# finance Documentation Release

G. Barone, J. Bilbao de Mendizabal, A. Katre

#### CONTENTS

1	change_stock module	3
2	conf module	5
3	market module	7
4	newfinance module	9
5	stock module	13
6	test module	15
7	testVirtualMarket module	17
8	virtualMarket module	19
9	Indices and tables	21
Рy	thon Module Index	23
In	dex	25

Contents:

CONTENTS 1

2 CONTENTS

CHAPTER	
ONE	

# CHANGE\_STOCK MODULE

СНАРТЕ	ER
TW	O'

# **CONF MODULE**

#### **CHAPTER**

## **THREE**

# **MARKET MODULE**

class market .market (cap)

Base for decribing stock evolution

 $m_cap = 0$ 

 $m\_time = 0$ 

#### **NEWFINANCE MODULE**

A collection of modules for collecting, analyzing and plotting financial data. User contributions welcome!

newfinance.candlestick (ax, quotes, width=0.2, colorup='k', colordown='r', alpha=1.0)

quotes is a sequence of (time, open, close, high, low, ...) sequences. As long as the first 5 elements are these values, the record can be as long as you want (eg it may store volume).

time must be in float days format - see date2num

Plot the time, open, close, high, low as a vertical line ranging from low to high. Use a rectangular bar to represent the open-close span. If close >= open, use colorup to color the bar, otherwise use colordown

ax: an Axes instance to plot to width: fraction of a day for the rectangle width colorup: the color of the rectangle where close >= open colordown: the color of the rectangle where close < open alpha: the rectangle alpha level

return value is lines, patches where lines is a list of lines added and patches is a list of the rectangle patches added

newfinance.candlestick2 (ax, opens, closes, highs, lows, width=4, colorup='k', colordown='r', alpha=0.75)

Represent the open, close as a bar line and high low range as a vertical line.

ax : an Axes instance to plot to width : the bar width in points colorup : the color of the lines where close >= open colordown : the color of the lines where close < open alpha : bar transparency

return value is lineCollection, barCollection

newfinance.fetch\_historical\_yahoo(ticker, date1, date2, cachename=None, dividends=False)

Fetch historical data for ticker between date1 and date2. date1 and date2 are date or datetime instances, or (year, month, day) sequences.

Ex: fh = fetch\_historical\_yahoo('^GSPC', (2000, 1, 1), (2001, 12, 31))

cachename is the name of the local file cache. If None, will default to the md5 hash or the url (which incorporates the ticker and date range)

set dividends=True to return dividends instead of price data. With this option set, parse functions will not work

a file handle is returned

newfinance.index\_bar (ax, vals, facecolor='b', edgecolor='l', width=4, alpha=1.0) Add a bar collection graph with height vals (-1 is missing).

ax : an Axes instance to plot to width : the bar width in points alpha : bar transparency

newfinance.parse\_yahoo\_historical (fh, adjusted=True, asobject=False)

Parse the historical data in file handle fh from yahoo finance.

adjusted If True (default) replace open, close, high, and low prices with their adjusted values. The adjustment is by a scale factor, S = adjusted close/close. Adjusted prices are actual prices multiplied by S.

Volume is not adjusted as it is already backward split adjusted by Yahoo. If you want to compute dollars traded, multiply volume by the adjusted close, regardless of whether you choose adjusted = TruelFalse.

asobject If False (default for compatibility with earlier versions) return a list of tuples containing

d, open, close, high, low, volume

If None (preferred alternative to False), return a 2-D ndarray corresponding to the list of tuples.

Otherwise return a numpy recarray with

date, year, month, day, d, open, close, high, low, volume, adjusted\_close

where d is a floating poing representation of date, as returned by date2num, and date is a python standard library datetime.date instance.

The name of this kwarg is a historical artifact. Formerly, True returned a cbook Bunch holding 1-D ndarrays. The behavior of a numpy recarray is very similar to the Bunch.

```
newfinance.plot_day_summary (ax, quotes, ticksize=3, colorup='k', colordown='r') quotes is a sequence of (time, open, close, high, low, ...) sequences
```

Represent the time, open, close, high, low as a vertical line ranging from low to high. The left tick is the open and the right tick is the close.

time must be in float date format - see date2num

ax : an Axes instance to plot to ticksize : open/close tick marker in points colorup : the color of the lines where close >= open colordown : the color of the lines where close < open return value is a list of lines added

```
newfinance.plot_day_summary2 (ax, opens, closes, highs, lows, ticksize=4, colorup='k', color-down='r')
```

Represent the time, open, close, high, low as a vertical line ranging from low to high. The left tick is the open and the right tick is the close.

ax : an Axes instance to plot to ticksize : size of open and close ticks in points colorup : the color of the lines where close >= open colordown : the color of the lines where close < open

return value is a list of lines added

```
newfinance.quotes_historical_yahoo(ticker, date1, date2, asobject=False, adjusted=True, cachename=None)
```

Get historical data for ticker between date1 and date2. date1 and date2 are datetime instances or (year, month, day) sequences.

See parse\_yahoo\_historical() for explanation of output formats and the asobject and adjusted kwargs.

Ex: sp = f.quotes\_historical\_yahoo('^GSPC', d1, d2,

```
asobject=True, adjusted=True)
```

returns = (sp.open[1:] - sp.open[:-1])/sp.open[1:] [n,bins,patches] = hist(returns, 100) mu = mean(returns) sigma = std(returns) x = normpdf(bins, mu, sigma) plot(bins, x, color='red', lw=2)

cachename is the name of the local file cache. If None, will default to the md5 hash or the url (which incorporates the ticker and date range)

```
newfinance.volume_overlay (ax, opens, closes, volumes, colorup='k', colordown='r', width=4, al-
pha=1.0)
```

Add a volume overlay to the current axes. The opens and closes are used to determine the color of the bar. -1 is missing. If a value is missing on one it must be missing on all

ax : an Axes instance to plot to width : the bar width in points colorup : the color of the lines where close >= open colordown : the color of the lines where close < open alpha : bar transparency

newfinance.volume\_overlay2 (ax, closes, volumes, colorup='k', colordown='r', width=4, al-pha=1.0)

Add a volume overlay to the current axes. The closes are used to determine the color of the bar. -1 is missing. If a value is missing on one it must be missing on all

ax : an Axes instance to plot to width : the bar width in points colorup : the color of the lines where close >= open colordown : the color of the lines where close < open alpha : bar transparency

nb: first point is not displayed - it is used only for choosing the right color

newfinance.volume\_overlay3 (ax, quotes, colorup='k', colordown='r', width=4, alpha=1.0)

Add a volume overlay to the current axes. quotes is a list of (d, open, close, high, low, volume) and close-open is used to determine the color of the bar

kwarg width: the bar width in points colorup: the color of the lines where close1 >= close0 colordown: the color of the lines where close1 < close0 alpha: bar transparency

#### CHAPTER

## **FIVE**

## STOCK MODULE

```
class stock.stock(cap)
    class for decribing stock evolution
    addHistoricaldata(currentVal=0, time=0)
    bet (betVal)
    evolve(change=0, use=False, timeLow=0, timeUp=1)
    getAll()
    getCap()
    getTime()
    getTimes()
    m_cap = 0
    m_iters = 0
    m_time = 0
    m_time_his = []
    m_val = []
    next()
```

CHAPTER	
SIX	

# **TEST MODULE**

CHAPTER
SEVEN

# **TESTVIRTUALMARKET MODULE**

#### VIRTUALMARKET MODULE

```
class virtualMarket .virtualMarket (nstocks, startingCapStock)
     Container Class of stocks
     AddStock (cap)
          Apend a stock with capital :param cap: intial capital
          Apend a stock to the container :param st: stock to be added
     Evolve()
          Calls the evolve method for each stock
     getStock(s)
          stock at position :param s: stock index :return: stock of index s
     listOllClosingValues()
          prints the list of final values of all stocks :returns: a print out
     m_allStocks = []
     m nstocks = 0
     m_overAllCapital = 0
     m_overAllVariation = 0
     m_startingCapStock = 0.0
     m_{time} = 100
     randomBet (low=0, up=100)
          Place a random order with in range low up on the stock :param low: lower limit :param up: upper limit
     setTime(time=100)
          Set time :param time: time to be set
```

## **CHAPTER**

## **NINE**

# **INDICES AND TABLES**

- genindex
- modindex
- search

#### PYTHON MODULE INDEX

```
C
change_stock, 3
conf, 5

m
market, 7

n
newfinance, 9
S
stock, 13
t
test, 15
testVirtualMarket, 17
V
virtualMarket, 19
```

finance	<b>Documentation</b>	า. Release
---------	----------------------	------------

24 Python Module Index

A	m_overAllCapital (virtualMarket.virtualMarket at		
addHistoricaldata() (stock.stock method), 13 AddStock() (virtualMarket.virtualMarket method), 19 AddStockC() (virtualMarket.virtualMarket method), 19	tribute), 19 m_overAllVariation (virtualMarket.virtualMarket at tribute), 19		
B bet() (stock.stock method), 13	m_startingCapStock (virtualMarket.virtualMarket attribute), 19 m_time (market.market attribute), 7		
C candlestick() (in module newfinance), 9 candlestick2() (in module newfinance), 9 change_stock (module), 3 conf (module), 5	m_time (stock.stock attribute), 13 m_time (virtualMarket.virtualMarket attribute), 19 m_time_his (stock.stock attribute), 13 m_val (stock.stock attribute), 13 market (class in market), 7 market (module), 7		
E	N		
evolve() (stock.stock method), 13 Evolve() (virtualMarket.virtualMarket method), 19	newfinance (module), 9 next() (stock.stock method), 13		
F	P		
fetch_historical_yahoo() (in module newfinance), 9	parse_yahoo_historical() (in module newfinance), 9 plot_day_summary() (in module newfinance), 10		
G	plot_day_summary2() (in module newfinance), 10		
getAll() (stock.stock method), 13 getCap() (stock.stock method), 13 getStock() (virtualMarket.virtualMarket method), 19 getTime() (stock.stock method), 13 getTimes() (stock.stock method), 13	Q quotes_historical_yahoo() (in module newfinance), 10 R randomBet() (virtualMarket.virtualMarket method), 19		
Í	S		
index_bar() (in module newfinance), 9  L listOllClosingValues() (virtualMarket.virtualMarket method), 19	setTime() (virtualMarket.virtualMarket method), 19 stock (class in stock), 13 stock (module), 13		
M	test (module), 15		
m_allStocks (virtualMarket.virtualMarket attribute), 19	testVirtualMarket (module), 17		
m_cap (market.market attribute), 7	V		
m_cap (stock.stock attribute), 13 m_iters (stock.stock attribute), 13 m_nstocks (virtualMarket.virtualMarket attribute), 19	virtualMarket (class in virtualMarket), 19 virtualMarket (module), 19 volume_overlay() (in module newfinance), 10		

#### finance Documentation, Release

volume\_overlay2() (in module newfinance), 11 volume\_overlay3() (in module newfinance), 11

26 Index