メディアプログラミング入門

第2回：時系列データ解析

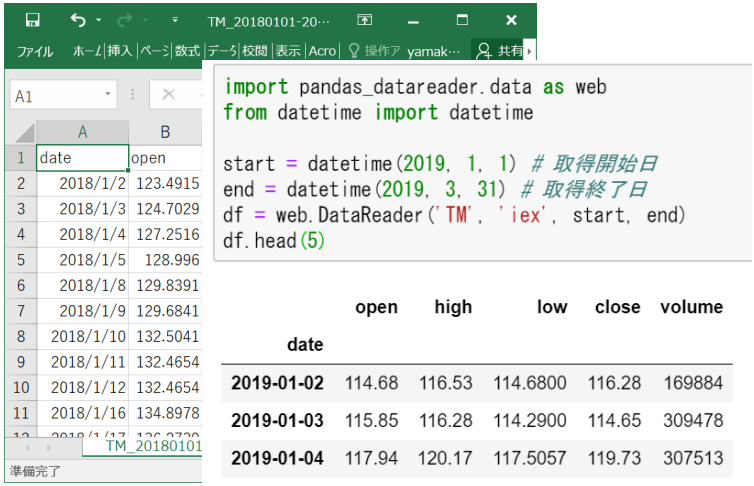
火５@本郷 2019年6月18日

情報理工学系研究科 数理・情報教育研究センター
准教授 山肩 洋子

第2回 データ解析：株価の変動を可視化してみよう

講義内容：実際の株式データ（個別銘柄の各日付における始値、高値、安値、終値、出来高、終値調整等）を使って、データの可視化を体験する

演習内容： データ解析ライブラリpandasと、グラフ描画ライブラリMatplotlibを利用しグラフを作成（タイトルや軸ラベルの描画、ローソク足、移動平均・移動平均乖離率、出来高等の描画、複数のグラフの配置）、機械学習による株価予測（ランダムフォレスト）




```
import pandas_datareader.data as web
from datetime import datetime

start = datetime(2019, 1, 1) # 取得開始日
end = datetime(2019, 3, 31) # 取得終了日
df = web.DataReader('TM', 'iex', start, end)
df.head(5)
```

date	open	high	low	close	volume
2019-01-02	114.68	116.53	114.6800	116.28	169884
2019-01-03	115.85	116.28	114.2900	114.65	309478
2019-01-04	117.94	120.17	117.5057	119.73	307513
2019-01-07	120.71	122.18	120.3200	121.28	156119
2019-01-08	123.16	123.48	121.9600	122.31	150370

実際の株式データはAPIによって直接ダウンロードしたり、Webよりcsv形式でダウンロードできる

このようなグラフを生成
銘柄や表示形式を変えてみよう！

前回の内容：実行環境の設定

課題：1/Preparation

- Python入門の実行環境のインストール
- モジュールアップデート
- 拡張モジュールのインストール
- ITC-LMSアンケートに回答

予習：2/TimeSeriesDataAnalysis[1-3].ipynb

- プログラムの実行（自分のPCにモジュールがインストールできなかった場合、またサンプルプログラムがエラーを出した場合は、ECCSのコンピュータで実行）
- 発展的演習はオプション

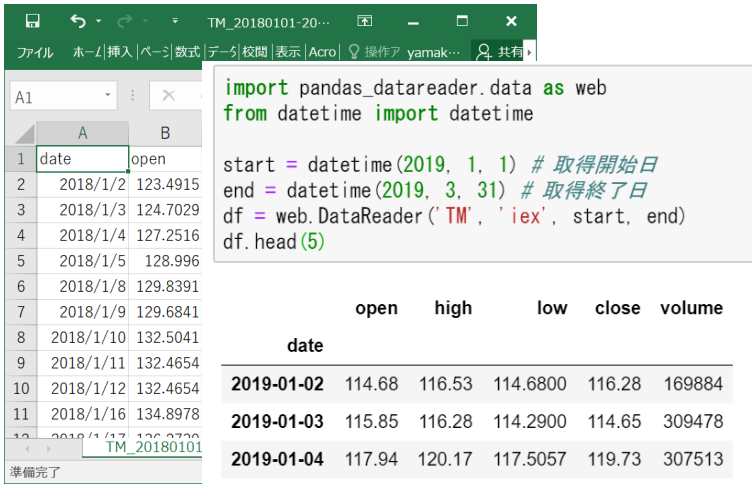
エラーとその解決策

- エラー報告があれば、ITC-LMSの掲示板やホームページでその解決策を掲示します！
  - エラーが起きたらとりあえずHPやITC-LMSをチェック！
  - 解決策が見つからなければあきらめずにITC-LMSの掲示板の質問スレッドかメッセージで質問してください（可能な限り一両日中に回答します）
- Preparation_YourOwnPC.pdfのなかで、condaのアップデートコマンドに誤植がありました
  - 旧　conda update –all
  - 新　conda update --all
- 6/15以降、pandas-datareaderでIEXからデータが取得できなくなる不具合
  → 修正版のTimeSeriesDataAnalysis1.ipynbを提供
- condaとpipを併用することの危険に関する解説
- ECCSの利用案内（Preparation_ECCS.pdf)

第2回 データ解析１：株価の変動を可視化してみよう

講義内容：実際の株式データ（個別銘柄の各日付における始値、高値、安値、終値、出来高、終値調整等）を使って、データの可視化を体験する

演習内容： データ解析ライブラリpandasと、グラフ描画ライブラリMatplotlibを利用しグラフを作成（タイトルや軸ラベルの描画、ローソク足、移動平均・移動平均乖離率、出来高等の描画、複数のグラフの配置）、機械学習による株価予測（ランダムフォレスト）




```
import pandas_datareader.data as web
from datetime import datetime

start = datetime(2019, 1, 1) # 取得開始日
end = datetime(2019, 3, 31) # 取得終了日
df = web.DataReader('TM', 'iex', start, end)
df.head(5)
```

date	open	high	low	close	volume
2019-01-02	114.68	116.53	114.6800	116.28	169884
2019-01-03	115.85	116.28	114.2900	114.65	309478
2019-01-04	117.94	120.17	117.5057	119.73	307513
2019-01-07	120.71	122.18	120.3200	121.28	156119
2019-01-08	123.16	123.48	121.9600	122.31	150370

実際の株式データはAPIによって直接ダウンロードしたり、Webよりcsv形式でダウンロードできる

このようなグラフを生成
銘柄や表示形式を変えてみよう！

時系列データ

- 一連のデータで一つの事象を表す
- ある程度同じ時間間隔でデータが記録される
  - 固定間隔の場合はサンプリングレートとして指定（ex. 音、映像）
  - 幅が一定でない、あるいは欠落がある場合は、個々のデータに**タイムスタンプ**を付与（ex. 株価は週末や独立記念日、クリスマスなどで株式市場が休場のためサンプルがない）

今回の授業で使用するモジュール

始めからインストールされているモジュール

- **matplotlib**: グラフを描画するためのライブラリ
- **pandas**: データ解析を簡単にするライブラリ。表や時系列データの処理に適している

これらはそれぞれPythonプログラミング入門の教材5-3と7-1で扱っています。

Anaconda navigatorでインストール可能なモジュール

- **pandas-datareader**: Web上の様々なソースに簡単にアクセスすることを可能とするパッケージ

pipでインストールするモジュール

- **mpl-finance**: matplotlibで経済データを描画するためのライブラリ。かつてはmatplotlibのサブモジュールだった

WebAPI (Application Programming Interface)

- リモートからサーバにアクセスしてサービスを利用する仕組み
- サービスを提供しているサーバに、指定された仕様でリクエストを送ると、処理結果を返してくれる
  - 様々な企業が様々な機能のWebAPIを公開
  - Google, Microsoft, IBM Watson, Youtube, Twitter, Facebook, Amazon,…
  - 画像認識、音声認識、自然言語処理、機械翻訳、データ分析、機械学習…
  - WebAPIを活用することで、処理能力が低い安価なスマートフォンでも、画像認識のような高性能な処理を組み込んだアプリが実装できる
  - ビジネスモデルや動作事例も含めて**第7回で解説予定**

演習：TimeSeriesDataAnalysis1.ipynb > **2. 株価の取得と記録**

Pandas-datareader：WebAPIとして公開されている様々な統計データを取得

- 株式や投資信託、為替などの履歴を取得可能
- その他、OECDがOECD加盟国のGDPや雇用指数など各種統計情報を提供したり、World Bankが人口やCO2排出量等のデータを提供

リクエスト

例）「2019年1月1日から2019年9月31日までのトヨタの株価は？」

```
import pandas_datareader.data as web
from datetime import datetime

start = datetime(2019, 1, 1) # 取得開始日
end = datetime(2019, 3, 31) # 取得終了日
df = web.DataReader('TM', 'iex', start, end)
df.head(5)
```

6/15から認証方式に切り替え！

WebAPIを使用するプログラムを実行

インターネット

Stooq

ポーランドの株価や為替データリポジトリ

レスポンス: DataFrame形式

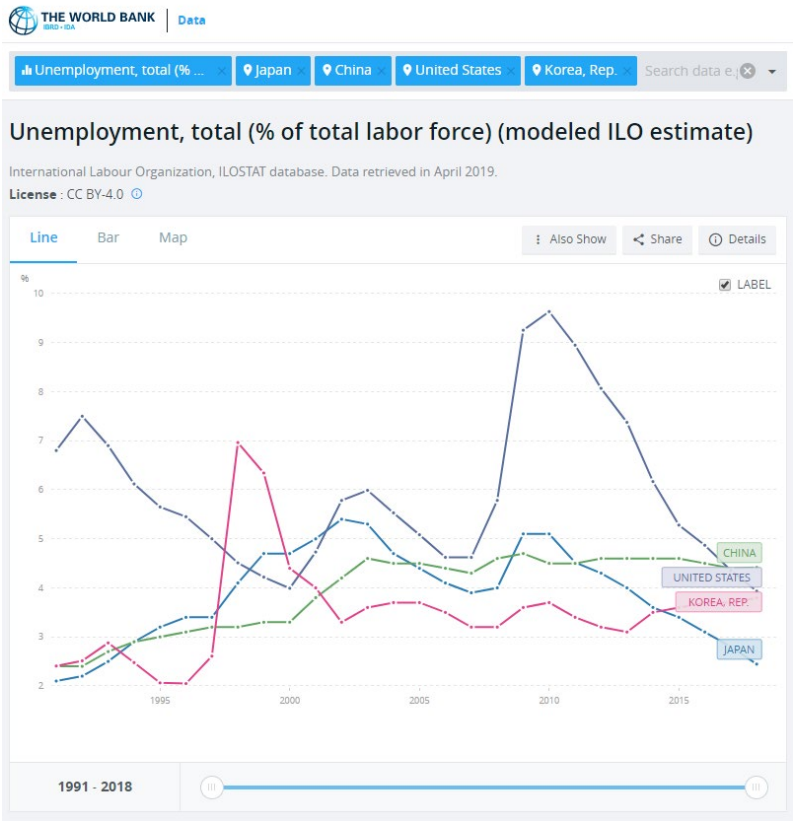
例）

date	open	high	low	close	volume
2019-01-02	114.68	116.53	114.6800	116.28	169884
2019-01-03	115.85	116.28	114.2900	114.65	309478
2019-01-04	117.94	120.17	117.5057	119.73	307513
2019-01-07	120.71	122.18	120.3200	121.28	156119
2019-01-08	123.16	123.48	121.9600	122.31	150370

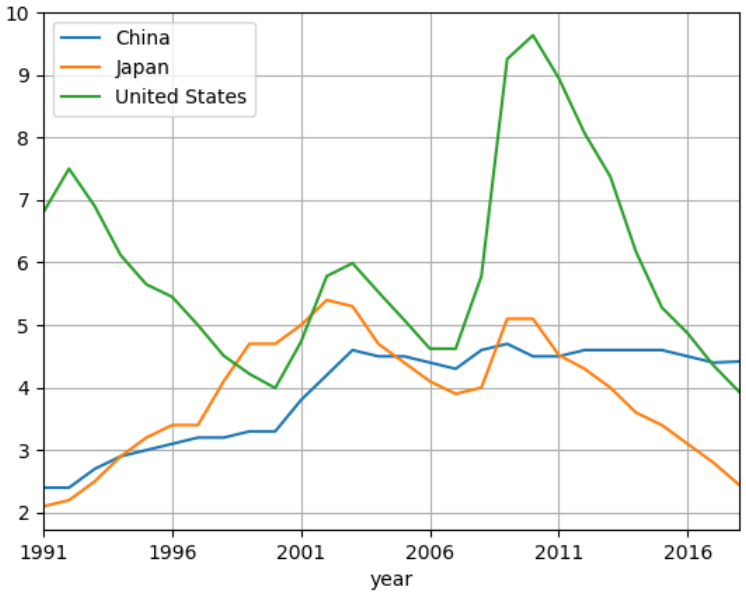
演習：TimeSeriesDataAnalysis1.ipynb > **2. 株価の取得と記録**

World Bank (世界銀行)

- サンプルプログラム：WorldBank.ipynb
- 貧困、経済、気候変動、保険、教育、ジェンダーなどの分野で、約8000の開発指標を無料公開



pandas-dataframeでダウンロードし、matplotlibでグラフ化



演習（Optinal）：WorldBank.ipynb

株価データの取得と
ローソク足チャートの描画

演習ファイル：TimeSeriesDataAnalysis1.ipynb

pandas-datareaderで利用可能なサービス

これらのサービスは変更する可能性があります。最新はこちらで確認

- Federal Reserve Economic Data (FRED)：米国経済（2万種類以上）
- Fama-French Data：ファーマ-フレンチの3ファクターモデル
- Bank of Canada：カナダ銀行
- Engima：世界最大の公開データリポジトリ
- Eurostat：欧州の統計データ
- **The Investors Exchange (IEX)：株価履歴 ←演習で使用**
- Moscow Exchange (MOEX)：モスクワ証券取引所
- Morningstar：株式、投資信託、ETF
- NASDAQ：米国のベンチャー向け株式市場
- OECD：OECD諸国の統計データ
- Quandl：株価、ETF等
- Robinhood：株価
- **Stooq.com：株価 ←日本の株価データも提供（サンプルプログラム：Stooq.ipynb）**
- Tiingo：株価、投資信託、ETF等
- Thrift Savings Plan (TSP)：投資信託
- **World Bank：約8000の開発指標 (サンプルプログラム：WorldBank.ipynb)**

演習：TimeSeriesDataAnalysis1.ipynb > **2. 株価の取得と記録**

演習１）TimeSeriesDataAnalysis1.ipynb

IEXからオンラインで指定銘柄の株価の変動履歴情報を取得し描画する

- DataReaderの第１引数である'TM'はTOYOTA MOTOR CORPのティッカーコード（銘柄コード）
- 取得できるのは以下の5種類
  - open : 始値
  - high : 高値
  - low : 安値
  - close : 終値
  - volume: 出来高
- このうちopen, high, low, closeの単位はドル。これら４つを合わせてOHLCという略称で呼ぶ。
- 出来高はその日売買が成立した株数

pandasにおけるIndexオブジェクト

- ローソク足チャートを簡単に描画できるmpl_financeとパッケージのcandlestick_ohlcという関数を利用
- この関数は、1行目から時間順に株価データが並んでおり、1列目に日付（ただし数値化されたもの）、2～4列目にOpen, High, Low, Closeが並んだ二次元配列を受け取る
- TimeSeriesDataAnalysis1.ipynb「4.1 前処理」を参照

- DataFrameのindexはタイムスタンプ（日付）
- これはstring型ではなく、datetimeという日付・時間型

date	open	high	low	close	volume
2019-01-02	114.68	116.53	114.6800	116.28	169884
2019-01-03	115.85	116.28	114.2900	114.65	309478
2019-01-04	117.94	120.17	117.5057	119.73	307513
2019-01-07	120.71	122.18	120.3200	121.28	156119
2019-01-08	123.16	123.48	121.9600	122.31	150370

演習：TimeSeriesDataAnalysis1.ipynb > **2. 株価の取得と記録**

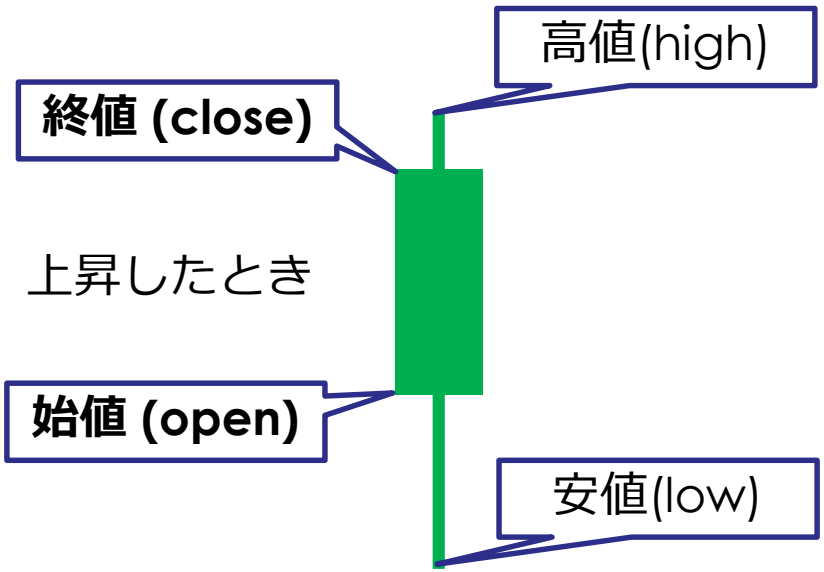
datetime：基本的な日付型および時間型

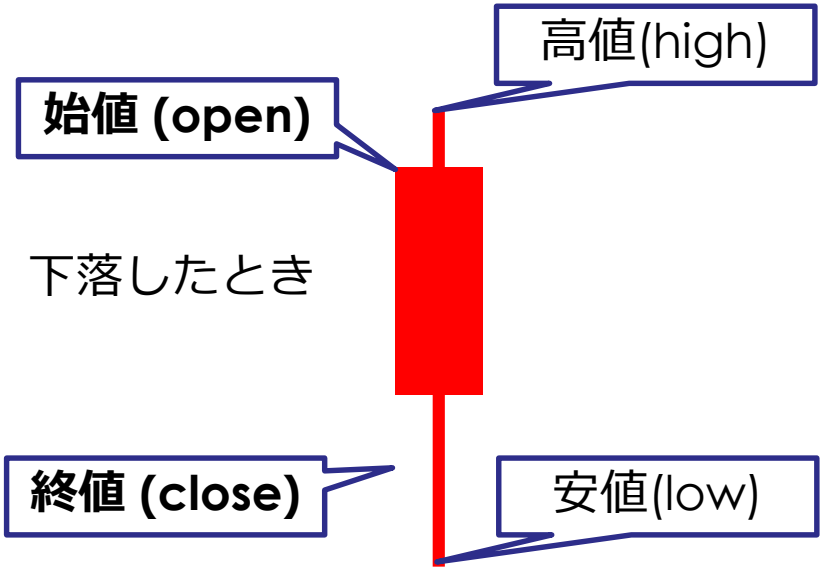
- 日付や時刻を対象にした**四則演算が可能**
  - 1日後や1週間後、1時間前を計算したり、「1990年3月15日から今日までの経過日数は？」などの計算も可能
  - 演習ファイル：Datetime.ipynb
- DataFrame型の時系列データをグラフ化する際、同じ間隔でサンプルされていないデータでも、datetime型のタイムスタンプがインデックスに設定されていれば、経過時間によってメモリの幅を自動調整

演習：TimeSeriesDataAnalysis1.ipynb > **2. 株価の取得と記録**

ローソク足チャート

- 株価などの相場の値動きを時系列に沿って図表として表す手法の一つ
- 単位期間中に初めに付いた始値、終値、高値、安値をローソクと呼ばれる一本の棒状の図形に作図し、時系列に沿って並べて値段の変動をグラフとして表したもの
- 上昇した時と下落した時で色が変わる





演習：TimeSeriesDataAnalysis1.ipynb > **4 ローソク足チャートを表示しよう**

mpl_financeによるローソク足チャートの描画

- ローソク足チャートを簡単に描画できるmpl_financeとパッケージのcandlestick_ohlcという関数を利用
- この関数は、1行目から時間順に株価データが並んでおり、1列目に日付（ただし数値化されたもの）、2～4列目にOpen, High, Low, Closeが並んだ二次元配列を受け取る
- TimeSeriesDataAnalysis1.ipynb「4.1 前処理」を参照

DataFrame型

date	open	high	low	close	volume
2019-01-02	114.68	116.53	114.6800	116.28	169884
2019-01-03	115.85	116.28	114.2900	114.65	309478
2019-01-04	117.94	120.17	117.5057	119.73	307513
2019-01-07	120.71	122.18	120.3200	121.28	156119
2019-01-08	123.16	123.48	121.9600	122.31	150370

matplotlib.dates.date2numにより
西暦1年1月1日からの経過日数に変換

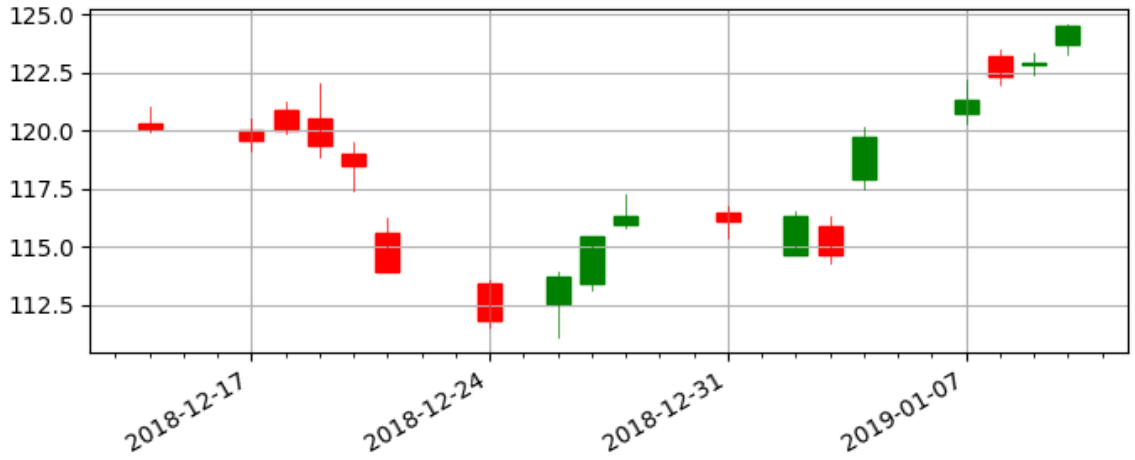
リスト型

```
[[7.370610e+05 1.146800e+02 1.165300e+02 1.146800e+02 1.162800e+02]
 [7.370620e+05 1.158500e+02 1.162800e+02 1.142900e+02 1.146500e+02]
 [7.370630e+05 1.179400e+02 1.201700e+02 1.175057e+02 1.197300e+02]
 [7.370660e+05 1.207100e+02 1.221800e+02 1.203200e+02 1.212800e+02]
 [7.370670e+05 1.231600e+02 1.234800e+02 1.219600e+02 1.223100e+02]]
```

演習：TimeSeriesDataAnalysis1.ipynb > **4 ローソク足チャートを表示しよう**

matplotlib.datesでロケータの設定

- matplotlibの軸で日付を扱うためのモジュール
- datetimeをインデックスに持つDataFrameをグラフ化
- **Locator**: 主軸は補助軸の**間隔**を設定する
  - set_major_locator: 主軸の間隔
  - set_minor_locator: 補助軸の間隔
- **formatter**: **軸ラベル**の出力形式を指定する
  - set_major_formatter: 主軸のラベル
  - set_minor_formatter: 補助軸のラベル（ないことが多い）



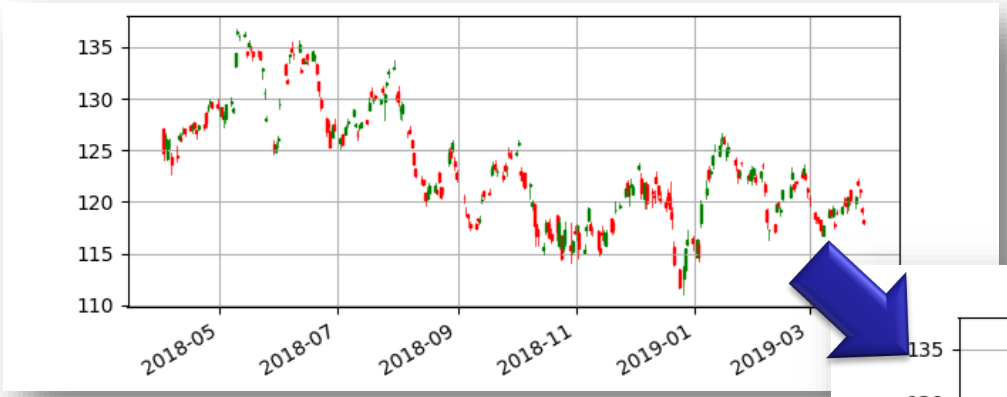
- 主軸は毎週月曜日ごと
- 補助軸は日ごと
- ラベルは自動設定

演習：TimeSeriesDataAnalysis1.ipynb > **4 ローソク足チャートを表示しよう**

ダウンサンプリングと
移動窓

演習ファイル：TimeSeriesDataAnalysis2.ipynb

1年分の日足データのローソク足チャートを描画すると…



日足データのローソク足チャート

週足データのローソク足チャート

- 日足データでは細かすぎて見づらい…
- 日足を週足に変換する
  → サンプルを記録する頻度を減らす
- 時系列データに対してサンプルの記録頻度を減らすことを**ダウンサンプリング**と呼ぶ

演習：TimeSeriesDataAnalysis2.ipynb > **2. ダウンサンプリング**

n項移動平均（〇日移動平均）を追加しよう

ローソク足チャートのみ

移動平均グラフを追加

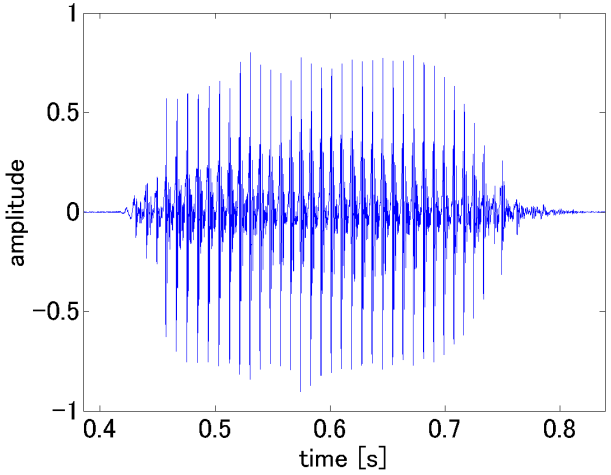
- 株価には長期的に見れば上がり傾向、下がり傾向がある（これを上昇トレンド、下降トレンドと呼ぶ）
- その日を含め、過去の一定期間（**移動窓**）の株価の平均を可視化するのが有効→これを**n項移動平均**と呼ぶ
  - n項には5日間、25日間、50日間などがよく使われるらしい
  - 株式市場の休場日（すなわち株価データが欠落している日）は含まないのが一般的

演習：TimeSeriesDataAnalysis2.ipynb > **3. n項移動平均**

音データに対するダウンサンプリングと移動窓

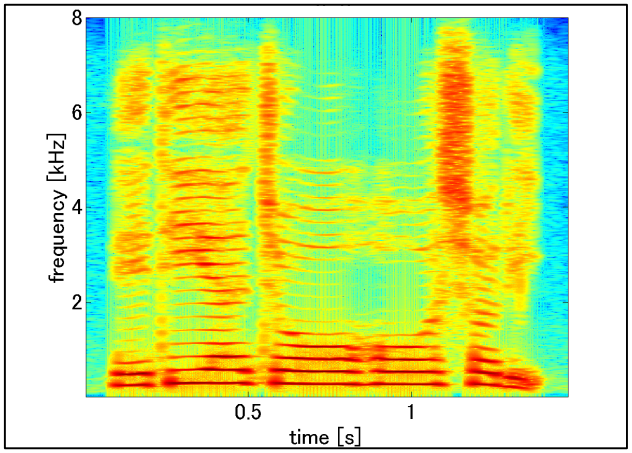
来週は音信号処理がテーマ

- 音のダウンサンプリング
  - データサイズを小さくしたいときに行われる
    例）CDの音 (44.1kHz)を16kHzのmp3に圧縮
  - 音質は低下する（高周波成分が欠落、雑音が乗る）
- 音の移動窓
  - スペクトログラムを計算する際に使う



音声波形

rollingしたあと
区間ごとに
周波数変換



スペクトログラム

時系列データに対するresampleとrolling

時系列データであるDataFrameあるいはSeriesが対象

- **resample**: 時系列を指定した区間ごとにグルーピングして処理を行う（**ダウンサンプリング**）

  演習）日足データから週足データを生成

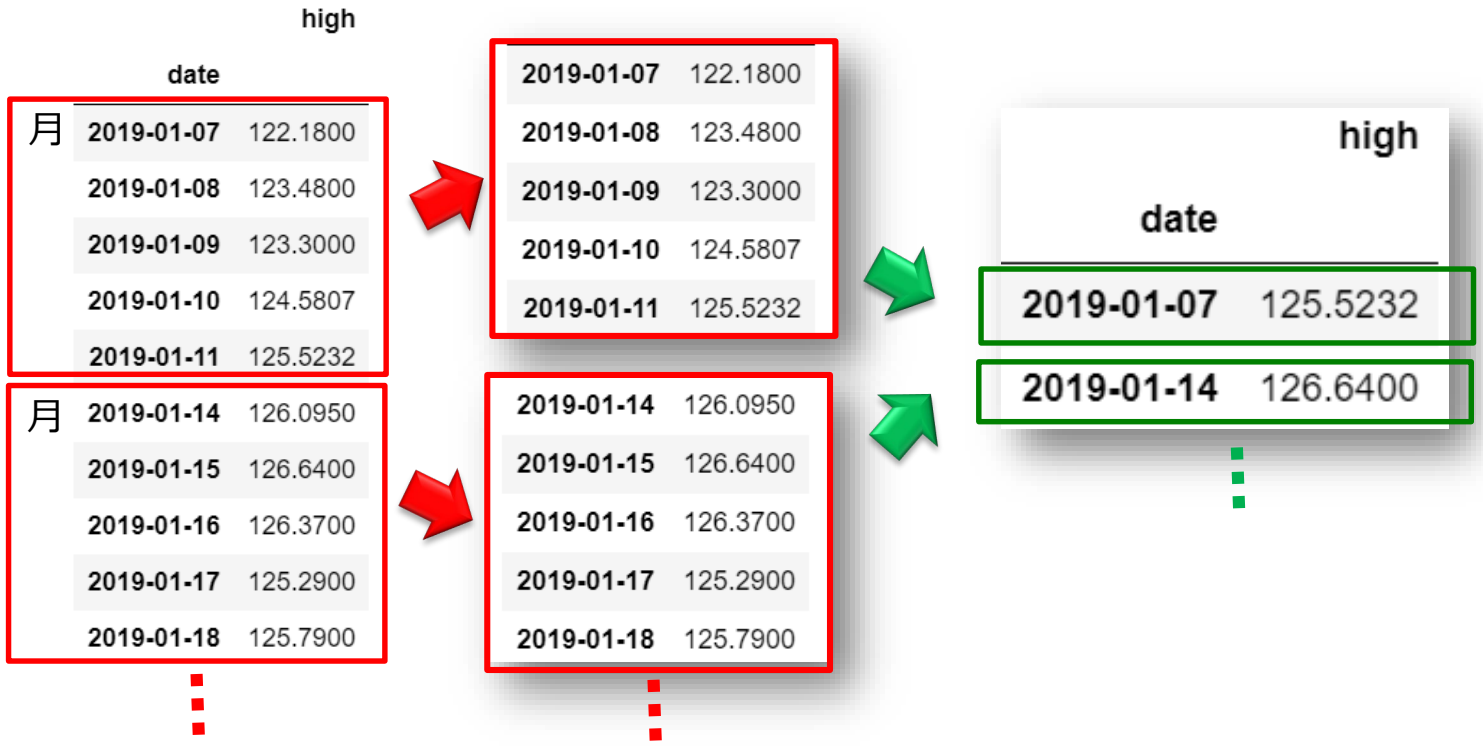
- **rolling**: 時系列の各点に対し、それ以前あるいは前後の指定した区間を切り出して処理を行う
  （**移動窓: moving window**）

  演習）n項移動平均を生成

日足データから週足データを生成

1. 月曜から日曜までの日足データを切り出す
2. 始値(open)は初日の始値 (first)、終値(close)は最終日の終値 (last)、
   高値(high)は最大値 (max)、安値(low)は最小値 (min)をそれぞれ抽出

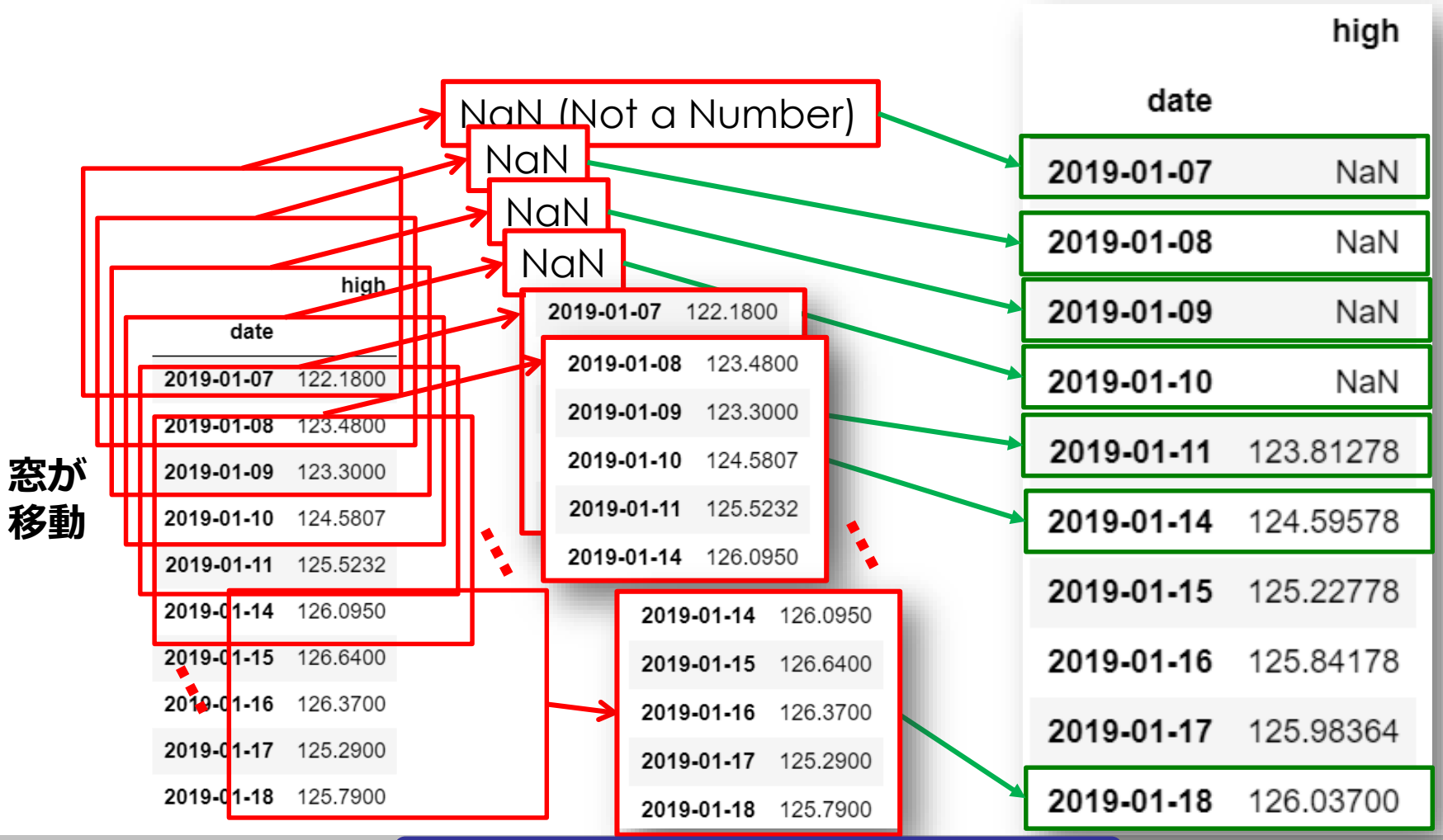
```
resample('W-MON', closed='left', label='left')
        .agg({'open': 'first', 'high': 'max', 'low': 'min', 'close': 'last', 'volume': 'sum'})
```



演習：TimeSeriesDataAnalysis2.ipynb > **2. ダウンサンプリング**

5日移動平均を生成

1. 各日ごとに、その日を含んでさかのぼって5日間を切り出し
2. 終値 (close)の平均値(mean)を抽出



演習：TimeSeriesDataAnalysis2.ipynb > **3. n項移動平均**

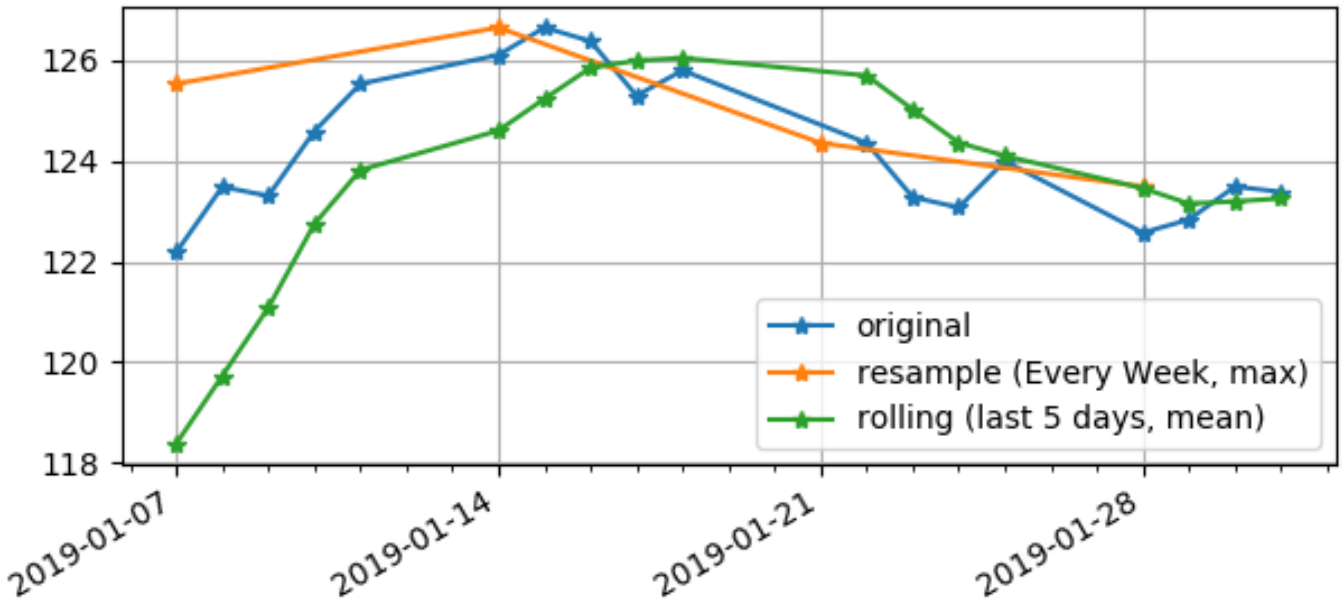
時系列データに対するResampleとRolling

- **resample**:
  時系列を指定した区間ごとに切り分けて処理（**ダウンサンプリング**）

  resample('W-MON', closed='left', label='left').agg({'high': 'max'})

- **rolling**: 時系列の各点に対し、それ以前の指定した区間に対して処理（**移動窓・窓関数**）

  rolling(5).mean()



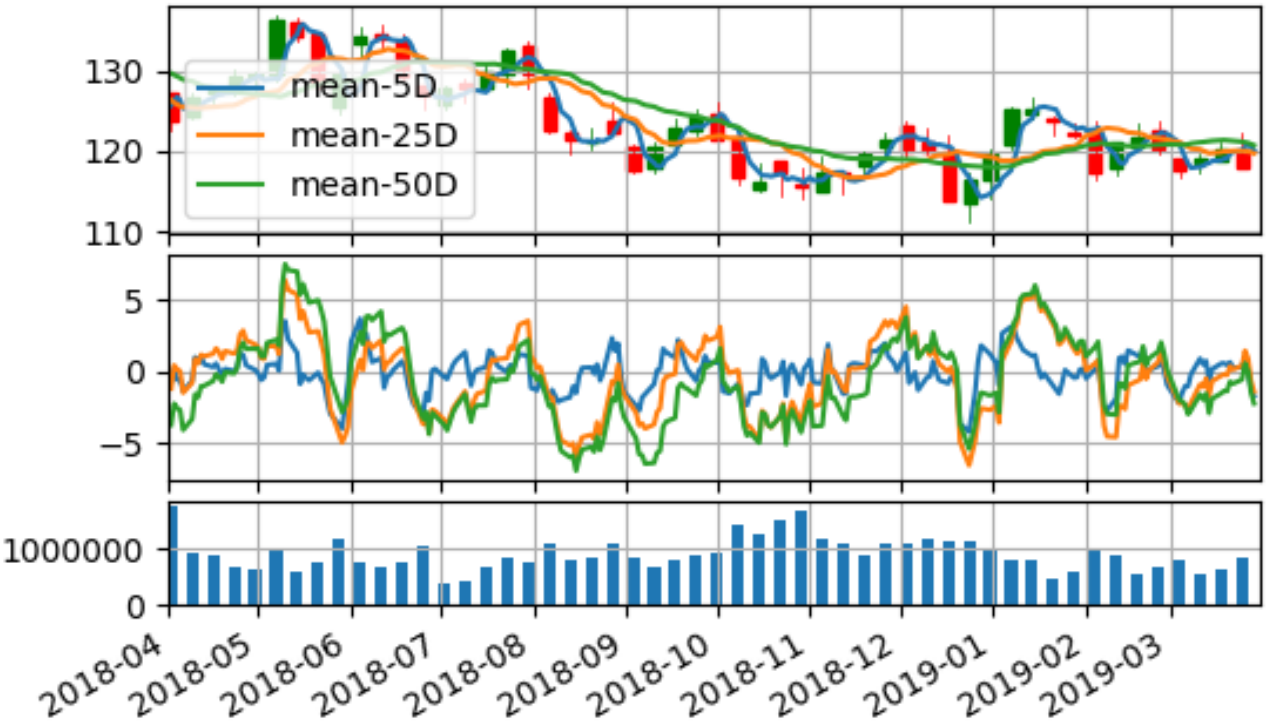
resampleとrollingに対する処理

関数	処理内容	resample	rolling
count	数を数える	○	○
sum	和	○	○
var	分散	○	○
mean	平均値	○	○
median	中央値	○	○
std	標準偏差	○	○
max	最大値	○	○
min	最小値	○	○
quantile	分位数・パーセンタイル	○	○
nunique	ユニークな要素の数	○	
first	最初の値	○	
last	最後の値	○	
ohlc	ohlcデータ	○	
prod	積	○	
size	サイズ	○	

関数	処理内容	resample	rolling
corr	相関係数		○
cov	畳み込み		○
skew	歪度		○
kurt	尖度		○
apply	関数を適応		○
aggregate	集計		○

matplotlibを使った複数グラフの配置

- 1枚のfigureに複数のグラフを配置
  - G=matplotlib.gridspec.GridSpec(6, 1）
    → 縦6×横1の領域を生成
  - ax0 = plt.subplot(G[:2, 0])
    → 縦3個分(0-2番目)、横1個分(0番目）の枠で一つ描画領域を生成



演習：TimeSeriesDataAnalysis2.ipynb > **4. グラフを並べて表示**

matplotlibでできること（一部）

- フォントのサイズを変える
  - 図全体を変える：plt.rcParams["font.size"] = 16
  - 個別に変える

    plt.title(“Title”, fontsize=20)、plt.xlabel(“xlabel”, fontsize=12)、plt.ylabel(“ylabel”, fontsize=12)、plt.legend(fontsize=16)、plt.tick_params(labelsize=10)など
- 折れ線グラフの色や書式を変える
  - plot()の中で以下のパラメータを設定

    線の太さ：linewidth, 線のスタイル：linestyle（‘solid’ (実線), ‘dashed’ (破線), ‘dashdot’ (破線&点線), ‘dotted’ (点線)), 色：color, マーカの種類：marker, マーカのサイズ：markersize, マーカの輪郭線の太さ：markeredgewidth、マーカの輪郭線の色：markeredgecolor, マーカの塗りつぶしの色：markerfacecolor、マーカの塗りつぶし方：fillstyle, アンチエイリアス（線を滑らかにする機能）
- 折れ線グラフの色や書式を変える
  - bar()の中で以下のパラメータを設定

    太さ:width、下側の余白：bottom、色：color、棒の輪郭の色：edgecolor、棒の輪郭の太さ: linewidth、yerr: y軸方向にエラーバーを追加、xerr: x軸方向にエラーバーを追加、ecolor: エラーバーの色、capsize：エラーバーの傘のサイズ

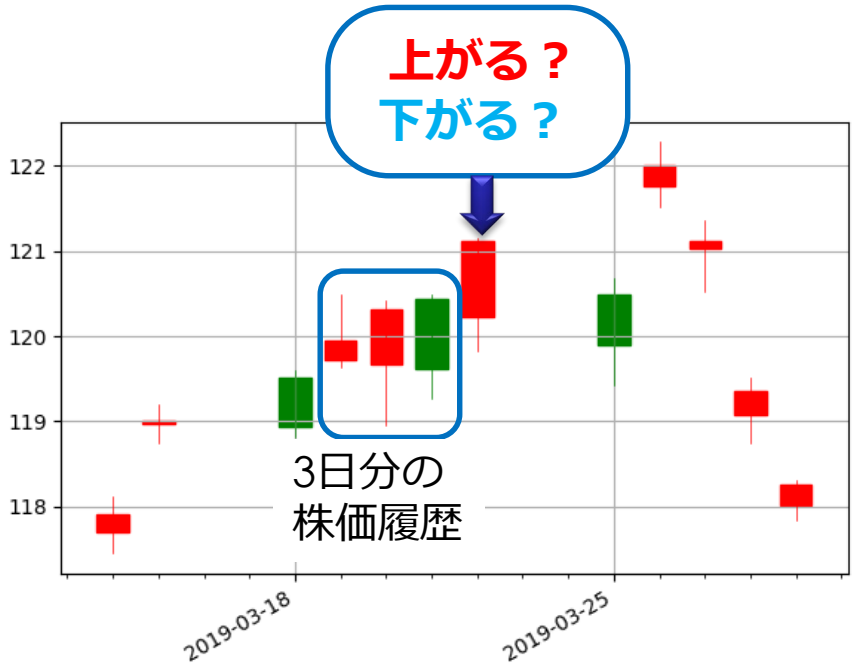
機械学習による株価の予測

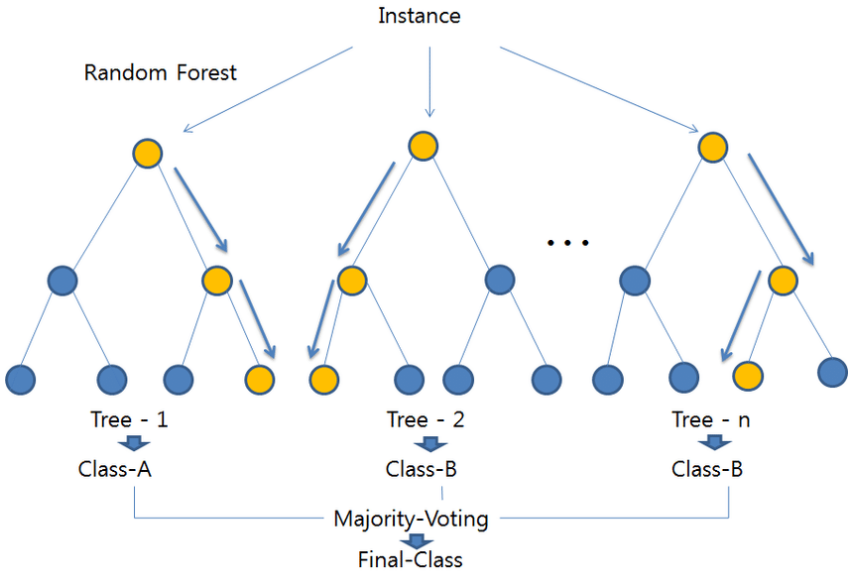
演習ファイル：TimeSeriesDataAnalysis3.ipynb

この授業では機械学習は詳しく扱いません。
アルゴリズムを詳しく学びたい方は
「データマイニング概論」 森純一郎先生（A1A2ターム）
などの受講をお勧めします。

機械学習で株価を予測しよう

- 今日・昨日・一昨日の株価履歴（始値、高値、安値、終値）から、明日の株価が上がる（始値より終値が高い）かを予測してみよう
- **演習内容：**Pythonの代表的な機械学習のライブラリである**scikit-learn**を使って、ランダムフォレストを用いた株価予測のプログラムを試行する。また、その精度評価及び予測においてどの情報が重要だったかも調べる。





- ランダムフォレストによる株価予測
- 説明変数の重要度を調べる

Fig ref.) Lee, et. al (2017). "Informal Quality Data Analysis via Sentimental analysis and Word2vec method. Journal of the Korean Society for Quality Management", 45, 117-128. 10.7469/JKSQM.2017.45.1.117.

明日の株価は上がりますか？下がりますか？

過去３日分の始値、高値、安値、終値のみから、
今日上がるか下がるかを予測してください！

- ３日連続上がったら上がるんじゃない？
  - (一昨日の終値が始値より高い and 昨日の終値が始値より高い and 昨日の終値が始値より高い) == True ならば上がる
- ここ二日、高値が終値よりもすごく高かったらそろそろ上がるかも？！
  - ((一昨日の高値)/(一昨日の終値) > Th1 and (今日の高値)/(今日の終値) > Th2) == True ならば上がる

ほかにも、今回はやりませんが。。。

- 今日発表された失業率が低かったから上がるんじゃない？
- アメリカの平均株価が上がったから上がるんじゃない？
- 年始だから上がるんじゃない？
- etc…

予測に役立つ変数＝説明変数

- その現象を説明するのに役立つ変数
  ＝これからその現象が起こるかどうかを予測するのに役立つ変数
- 先ほどの例では
  - 昨日と一昨日の(始値)/(終値)あるいは(高値)/(安値)
  - JASDAQ指数、日経平均株価、他銘柄の株価、為替
  - 世界情勢、政府報道（雇用統計等）、SNS（Twitter、facebook)
- 演習では、今日・昨日・一昨日の(始値)/(終値)、(高値)/(終値) 、(安値)/(終値)を説明変数とする

ある日のデータ=

[(一昨日の始値)/(一昨日の終値), (一昨日の始値)/(一昨日の終値), (一昨日の始値)/(一昨日の終値),
(昨日の始値)/(昨日の終値), (昨日の始値)/(昨日の終値), (昨日の始値)/(昨日の終値),
(今日の始値)/(今日の終値), (今日の始値)/(今日の終値), (今日の始値)/(今日の終値),]

のような９次元のベクトル

+ 上がった場合[１]、下がった場合 [0] のラベルを追加

演習：TimeSeriesDataAnalysis3.ipynb > **2. 前処理**

前処理により生成したデータ

演習ファイル： TimeSeriesDataAnalysis3.ipynb

明日の終値が始値より
高ければ１、
低ければ0

今日のデータ　昨日のデータ　一昨日のデータ

date	open	high	low	open-1	high-1	low-1	open-2	high-2	low-2	updown
2018-01-04	0.993568	1.000000	128.0754	0.988857	1.000811	126.1081	0.992677	1.000467	124.4025	1
2018-01-05	0.994397	1.000075	129.7228	0.993568	1.000000	128.0754	0.988857	1.000811	126.1081	1
2018-01-08	0.994138	1.000149	130.6047	0.994397	1.000075	129.7228	0.993568	1.000000	128.0754	0
2018-01-09	1.000749	1.000823	129.5871	0.994138	1.000149	130.6047	0.994397	1.000075	129.7228	0
2018-01-10	1.002419	1.006525	132.1843	1.000749	1.000823	129.5871	0.994138	1.000149	130.6047	1

9次元の説明変数から
updownの0/1を予想

このそれぞれがテストセット
- このスライドの例では5個のデータがある
- 演習では310個のデータが得られている

演習：TimeSeriesDataAnalysis3.ipynb > **2. 前処理**

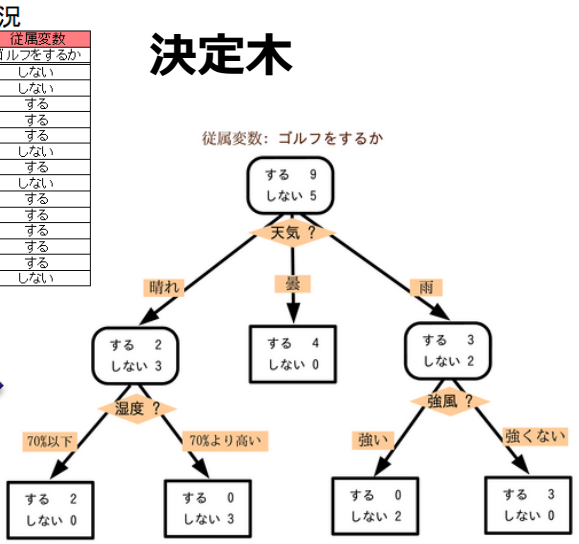
RandomForestによる二値分類

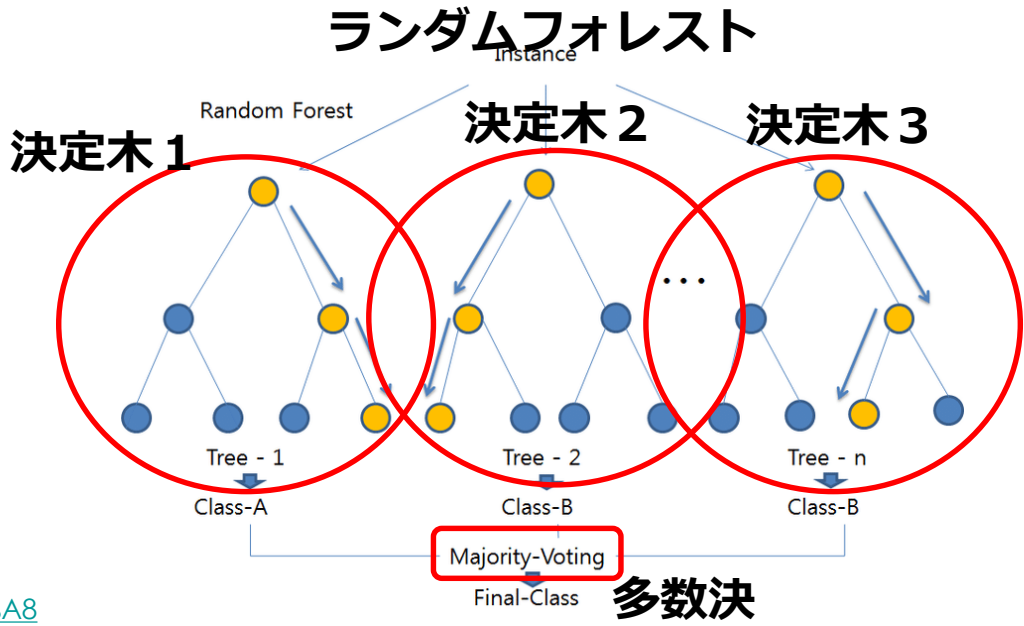
- 決定木：条件を満たすかどうかで木のように分岐させてTrueかFalseを判定する手法
- ランダムフォレスト：条件を変えて決定木をたくさん作って、それらの間で多数決で答えを決める
  - 学習データから、もう一段小さなサブセットを作る。このとき、そのデータの組み合わせをランダムに変えたり、組み合わせる説明変数をランダムに変える
  - そうして得られたたくさんのサブセットからしれじれ決定木を生成し、各自に判定させ、多数決で正解を決める

ゴルフクラブ来場状況

独立変数				従属変数
天気	気温(℃)	湿度(%)	風が強いか	ゴルフをするか
晴れ	29	85	強くない	しない
晴れ	27	90	強い	しない
曇	28	78	強くない	する
雨	21	96	強くない	する
雨	20	80	強くない	する
雨	18	70	強い	しない
曇	18	65	強い	する
晴れ	22	95	強くない	しない
晴れ	21	70	強くない	する
雨	24	80	強くない	する
晴れ	24	70	強い	する
曇	22	90	強い	する
曇	27	75	強くない	する
雨	22	80	強い	しない

学習





図はWikipedia「決定木」より引用：
https://ja.wikipedia.org/wiki/%E6%B1%BA%E5%AE%9A%E6%9C%A8

演習：TimeSeriesDataAnalysis3.ipynb > **４．学習**

精度評価における学習データと評価データ

- 精度評価：いくつ中いくつが当てられたか？（パーセント）
- 与えられたデータをすべて使って学習していいか？→ダメ！
  - **Closed test**:
    モデルを学習したデータと同じデータセットで精度を評価すること
    - 当然高い精度が出るが、未知のデータに対する予測性能を意味しない
    - ここで精度が低ければ見込みなしともいえる
  - **Open test**:
    モデルを学習するデータとは違うデータで評価する
    - 与えられたデータをあらかじめ**学習データ(training data)**と**評価データ(test data)**にわけ、学習データでモデル学習して評価データで評価
    - 未知のデータに対する予測性能を意味する
    - Closed testの精度は非常に高いが、Open testの精度が低いとき、過学習と呼ぶ
- 学習データと評価データをどう分けるか？
  - 学習データが小さければ小さいほどモデルの性能は下がる
  - 評価データが小さいと、異常値（Outlier)に引きずられる
  - 今回は学習データ：評価データ＝7:3で分割
  - 交差検定法を使うのが一般的
    （評価データを入れ替えて出した精度の平均値を最終的な精度とする）

演習：TimeSeriesDataAnalysis3.ipynb > **3. 学習データを分割**

TrueとFalse，　PositiveとNegative

- **判別モデルが予測したラベルの正負（Positive｜Negative)は真 (True) か偽 (False)か？(教材に誤りがありました！）**
  - 株価が上がり、判別モデルも上がる（Positive)と予想
    →**正**しいPositive (True Positive, TP) → 予測成功！
  - 株価が下がり、判別モデルも下がる（Negative)と予想
    →**正**しいNegative（True Netagive，TN)→ 予測成功！
  - 株価が上がったけど判別モデルは下がる（Negative)と予習した
    →**偽**のNegative (False Negative, TN) → 予測失敗
  - 株価が下がったけど判別モデルは上がる（Positive)と予習した
    →**偽**のPositive (False Positive, FP) → 予測失敗
- 正解率 (accuracy) = (TP+TN) / (TP + FP + TN + FN)
  - 演習のscoreは正解率の平均（mean accuracy)

余談）Trueだけが重要な課題もある→異なる評価手法を使用

例）集合写真に移っている人の顔領域を取り出そう！

- 見つけた領域のうちいくつが本当の顔領域か？：適合率(precision) = TP / (TP + FP)
- 何人中何人を見つけられたか: 再現率（recall) = TP / (TP + FN)
- F値 (F-measure) = 適合率と再現率の調和平均

演習：TimeSeriesDataAnalysis3.ipynb > **5. 評価**

予測性能の混同行列（Confusion matrix）

- 正しいラベルと判定されたラベルの関係を表す行列
  - 画像に写っている動物が「犬、猫、馬」を判別するとしたら「本当は犬で犬と判定された数」、「本当は犬なのに猫と判定された数」、「本当は犬なのに馬と判定された数」、「本当は猫なのに犬と判定された数」…というように、３ｘ３の行列でその数を並べたもの
- 今回は上がるか下がるかの2値なので、２×２の行列となる

		予測されたラベル	
		上がる	下がる
実際のラベル	上がる	TP	FN
	下がる	FP	TN

教材に誤りがありました！

		予測されたラベル	
		上がる	下がる
実際のラベル	上がる	36	15
	下がる	26	16

予測正解

機械学習では前処理が重要！

- 収集したデータに対し機械学習を適用する際は、その前処理が極めて重要（直感9割）
- 前処理のとは？
  - **データクリーニング**：記録に失敗していたり、明らかにおかしいデータを除去する。人手で判断しなければならないことも多い（手間を惜しむと精度が出ない…）
  - **ベクトル化**：文字列、頻度、順位、距離など様々な表現を数値に変換
    - 例えばその日よく検索された単語が株価の予想に有効だと考えたする
      → 文字列はそのままでは学習できない！特徴ベクトルに変換する必要がある
    - 文字列の最も簡単なベクトル表現はone-hot vector: その文書で使われ得る語彙が1万単語だとすると、1単語を1万次元のベクトルで表現して、そのうちその単語に相当するアドレスの値を1、残りを0として表現したもの
    - ものすごく疎（スパース）なデータになるので、圧縮する手法がいろいろ考えられている(word2vecによるwordembedding: 第5回で扱う予定)
  - **正規化・標準化**：各パラメータが取り得る値のスケールをそろえる
    これが精度に大きく影響を与えることが知られている
  - Etc.

機械学習による株価の予測：まとめ

- 予測精度はどうでしたか？
  - Random Forestではランダムに学習サブデータが生成されるため、計算するたびに答えが変わる
  - 大体0.5くらい？→今回の演習レベルではムリ！
  - 世の中には機械学習で株価を予測して設けている人もいる（らしい）
  - handcraftな説明変数も駆使すべし
- データを可視化することは非常に重要
  - データを沢山突っ込めばあとは学習機がなんとかしてくれる…は✕
  - 人間が見て区別できないものは機械でも区別できないことが多い（解けない問題を解かせてないか？）
  - 様々な場面で可視化して、よい説明変数を探り出す

課題：株価履歴データからグラフを描画するプログラムの作成

- ある銘柄の株価データが与えられると、下のようなデザインのグラフを生成するプログラムを作成してください。
- ただし、以下の条件を満たしてください
  - 'TM_20180101-20190331_stock.csv'を読み込んでください
  - 2019年1月1日～2019年1月31日のデータを描画
  - 日足データをそのまま描画（週足ではない）
  - 4日移動平均と13日移動平均を重畳描画し、その凡例をつける
  - X幅の主軸は週（毎週月曜）ごと、補助軸は日ごと
  - X軸のラベルの主軸は年月日、補助軸はなし
  - 出力した画像をmi_exp2_ans.pngという名前の画像として保存