month",␣
 ↪stepmode="backward"), dict(count=1, label="YTD", step="year",␣
 ↪stepmode="todate"), dict(count=1, label="1y", step="year",␣
 ↪stepmode="backward"), dict(step="all")])), rangeslider=dict( visible=True),␣
 ↪type="date"))
        # Add line graph
        fig.add_scatter(x = df.index, y = df[i], name = i)
        # Update Layout
        fig.update_layout({'plot_bgcolor': "white"})
        #fig.update_traces(line_width = 3)
        fig.update_layout(legend=dict(orientation="h",))

    #changes y to logarithm scale
    fig.update_yaxes(type="log")
    fig.show()

# Function to scale stock prices based on their initial starting price
# The objective of this function is to set all prices to start at a value of 1
def price_scaling(raw_prices_df):
    scaled_prices_df = raw_prices_df.copy()
    for i in raw_prices_df.columns[0:]:
          scaled_prices_df[i] = raw_prices_df[i]/raw_prices_df[i][0]
    return scaled_prices_df
```

1.3 8.1 Import and Analyse Data

iShares S&P 500 Value ETF (IVE)

iShares S&P 500 Growth ETF (IVW)

[3]:
```
#read CSV file
df = pd.read_csv("IVE_IVW")
#import to df, replace the colum Date to Index
df.set_index(['Date'], inplace = True)
#calc scaled prices
scaled_df = price_scaling(df)
# Calculate the portfolio percentage daily return and replace NaNs with zeros
#calculate percentage daily return
p_change_df = df.pct_change() * 100
p_change_df.replace(np.nan, 0, inplace = True)
```

[4]:
```
df.describe().round(2)
```

[4]:

	IVE	IVW
count	5031.00	5031.00
mean	75.61	28.72
std	37.41	20.15
min	22.09	7.42
25%	44.46	12.63
50%	66.67	21.25
75%	100.12	40.01
max	173.21	83.42

[5]:
```
p_change_df.describe().round(2)
```

[5]:

	IVE	IVW
count	5031.00	5031.00
mean	0.04	0.05
std	1.22	1.21
min	-11.19	-11.88
25%	-0.43	-0.43
50%	0.07	0.09
75%	0.58	0.60
max	10.70	10.60

[6]:
```
scaled_df.describe().round(2)
```

[6]:

	IVE	IVW
count	5031.00	5031.00
mean	2.19	2.74
std	1.08	1.92
min	0.64	0.71
25%	1.29	1.21
50%	1.93	2.03
75%	2.89	3.82
max	5.01	7.97

1.4 8.2 Plotly Data

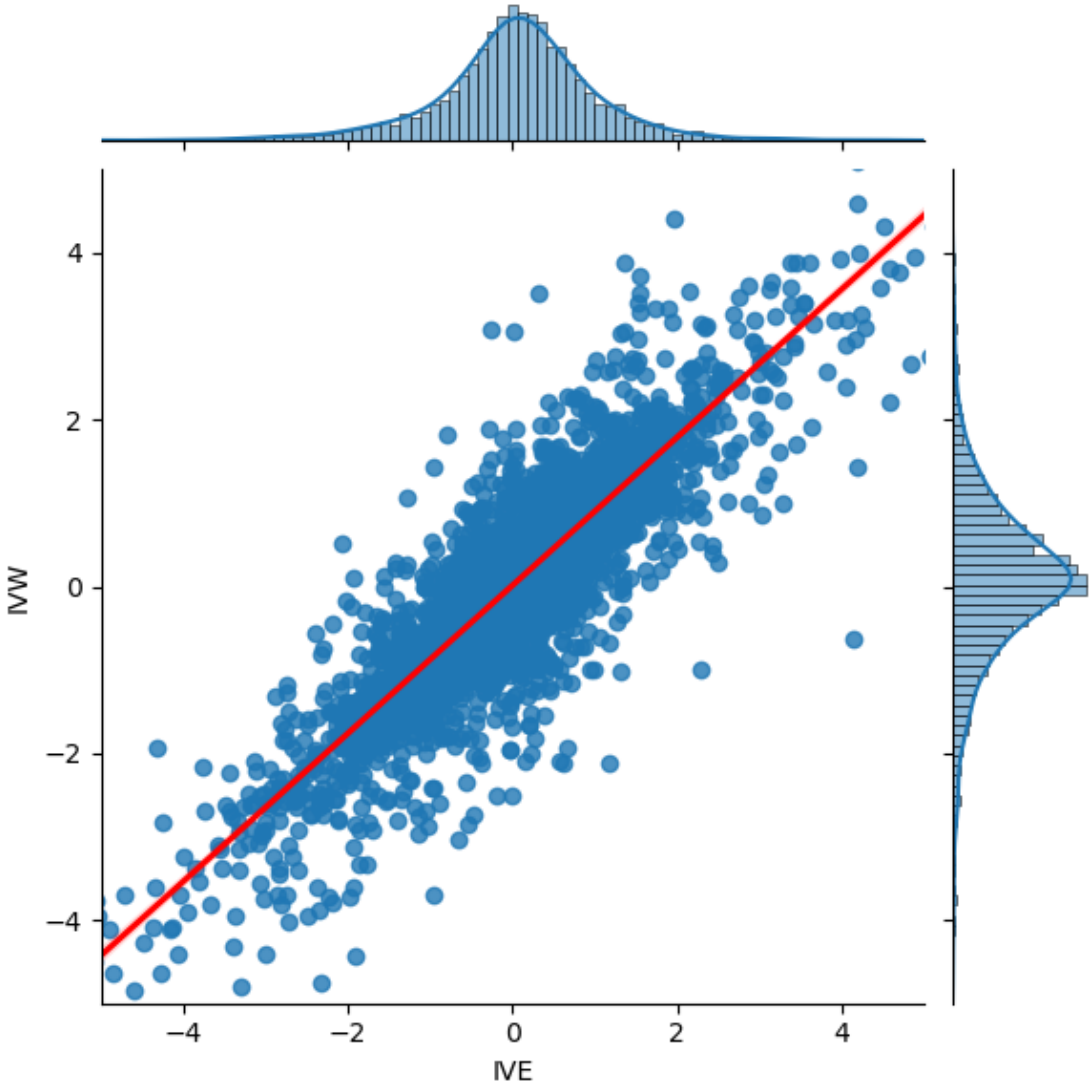
[7]:
```
plotly_data(scaled_df, "")
log_plotly_data(scaled_df, "")
plotly_data(p_change_df,"percentage Daily return")
```

[8]:
```
import plotly.express as px
# Plot histograms for stocks daily returns using plotly express
fig = px.histogram(p_change_df)
fig.update_layout({'plot_bgcolor': "white"})
```

[9]:
```
#joint 2 stocks into a scatter
t = 5

sns.jointplot(x= p_change_df['IVE'], y= p_change_df['IVW'], kind="reg",␣
 ↪line_kws=dict(color="r"), marker = "o", xlim = {-t, t}, ylim = {-t, t})
```

[9]:
```
<seaborn.axisgrid.JointGrid at 0x2625a1a4690>
```


IVW
IVE
4
2
0
−2
−4
−4
−2
0
2
4