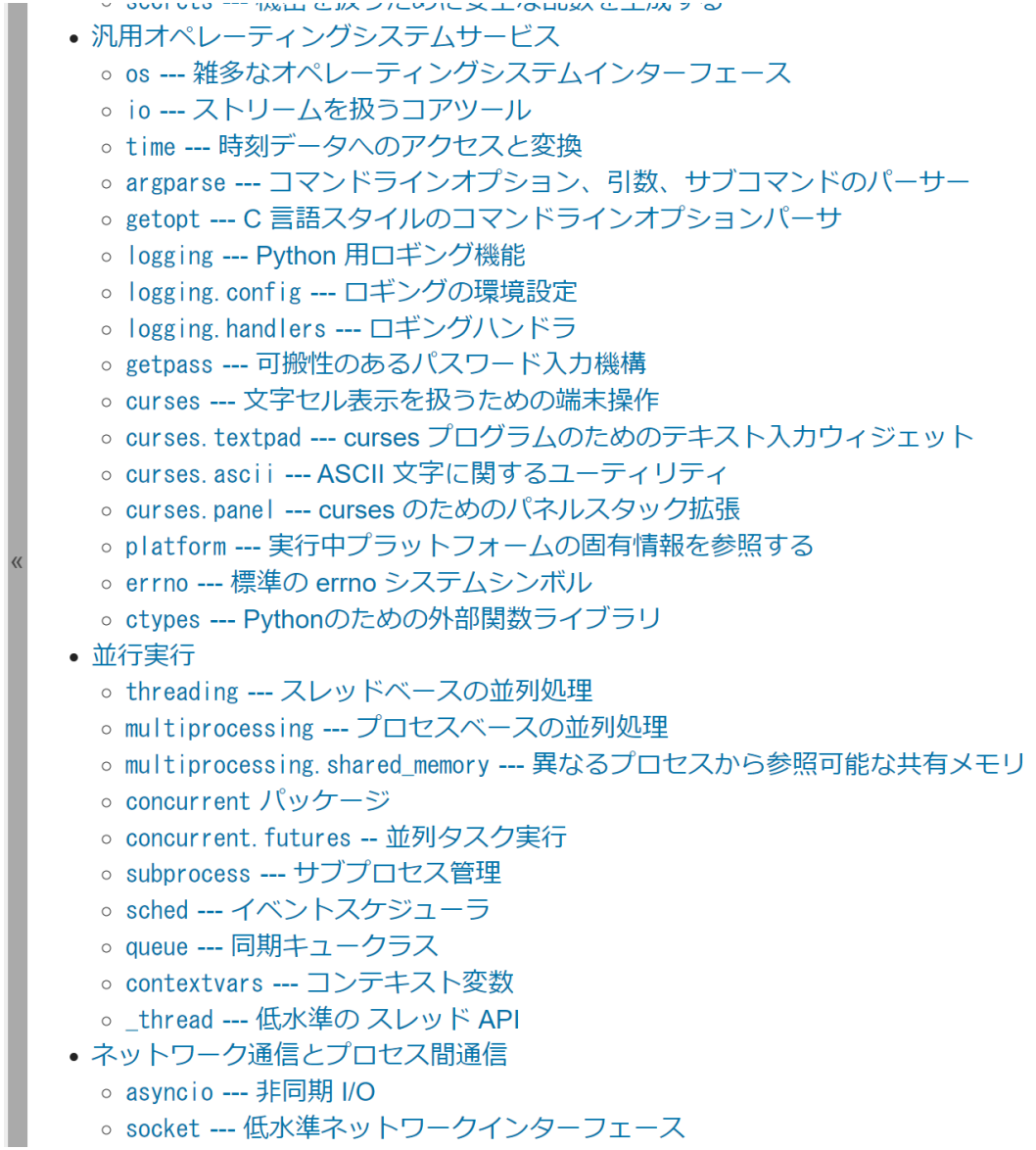
Pandas超速入門

20220526　セツ

1.
モジュール

- 汎用型プログラム言語としてpython自体は結構の数のモジュールを提供している
- 今日のスクリプトの中に
- Osモジュール以外は
- 原生モジュールではない
- つまり
- 前もってインストールする必要がある
- そのためのanaconda



- 汎用オペレーティングシステムサービス
  - os --- 雑多なオペレーティングシステムインターフェース
  - io --- ストリームを扱うコアツール
  - time --- 時刻データへのアクセスと変換
  - argparse --- コマンドラインオプション、引数、サブコマンドのパーサー
  - getopt --- C 言語スタイルのコマンドラインオプションパーサ
  - logging --- Python 用ロギング機能
  - logging.config --- ロギングの環境設定
  - logging.handlers --- ロギングハンドラ
  - getpass --- 可搬性のあるパスワード入力機構
  - curses --- 文字セル表示を扱うための端末操作
  - curses.textpad --- curses プログラムのためのテキスト入力ウィジェット
  - curses.ascii --- ASCII 文字に関するユーティリティ
  - curses.panel --- curses のためのパネルスタック拡張
  - platform --- 実行中プラットフォームの固有情報を参照する
  - errno --- 標準の errno システムシンボル
  - ctypes --- Pythonのための外部関数ライブラリ
- 並行実行
  - threading --- スレッドベースの並列処理
  - multiprocessing --- プロセスベースの並列処理
  - multiprocessing.shared_memory --- 異なるプロセスから参照可能な共有メモリ
  - concurrent パッケージ
  - concurrent.futures -- 並列タスク実行
  - subprocess --- サブプロセス管理
  - sched --- イベントスケジューラ
  - queue --- 同期キュークラス
  - contextvars --- コンテキスト変数
  - _thread --- 低水準の スレッド API
- ネットワーク通信とプロセス間通信
  - asyncio --- 非同期 I/O
  - socket --- 低水準ネットワークインターフェース

2.自分の作業フォルダーを

- デフォルトの設定なら
- Anacondaで起動するspyderの作業フォルダーは
- 自分のユーザーフォルダー
- データを管理するため
- 作業用のフォルダーを作るのがオススメ
- 作業フォルダーのチェンジは
- Osモジュールで可能



3. OSモジュールの常用コマンド

- 今の作業フォルダーを調べる：
- Os.getcwd()　　get current working directory
- 作業フォルダーの指定：
- Os.chdir(path)　　change directory
- ファイルやディレクトリの一覧
- Os.listdir()　　list directory
- ディレクトリを作成
- Os.mkdir()　　make directory

4.その.chdir()の　.　は何？

- Pythonとmethodで検索したら、分かりやすい説明がある
- ここの要約すると
- モジュールの中に既に用意された関数のこと
- をこのモジュール（クラスclass）のメソードと呼ぶ
- モジュールの中の関数を使いたい場合
- モジュール名.メソード()で大丈夫
- ()の中は関数の引数
- メソードを使うたびにモジュールの名前を書く必要がある

5.Import モジュール as 略語

- 今日のスクリプトの最初の数行で書いた通り
- 基本どこでも
- Pandasとnumpyが慣用の略語が存在する
- Import pandas as pd
- Import numpy as np

6.Pandasを使ってデータを読み込む

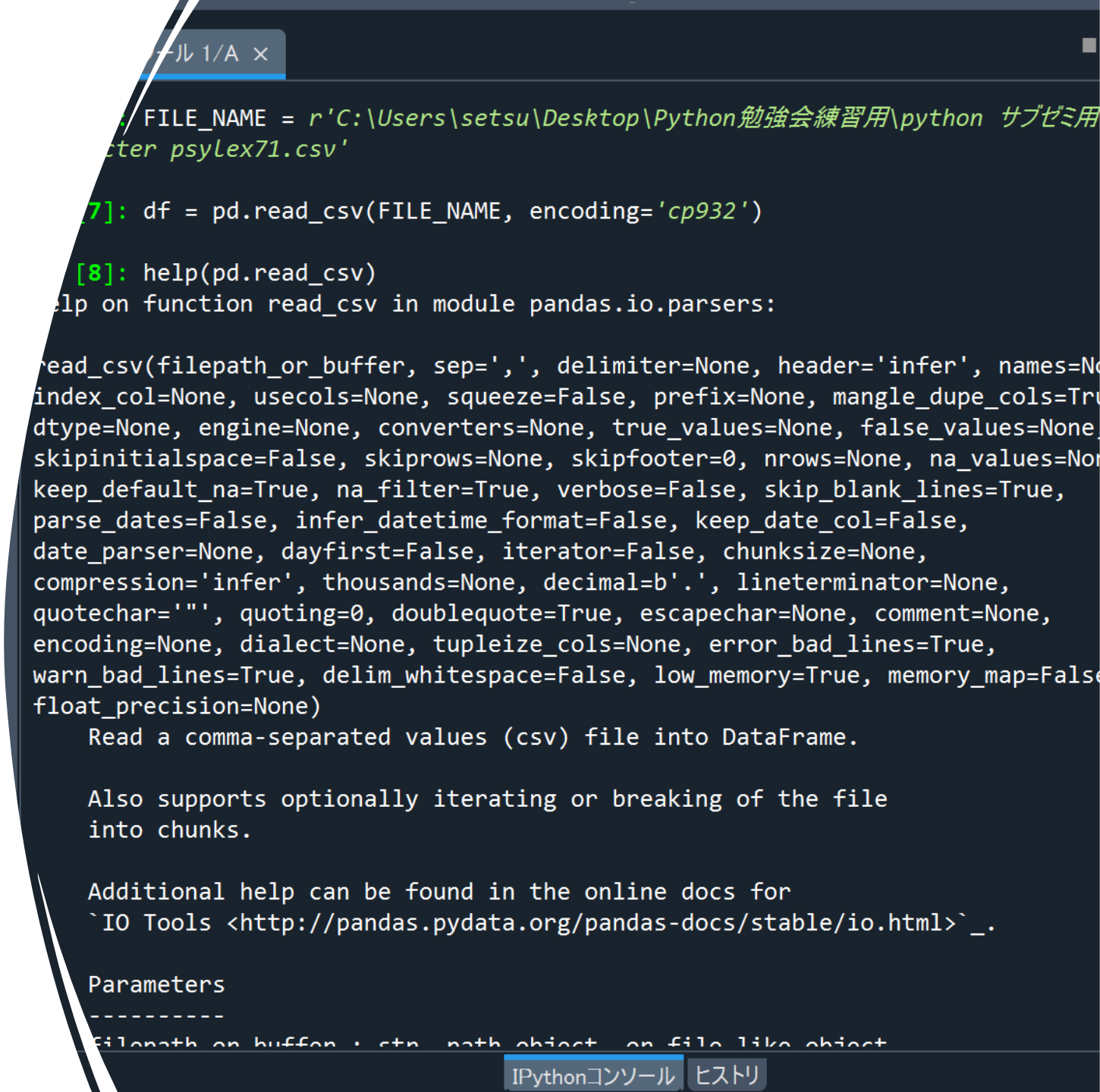
- 実はpandasなしでもpython単体でもファイルを読み込んだり、書き出したりすることができる。
- でもとても不便で、数回基本知識の勉強なしではできなPandasなら、一行ですぐできる
- もう一度定数と定数の命名規則を強調
- 変数と違って、定数とは、前もって決定されるもの
- 作業ダイレクトと読み込まれるファイルの名前など
- pd.read_csv(ファイルの名前,　encoding='文字コード')

6.1. よくある質問

- 文字コード：これは一番もどかしい
- 日本語関連なら、大体SHIFTーJSとUTF-8
- 特にUTF-8は日本語だけではなく、世界中の言語でも通用、またmacの標準で、とりあえず、UTF－８で試して、もし文字化けの場合CP932（SHIFTーJS）
- ファイルのheader
- つまり、データフレイムの列の名前の部分、デフォルトの場合、読み込んだファイルの第一行は自動的にheaderとして処理される
- もしファイルにはheaderの行がない場合。
- pd.read_csvの引数にheader=None

6.2 read_csvの他？

- CSVファイルの全称は？
- Comma Separated Value（＝カンマ（,）で区切った値）
- つまり、カンマではなくtabで区切ったファイルもある
- read_tableも使える、具体てきにはread_csvと同じ
- 引数の中にsepというものが必要
- つまり、sep=‘,’の場合、pd.read_talbeでもcsvファイルを読み込むことが可能
- 最後は引数がわからない場合
- Help(pd.read_csv)でやってみる

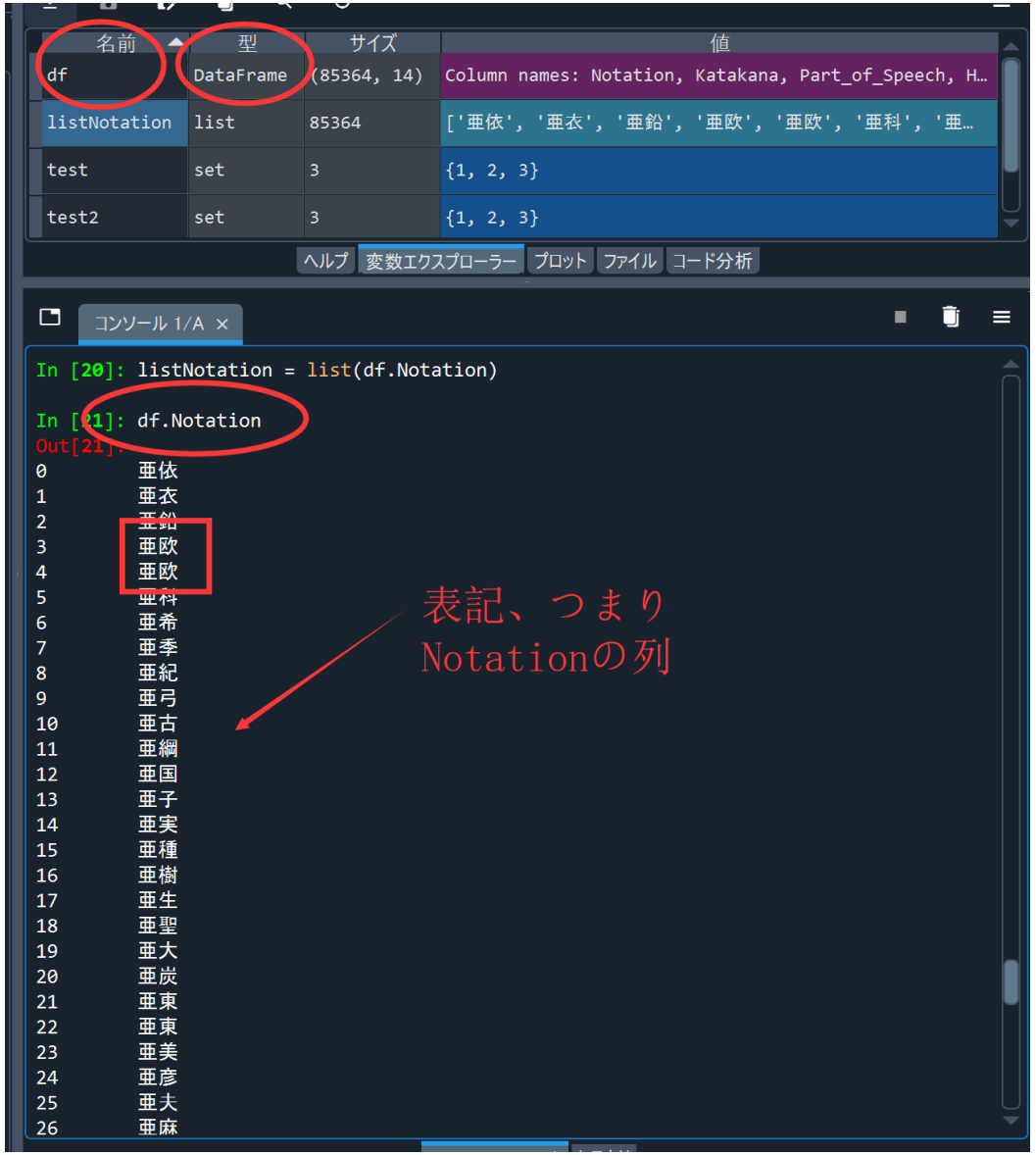


7.Pandasを使うための変数の型

- コンピュータで、プログラム内で値が変化することのできる数。名前を付けて識別される（変数名という）
- 例えば、その前紹介した　整数型INT　文字列STR
- 今は沢山の変数を一つに格納できる他の2つの型を紹介
- setとlist
- 基本的には同じが{}または[]でデータを囲むことで実現
- {1,2,3}は一つのセットで[1,2,3]はリスト
- 区別はsetの中身は変更できない、重複された要素はない

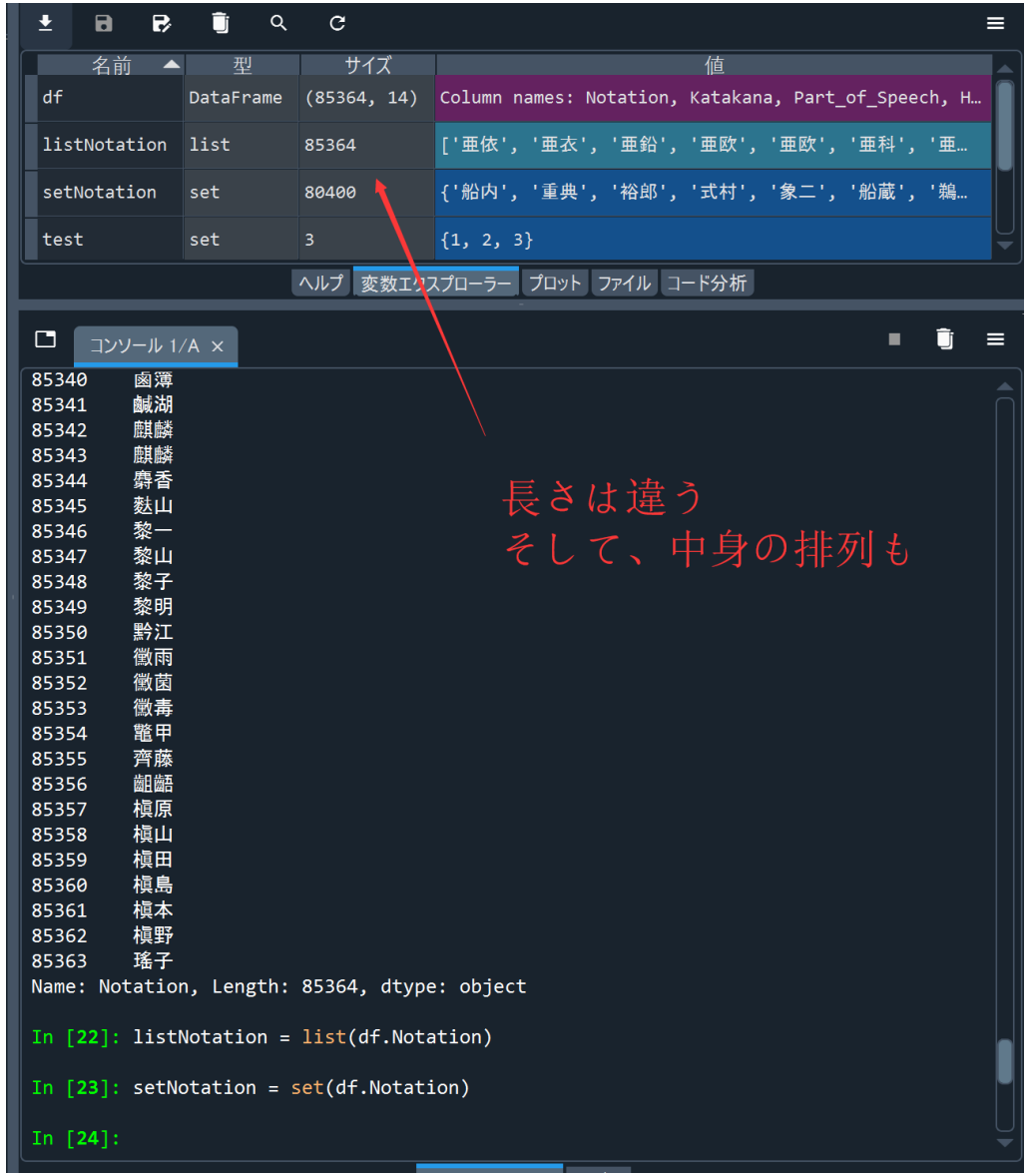
7.1 dataframeからデータを

- Pandasで読み込んだデータ形式はdataframeと呼ぶ
- 作業中頻繁にdataframeの名前を書くために、メインのdataframeの名前をなるべく短くにしたほうがいい
- 今のスクリプトの中にdfと名付ける
- そしてdf.列の名前で、相応の列を読み出すことが可能
- 前のメソードとは違うので、ご注意



7.1 setとlistの使い分け

- とりあえず、先のNotationの列をsetとlistに入れて見
- listNotation=list(df.Notation)
- setNotation=set(df.Notation)
- 一つの語彙を固有名詞と普通名詞両方とも収録されたケースがあって表記の重複がある
- Setは重複は許されてないので
- 重複する要素を除去することが可能



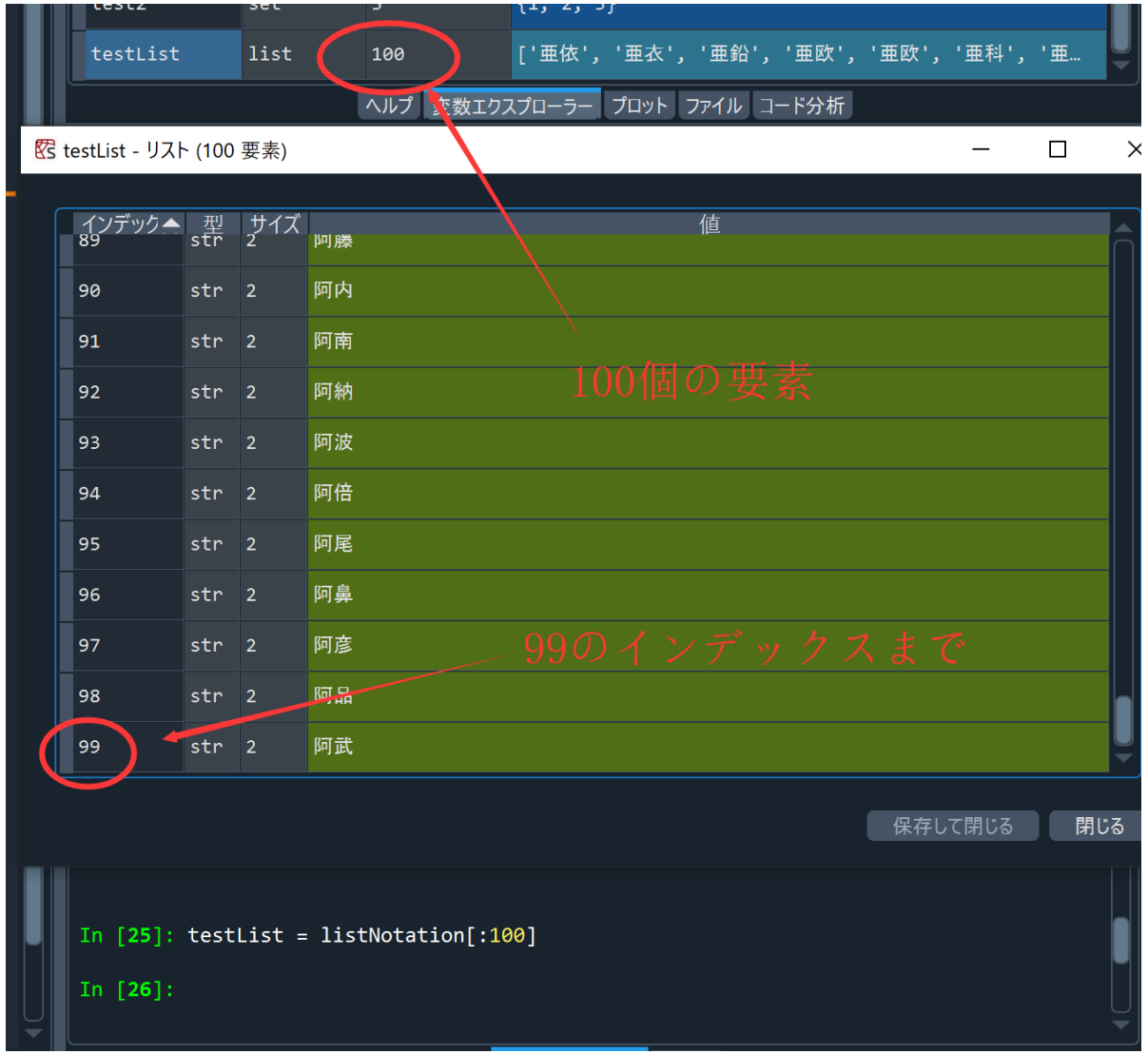
7.2 長過ぎるデータをスライス

- まずはインデックスの概念を紹介
- 文字列、リストなどのもの、対応する中身の位置
- そして
- 文字列、リストなどのものにたいして[]で抽出することが可能
- s = ‘Hellow World’
- s[1] = ??



[]は単なるの抽出だけではなく

- []によるスライス（切片）の操作
- [start:end:step]
- Start:抽出の起点、でも起点自身も抽出される対象である
- End:抽出の終点、startとは違って、終点自身は抽出の対象ではない
- Step:一つづつ抽出するか、二文字間隔で抽出も可能、更に負数も大丈夫だ。



7.3 []の[]も良くある操作

- リストは物入れだと思え
- その中身に対して更にスライスするのは可能？
- もちろん
- 先のtestListを例として三番目の「亜鉛」を
- 取り出して、1文字を取り出したり、逆転したりするのもok
- [：：ー１]とは逆転する操作
- 1ページ前のsetpのことを思い出せ！

```
In [28]: testFirstChar = testList[2][0]

In [29]: testFirstChar
Out[29]: '亜'

In [30]: testReversal = testList[2][::-1]

In [31]: testReversal
Out[31]: '鉛亜'

In [32]:
```

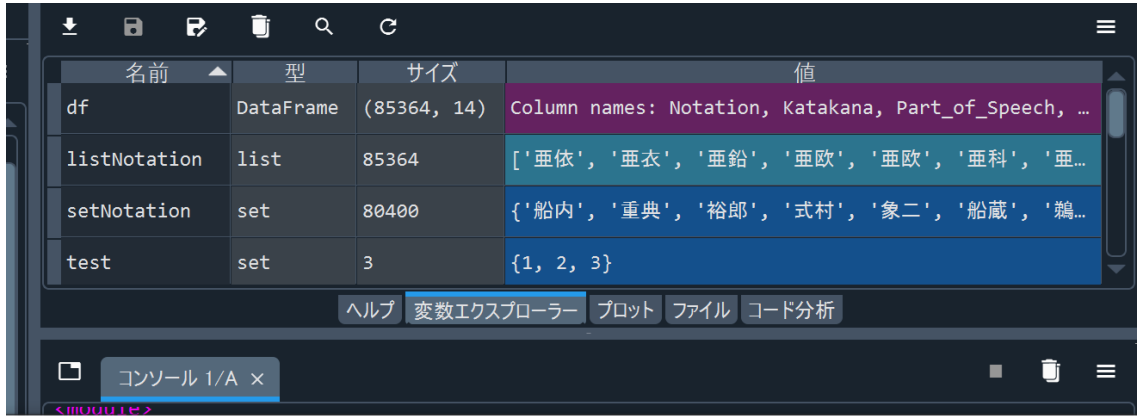
7.4 他のリストに対する多彩の操作

- .sort()による要素をある順で並べ替える
- len()による長さを調べる
- .count()による特定要素の頻度を探す
- .join()による文字列に戻る
- リスト関連のコマンドは他にもたくさんありますが、長くなるので割愛します



7.5 BIGDATAのための辞書型

- NTTDATABASEの中に漢字2文字の熟語はおよそ8万も収録されている
- データを操作する際に
- とある単語のある性質を調べること良くある
- 例えば「学校」の出現頻度
- そのための辞書型
- 辞書型の構成は：KeyとValue
- 学校という単語をkey
- 学校その単語が対応する頻度をvalue
- その鍵（key）によって値（value）を





7.6 辞書型の作り型

- 辞書型の作り型が複数あるが
- Pandasと合わせて操作する場合最も簡単の方法は
- dict(zip(keyとするの列,valueとするの列))
- そして作られた辞書自身も変量なので名前が必要
- 命名する際にもなるべく読めるように
- dictNotation2Freq = dict(zip(df.Notation, df.Nttfreq))
- dictNotation2Freq['調べたい語彙']で
- 出現頻度を簡単に探せる

```
In [41]: dictNotation2Freq = dict(zip(df.Notation, df.Nttfreq))

In [42]: dictNotation2Freq['学校']
Out[42]: 51047

In [43]: dictNotation2Freq['大学']
Out[43]: 51971

In [44]: dictNotation2Freq['亜鉛']
Out[44]: 545

In [45]: dictNotation2Freq['偶然']
Out[45]: 1473

In [46]:
```

7.7 辞書型を作る場合の注意事項

- 辞書型は鍵を持って値を探すもの
- 故に、鍵となるものは唯一無二
- 数ページ前が記述した通り、今のdfの中に一つの語彙を品詞ごとに収録されている
- 品詞によって語彙の出現頻度が変わる
- そのまま辞書型を作ると
- 後収録されたvalueは古いvalueを上書きしてしまう
- スクリプトに確認してみよう

```
In [48]: list1 = ['a', 'b', 'b', 'c']

In [49]: list2 = [1, 2, 3, 4]

In [50]: tmp = dict(zip(list1,list2))

In [51]: print(tmp)
{'a': 1, 'b': 3, 'c': 4}

In [52]: list3 = ['a', 'b', 'b', 'c', 'c']

In [53]: list4 = [1, 2, 3, 4, 3]

In [54]: tmp2 = dict(zip(list3,list4))

In [55]: print(tmp2)
{'a': 1, 'b': 3, 'c': 3}

In [56]:
```

8. Pandasを使ってdataを操る

- 1ページ前が記述したように
- 表記2頻度の辞書型を作る際に、様々の品詞による頻度が上書きする問題がある
- 今は一番高い頻度だけ保持するストラテジーを取る
- そのため、元のdfを頻度による昇順の操作が必要
- 基本的にはexcelで実現できる操作なら
- Pandasはほぼ可能、もちろんexcelではできない操作も
- そしてメソードを思い出せ
- 今はpandasの内蔵関数つまりpandasのメソードを使おう

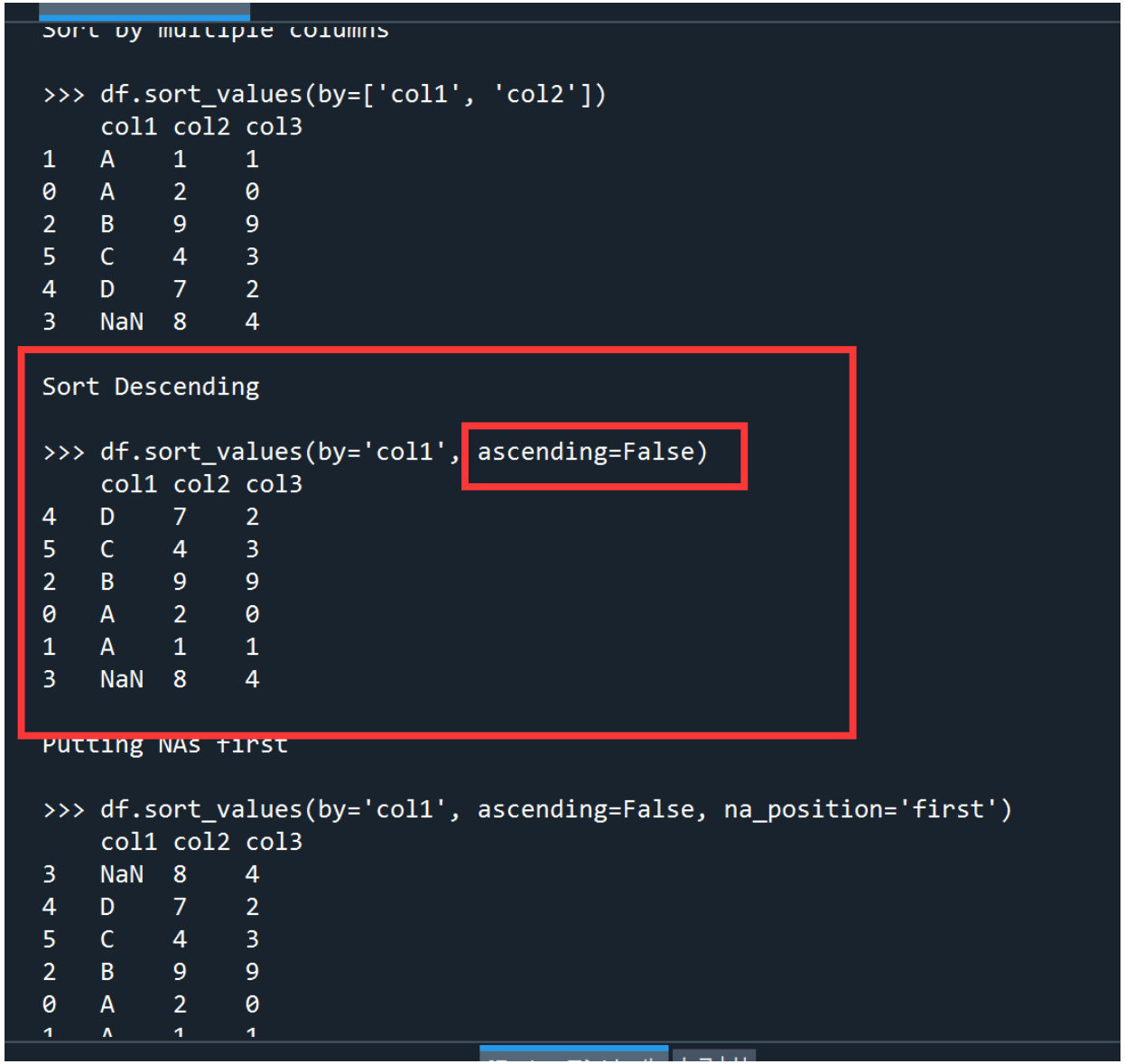
8.1 pandasのメソードについて

- とりあえずgoogle先生に
- すべて覚える必要がない！
- もうこの時代に入ってしまったので
- その時代の利点を活かそう
- もちろん今日のスクリプトの中に答えが書いてる



8.2 昇順して辞書を作る

- もちろんgoogle先生以外
- Python内蔵のhelpコマンドも覚えてください
- 今は昇順だけと、降順する場合？
- Helpコマンドで確認！
- そしてスクリプト通りに辞書を作ろ



9. 簡単なif文で条件判断

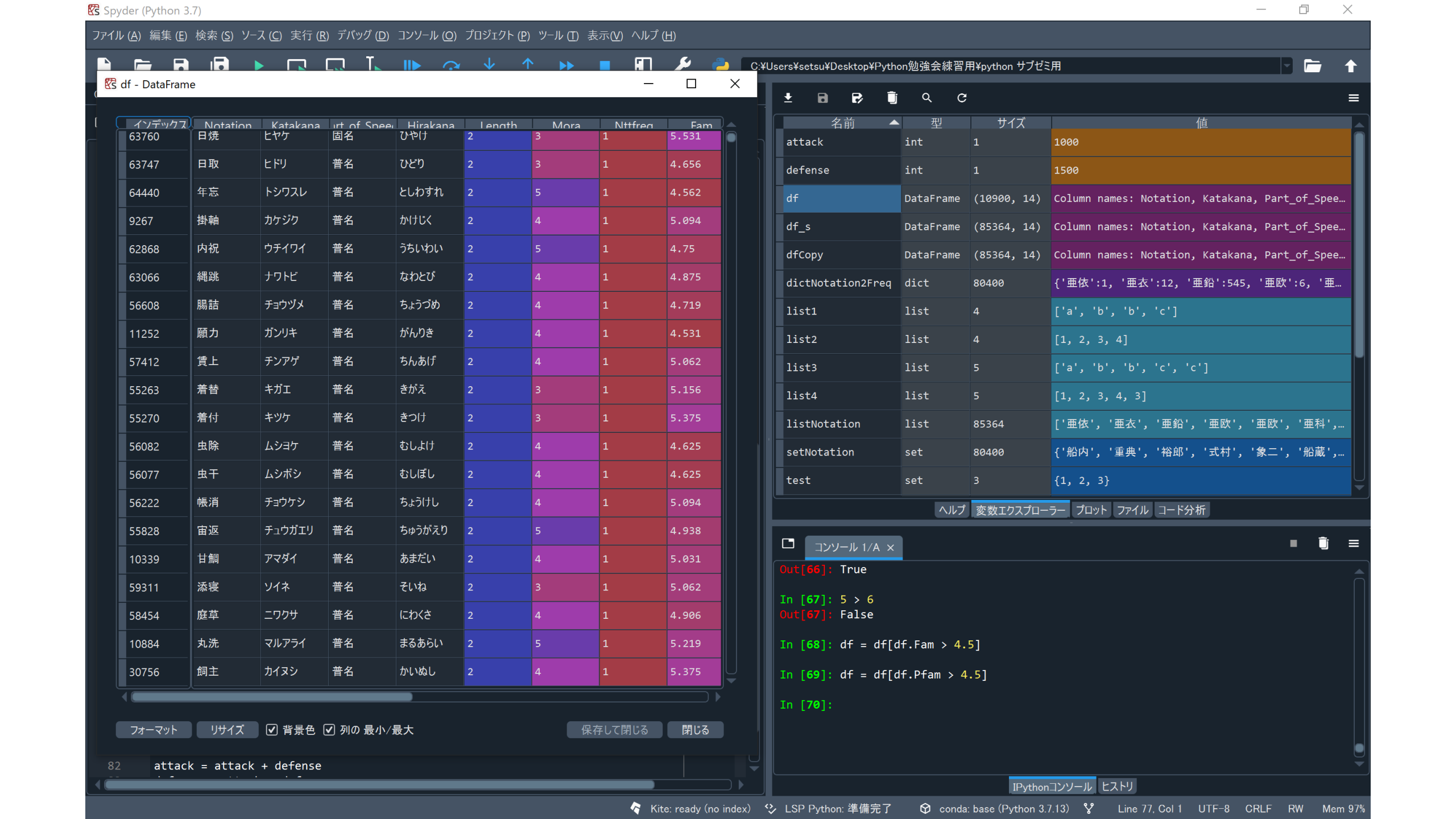
- Bigdataの時代なので
- 一番の問題はdataの過剰
- 例えば、abemaに就職した先輩の話の中に
- お客様の年齢と性別によって好きな番組を推測しようとする
- お客様の利用データがあって、最初の問題は
- ザッピングの特定と除去
- 今のdfも同じ問題がある
- つまり親密度が過ぎる語彙は多分使えない

9.1 if文の最も簡単の例>と<

- If文で判断した結果は真と偽のみ
- True False
- でも実際にpandasを使う場合はもっと簡単です
- Pandasでのデータ判断の本質も[]によるスライス
- df[df.列の名 > 閾値]
- 例えば文字判断課題の目安は親密度と音声親密度4.5以上

- 6. 式 (expression) - 比較 — Python 3.8.5 ドキュメント

演算子	結果
`x < y`	`x` が `y` より小さければ `True`
`x <= y`	`x` が `y` より小さいか等しければ `True`
`x > y`	`x` が `y` より大きければ `True`
`x >= y`	`x` が `y` より大きいか等しければ `True`
`x == y`	`x` と `y` の値が等しければ `True`
`x != y`	`x` と `y` の値が等しくなければ `True`
`x is y`	`x` と `y` が同じオブジェクトであれば `True`
`x is not y`	`x` と `y` が同じオブジェクトでなければ `True`
`x in y`	`x` が `y` に含まれていれば `True`
`x not in y`	`x` が `y` に含まれていなければ `True`


Spyder (Python 3.7)
ファイル (A) 編集 (E) 検索 (S) ソース (C) 実行 (R) デバッグ (D) コンソール (O) プロジェクト (P) ツール (T) 表示(V) ヘルプ (H)
C:¥Users¥setsu¥Desktop¥Python勉強会練習用¥python サブゼミ用
df - DataFrame
インデックス Notation Katakana rt of Spee Hiragana Length Mora Nttfreq Fam
63760 日焼 ヒヤケ 固名 ひやけ 2 3 1 5.531
63747 日取 ヒドリ 普名 ひどり 2 3 1 4.656
64440 年忘 トシワスレ 普名 としわすれ 2 5 1 4.562
9267 掛軸 カケジク 普名 かけじく 2 4 1 5.094
62868 内祝 ウチイワイ 普名 うちいわい 2 5 1 4.75
63066 縄跳 ナワトビ 普名 なわとび 2 4 1 4.875
56608 腸詰 チョウヅメ 普名 ちょうづめ 2 4 1 4.719
11252 願力 ガンリキ 普名 がんりき 2 4 1 4.531
57412 賃上 チンアゲ 普名 ちんあげ 2 4 1 5.062
55263 着替 キガエ 普名 きがえ 2 3 1 5.156
55270 着付 キツケ 普名 きつけ 2 3 1 5.375
56082 虫除 ムシヨケ 普名 むしよけ 2 4 1 4.625
56077 虫干 ムシボシ 普名 むしぼし 2 4 1 4.625
56222 帳消 チョウケシ 普名 ちょうけし 2 4 1 5.094
55828 宙返 チュウガエリ 普名 ちゅうがえり 2 5 1 4.938
10339 甘鯛 アマダイ 普名 あまだい 2 4 1 5.031
59311 添寝 ソイネ 普名 そいね 2 3 1 5.062
58454 庭草 ニワクサ 普名 にわくさ 2 4 1 4.906
10884 丸洗 マルアライ 普名 まるあらい 2 5 1 5.219
30756 飼主 カイヌシ 普名 かいぬし 2 4 1 5.375
フォーマット リサイズ 背景色 列の 最小/最大 保存して閉じる 閉じる
82 attack = attack + defense
名前 型 サイズ 値
attack int 1 1000
defense int 1 1500
df DataFrame (10900, 14) Column names: Notation, Katakana, Part_of_Spee…
df_s DataFrame (85364, 14) Column names: Notation, Katakana, Part_of_Spee…
dfCopy DataFrame (85364, 14) Column names: Notation, Katakana, Part_of_Spee…
dictNotation2Freq dict 80400 {'亜依':1, '亜衣':12, '亜鉛':545, '亜欧':6, '亜…
list1 list 4 ['a', 'b', 'b', 'c']
list2 list 4 [1, 2, 3, 4]
list3 list 5 ['a', 'b', 'b', 'c', 'c']
list4 list 5 [1, 2, 3, 4, 3]
listNotation list 85364 ['亜依', '亜衣', '亜鉛', '亜欧', '亜欧', '亜科',…
setNotation set 80400 {'船内', '重典', '裕郎', '式村', '象二', '船蔵',…
test set 3 {1, 2, 3}
ヘルプ 変数エクスプローラー プロット ファイル コード分析
コンソール 1/A
Out[66]: True
In [67]: 5 > 6
Out[67]: False
In [68]: df = df[df.Fam > 4.5]
In [69]: df = df[df.Pfam > 4.5]
In [70]:
IPythonコンソール ヒストリ
Kite: ready (no index) LSP Python: 準備完了 conda: base (Python 3.7.13) Line 77, Col 1 UTF-8 CRLF RW Mem 97%

10. 簡単な群わけと新しい列の追加

- 生のデータで使うのは結構の制限がある
- 大体の場合データを正規変化したり、対数変化したりする
- その場合dfに新しい列を追加する必要がある
- そのやり方も簡単で
- 特にnumpyとscipyなどのパッケージがあるので
- 複雑な公式を書かなくても簡単に実現可能
- df[‘新しい列名’]　＝　変数or特定の操作
- スクリプトを見て実行しよう

dfHigh - DataFrame

インデックス	Notation	Katakana	rt of Spee	Hirakana	Length	Mora	Nttfreq	Fa
40907	慎重	シンチョウ	普名	しんちょう	2	4	5759	5.781
55249	着手	チャクシュ	普名	ちゃくしゅ	2	3	5762	5.656
30607	視野	シヤ	普名	しや	2	2	5766	5.844
72501	放置	ホウチ	普名	ほうち	2	3	5781	5.844
14273	漁業	ギョギョウ	普名	ぎょぎょう	2	3	5789	5.594
13564	休日	キュウジツ	普名	きゅうじつ	2	4	5790	6.469
76012	迷惑	メイワク	普名	めいわく	2	4	5798	6.188
32176	実力	ジツリョク	普名	じつりょく	2	4	5805	6.438
7308	過剰	カジョウ	普名	かじょう	2	3	5813	5.562
1383	医学	イガク	普名	いがく	2	3	5820	6
50108	総務	ソウム	普名	そうむ	2	3	5821	5.562
35809	春闘	シュントウ	普名	しゅんとう	2	4	5830	5.188
56630	調子	チョウシ	普名	ちょうし	2	3	5832	6.031
56629	調子	チョウシ	固名	ちょうし	2	3	5832	6.031
19138	検証	ケンショウ	普名	けんしょう	2	4	5845	5.562
77439	役所	ヤクドコロ	普名	やくどころ	2	5	5852	5.938
77438	役所	ヤクショ	固名	やくしょ	2	3	5852	5.938
18280	劇団	ゲキダン	普名	げきだん	2	4	5864	5.812
44741	正常	セイジョウ	普名	せいじょう	2	4	5866	6.156
51960	退陣	タイジン	普名	たいじん	2	4	5866	5
40797	審判	シンパン	固名	しんぱん	2	4	5872	5.906
40798	審判	シンパン	普名	しんぱん	2	4	5872	5.906
48414	繊維	センイ	普名	せんい	2	3	5873	5.5
11874	気温	キオン	普名	きおん	2	3	5875	6.281

フォーマット　リサイズ　☑ 背景色　☑ 列の 最小/最大　保存して閉じる　閉じる

dfLow - DataFrame

インデックス	Notation	Katakana	rt of Spee	Hirakana	Length	Mora	Nttfreq	Fa
50946	袖丈	ソデタケ	普名	そでたけ	2	4	1	4.562
13644	宮仕	ミヤヅカエ	普名	みやづかえ	2	5	1	4.688
51323	多目	オオメ	普名	おおめ	2	3	1	4.938
50117	総立	ソウダチ	普名	そうだち	2	4	1	4.781
49867	早漏	ソウロウ	普名	そうろう	2	4	1	4.531
54239	段取	ダンドリ	普名	だんどり	2	4	1	4.875
53973	淡白	タンパク	普名	たんぱく	2	4	1	5.75
54325	値下	ネサゲ	普名	ねさげ	2	3	1	5.062
52392	大荒	オオアレ	普名	おおあれ	2	4	1	4.875
52391	大荒	オオアレ	固名	おおあれ	2	4	1	4.875
53173	滝壷	タキツボ	普名	たきつぼ	2	4	1	4.514
45352	生唾	ナマツバ	普名	なまつば	2	4	1	4.781
15607	玉葱	タマネギ	普名	たまねぎ	2	4	1	5.875
48685	前向	マエムキ	普名	まえむき	2	4	1	5.406
48651	前掛	マエカケ	普名	まえかけ	2	4	1	4.844
49296	粗茶	ソチャ	普名	そちゃ	2	2	1	4.781
47356	絶倫	ゼツリン	普名	ぜつりん	2	4	1	4.625
60321	冬着	フユギ	普名	ふゆぎ	2	3	1	5.188
60960	湯垢	ユアカ	普名	ゆあか	2	3	1	4.625
63761	日焼	ヒヤケ	普名	ひやけ	2	3	1	5.531
63760	日焼	ヒヤケ	固名	ひやけ	2	3	1	5.531
63747	日取	ヒドリ	普名	ひどり	2	3	1	4.656
64440	年忘	トシワスレ	普名	としわすれ	2	5	1	4.562
9267	掛軸	カケジク	普名	かけじく	2	4	1	5.094

フォーマット　リサイズ　☑ 背景色　☑ 列の 最小/最大　保存して閉じる　閉じる