Chapter_03_probability

June 4, 2022

1 Probability

1.0.1 After this Chapter you will be able to:

- Compute and understand Random variable
- Compute and understand Intersection, union, independency
- Compute and understand Conditional probability

1.0.2 Exercise (Trading):

- Apply conditional probability to stock market

Join our community: https://discord.gg/wXjNPAc5BH

Read our book: https://www.amazon.com/gp/product/B09HG18CYL

Quantreo's YouTube channel: https://www.youtube.com/channel/UCp7jckfiEglNf_Gj62VR0pw

[ ]:
```
!pip install yfinance
```

```
Collecting yfinance
  Downloading yfinance-0.1.67-py2.py3-none-any.whl (25 kB)
Requirement already satisfied: numpy>=1.15 in /usr/local/lib/python3.7/dist-
packages (from yfinance) (1.19.5)
Requirement already satisfied: requests>=2.20 in /usr/local/lib/python3.7/dist-
packages (from yfinance) (2.23.0)
Requirement already satisfied: multitasking>=0.0.7 in
/usr/local/lib/python3.7/dist-packages (from yfinance) (0.0.10)
Collecting lxml>=4.5.1
  Downloading lxml-4.7.1-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x86_64.m
anylinux_2_24_x86_64.whl (6.4 MB)
     |                  | 6.4 MB 5.0 MB/s
Requirement already satisfied: pandas>=0.24 in
/usr/local/lib/python3.7/dist-packages (from yfinance) (1.1.5)
Requirement already satisfied: pytz>=2017.2 in /usr/local/lib/python3.7/dist-
packages (from pandas>=0.24->yfinance) (2018.9)
Requirement already satisfied: python-dateutil>=2.7.3 in
/usr/local/lib/python3.7/dist-packages (from pandas>=0.24->yfinance) (2.8.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-
packages (from python-dateutil>=2.7.3->pandas>=0.24->yfinance) (1.15.0)
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-
```

```
packages (from requests>=2.20->yfinance) (2.10)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in
/usr/local/lib/python3.7/dist-packages (from requests>=2.20->yfinance) (1.24.3)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.7/dist-packages (from requests>=2.20->yfinance)
(2021.10.8)
Requirement already satisfied: chardet<4,>=3.0.2 in
/usr/local/lib/python3.7/dist-packages (from requests>=2.20->yfinance) (3.0.4)
Installing collected packages: lxml, yfinance
  Attempting uninstall: lxml
    Found existing installation: lxml 4.2.6
    Uninstalling lxml-4.2.6:
      Successfully uninstalled lxml-4.2.6
Successfully installed lxml-4.7.1 yfinance-0.1.67
```

[ ]:
```
!pip install ta
```

```
Collecting ta
  Downloading ta-0.8.0.tar.gz (24 kB)
Requirement already satisfied: numpy in /usr/local/lib/python3.7/dist-packages
(from ta) (1.19.5)
Requirement already satisfied: pandas in /usr/local/lib/python3.7/dist-packages
(from ta) (1.1.5)
Requirement already satisfied: python-dateutil>=2.7.3 in
/usr/local/lib/python3.7/dist-packages (from pandas->ta) (2.8.2)
Requirement already satisfied: pytz>=2017.2 in /usr/local/lib/python3.7/dist-
packages (from pandas->ta) (2018.9)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-
packages (from python-dateutil>=2.7.3->pandas->ta) (1.15.0)
Building wheels for collected packages: ta
  Building wheel for ta (setup.py) … done
  Created wheel for ta: filename=ta-0.8.0-py3-none-any.whl size=28895
sha256=2bfd045699199b5540345dba78ce79a96ab7fe694e2722e029fc37b50cd68281
  Stored in directory: /root/.cache/pip/wheels/7e/da/86/65cba22446ae2ef148de2079
907264ef27feecfb7f51a45e0d
Successfully built ta
Installing collected packages: ta
Successfully installed ta-0.8.0
```

[ ]:
```
!pip install scipy==1.7.1
```

```
Requirement already satisfied: scipy==1.7.1 in /usr/local/lib/python3.7/dist-
packages (1.7.1)
Requirement already satisfied: numpy<1.23.0,>=1.16.5 in
/usr/local/lib/python3.7/dist-packages (from scipy==1.7.1) (1.19.5)
```

[ ]:
```
import numpy as np
import pandas as pd
```

```python
import yfinance as yf

import warnings
warnings.filterwarnings("ignore")
```

[ ]:
```python
# The code here will allow you to switch your graphics to dark mode for those␣
↪who choose to code in dark mode
import matplotlib.pyplot as plt

import matplotlib as mpl
from matplotlib import cycler
colors = cycler('color',
                ['#669FEE', '#66EE91', '#9988DD',
                 '#EECC55', '#88BB44', '#FFBBBB'])
plt.rc('figure', facecolor='#313233')
plt.rc('axes', facecolor="#313233", edgecolor='none',
       axisbelow=True, grid=True, prop_cycle=colors,
       labelcolor='gray')
plt.rc('grid', color='474A4A', linestyle='solid')
plt.rc('xtick', color='gray')
plt.rc('ytick', direction='out', color='gray')
plt.rc('legend', facecolor="#313233", edgecolor="#313233")
plt.rc("text", color="#C9C9C9")
```

[ ]:
```python
# Import the data and create variations of GOOGLE
df = yf.download("GOOG")[["Adj Close"]].pct_change(1).dropna()
df
```

```
[*********************100%***********************]  1 of 1 completed
```

[ ]:
```
            Adj Close
Date
2004-08-20   0.079430
2004-08-23   0.010064
2004-08-24  -0.041408
2004-08-25   0.010775
2004-08-26   0.018019
…                 …
2021-12-13  -0.013254
2021-12-14  -0.011820
2021-12-15   0.016541
2021-12-16  -0.017168
2021-12-17  -0.014054

[4364 rows x 1 columns]
```

2 Probability

[ ]:
```python
# Compute the chance of a increase
p_increase = len(df[df["Adj Close"]>0])/len(df)*100

# Compute the chance of a decrease
p_decrease = len(df[df["Adj Close"]<0])/len(df)*100

# %Increase + %Descrease close to one but not mandatory equal 1
print(f"Weights| \t Increase: {'%.2f' % p_increase} \t Decrease: {'%.2f' %␣
↪p_decrease}")
```

```
Weights|         Increase: 52.66         Decrease: 47.32
```

3 Conditional Probability

[ ]:
```python
# Bollinger Band class
from ta.volatility import BollingerBands
```

[ ]:
```python
# Import "Stationary" data

# Import close price
df = yf.download("EURUSD=X")[["Adj Close"]]

# Create the returns
df["return"] = df["Adj Close"].pct_change(1).dropna()
```

```
[*********************100%***********************]  1 of 1 completed
```

[ ]:
```python
# Initialize BB class
indicator_bb = BollingerBands(close=df["Adj Close"], window=20, window_dev=1.5)

# Add Bollinger Bands features
df['bb_bbm'] = indicator_bb.bollinger_mavg()
df['bb_bbh'] = indicator_bb.bollinger_hband()
df['bb_bbl'] = indicator_bb.bollinger_lband()
```

[ ]:
```python
df[['Adj Close', 'bb_bbl', 'bb_bbh']].loc["2009"].plot(figsize=(15,8))
```

[ ]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f6ffcd021d0>
```

[ ]:

```
# Import close price
df = yf.download("ETH-USD")[["Adj Close"]]

# Create the returns
df["return"] = df["Adj Close"].pct_change(1).dropna()

# Initialize BB class
indicator_bb = BollingerBands(close=df["Adj Close"], window=20, window_dev=1.5)

# Add Bollinger Bands features
df['bb_bbm'] = indicator_bb.bollinger_mavg()
df['bb_bbh'] = indicator_bb.bollinger_hband()
df['bb_bbl'] = indicator_bb.bollinger_lband()


# Shift the values
df["Adj Close Yesterday"] = df["Adj Close"].shift(1)
df["bbh Yesterday"] = df["bb_bbh"].shift(1)

# Drop the row containing missing values
df = df.dropna()


# Variable initialization
n = 10
dates = []
```

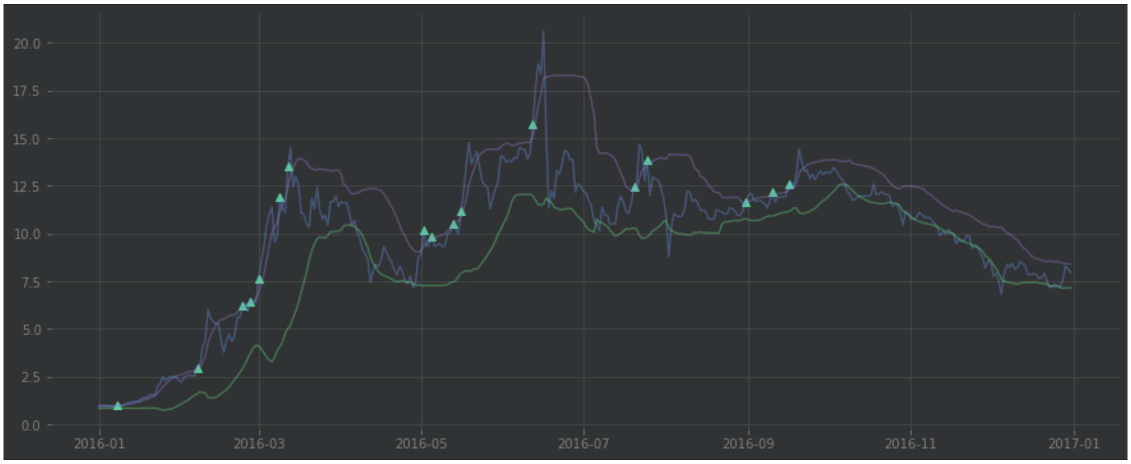
```
returns = []

#
for i in range(len(df)-n-1):
  row = df.iloc[i:i+1,:]

  # Create signal conbdition
  if (row["Adj Close Yesterday"].values[0] < row["bbh Yesterday"].values[0])␣
 ↪and\
  (row["Adj Close"].values[0] > row["bb_bbh"].values[0]):

    # Save the date
    dates.append(row.index[0])

    # Compute the returns from signal to 10days later
    ret = []
    for day in range(1,n):
      ret.append(df.iloc[i+day:i+1+day,:]["return"].values[0])
    returns.append(ret)

# List of lists to dataframe
df_returns = pd.DataFrame(returns).cumsum(axis=1)
```

```
[*********************100%***********************]  1 of 1 completed
```

[ ]:

```
# We plot all the signl to be sure that they be correct

# Select all signal in a index list to plot only this points
idx = dates
year="2016"

# Adapt the size of the graph
plt.figure(figsize=(15,6))

# Plot the points of the open long signal in green
plt.scatter(df.loc[idx].loc[year].index, df.loc[idx]["Adj Close"].loc[year],␣
 ↪color= "#57CE95", marker="^")

# Plot the points of the close long signal in blue

# Plot the rsi to be sure that the conditions are completed
plt.plot(df["Adj Close"].loc[year].index, df["Adj Close"].loc[year], alpha=0.35)
plt.plot(df["bb_bbl"].loc[year].index, df["bb_bbl"].loc[year], alpha=0.35)
plt.plot(df["bb_bbh"].loc[year].index, df["bb_bbh"].loc[year], alpha=0.35)
```

```python
# Show the graph
plt.show()
```



[ ]:

```python
# Compute %Increase %Decrease of returns from signal to 10days later
for i in range(n-1):
  g = df_returns.iloc[:,i:i+1].values
  p_increase = len(g[g>0])/len(g)*100
  p_decrease = len(g[g<0])/len(g)*100

  print(f"Weights day {i+1}| \t Increase: {'%.2f' % p_increase} %\t Decrease:
 ↪ {'%.2f' % p_decrease} %")
print(f" ")

# Compute %Increase %Decrease of the asset on all the situations
p_increase = len(df[df["return"]>0])/len(df)*100
p_decrease = len(df[df["return"]<0])/len(df)*100
print(f"Weights| \t Increase: {'%.2f' % p_increase} % \t Decrease: {'%.2f' %␣
 ↪p_decrease} % ")
```

```
Weights day 1|    Increase: 57.66 %       Decrease: 42.34 %
Weights day 2|    Increase: 57.66 %       Decrease: 42.34 %
Weights day 3|    Increase: 70.27 %       Decrease: 29.73 %
Weights day 4|    Increase: 66.67 %       Decrease: 33.33 %
Weights day 5|    Increase: 69.37 %       Decrease: 30.63 %
Weights day 6|    Increase: 70.27 %       Decrease: 29.73 %
Weights day 7|    Increase: 72.07 %       Decrease: 27.93 %
Weights day 8|    Increase: 70.27 %       Decrease: 29.73 %
Weights day 9|    Increase: 71.17 %       Decrease: 28.83 %

Weights|          Increase: 51.24 %       Decrease: 48.76 %
```

```
[ ]: dft = df["Adj Close"].pct_change(n).dropna()

     # Compute %Increase %Decrease of the asset on all the situations
     p_increase = len(dft[dft>0])/len(dft)*100
     p_decrease = len(dft[dft<0])/len(dft)*100
     print(f"Weights| \t Increase: {'%.2f' % p_increase} % \t Decrease: {'%.2f' % p_decrease} % ")
```

```
Weights|         Increase: 57.30 %       Decrease: 42.70 %
```

4 Hypothesis test (This part comes with chapter “Hypothesis test”. Don’t read it before if you are not confortable with them)

```
[ ]: import scipy
     scipy.__version__
```

```
[ ]: '1.7.1'
```

```
[ ]: from scipy.stats import ttest_ind

     # H0: RETURN_bollinger   RETURN_normal
     # H1: RETURN_bollinger > RETURN_normal
     # Alpha: 0.05=5%

     p = ttest_ind(dft, df_returns.iloc[:,-1].values,alternative="less").pvalue
     print(f"H0: Strategy returns   Random returns \t p_value: {np.round(p*100,2)} %")
```

```
H0: Strategy returns   Random returns    p_value: 0.8 %
```

We have a p value of ***0.8%***. It means that there is 0.8% chance that the mean of the strategy return is inferior or equal to the mean of return. So, we can reject H0.

```
[ ]:
```