○「MQL4と外部アプリの連携；(その２)」　別稿２　amenbo the 3rd

2018.04.14

◆構成要素別プログラム；日本語形態素解析プログラム（例）

本稿では、テキスト・データを形態素に分解する一番簡単と思われる「手法」を解説します。
読者が、本稿を核（ベース）として更に高度な方法を身に着けることを期待します。
（Python コード（文法）の解説はしません、WEB 上に豊富にある情報を参照ください）

目次

１．Python 用モジュール追加	Ｐ１
２．基礎の基礎（Python プログラム）	
（１）プログラム例－１	Ｐ２
（２）実行結果	Ｐ３
（３）概要解説	Ｐ４
３．テキスト・ファイルのデータ解析（Python プログラム）	
（１）プログラム例－２	Ｐ４
（２）実行結果	Ｐ７
（３）概要解説	Ｐ１１

１．Python 用モジュール追加

本稿で使う日本語形態素解析用のライブラリは「janome（蛇の目）」です。
色々なライブラリがありますが、インストールが簡単なので選びました。
コマンドプロンプト（コンソール）上、python への path が通る場所（フォルダー）で
下記の「pip」コマンドを使いインストールします。

```
pip install janome
```

上記のコマンドを実行すると、
PyPI サイト（PyPI-the Python Package Index）に登録された「janome」ライブラリを
Python にインストールします。
通常のインストール先は「(python のインストール先）/Lib/site-packages」フォルダです。

・janome；　「日本語形態素解析」用のライブラリです

※「Janome」の詳細は、下記を参照ください。
他にも WEB 上に情報が多々ありますので、興味のある方は調査すると面白いです。

http://mocobeta.github.io/janome/

２．基礎の基礎（Python プログラム）

（１）プログラム例－１

[text_mining_01.py]

```
# -*- coding: utf-8 -*-
"""
Created on Tue Mar 20 23:39:48 2018
「text_mining_01.py」
@author: amenbo
"""
from janome.tokenizer import Tokenizer
#t=Tokenizer()#ケース１
t=Tokenizer("user_simpledic.csv",udic_type="simpledic",udic_enc="utf8")#ケース２
'''
  utf8 で作ったユーザー辞書を使う場合
  terapad 等でテキスト「.txt」として作成してから、「.csv」に変える
'''
for token in t.tokenize('日経平均先物や為替とビットコインの間に関連性、つまり相関はあるのでしょうか。'):
    print(token)
```

※「ケース１」と「ケース２」は、どちらかの「#」を外して動作させてください。
　「ケース１」と「ケース２」では動作が若干異なります。
※「#」記号より後ろの１行分（文字）はコメントと見なされます。

ユーザー辞書；(本稿では、簡易版の辞書としました。)
[user_simpledic.csv]

```
日経平均先物,カスタム名詞,ニッケイヘイキンサキモノ
ビットコイン,カスタム名詞,ビットコイン
関連性,カスタム名詞,カンレンセイ
```

作成方法；　辞書は「文字コード；utf-8」で作っておくほうが無難です。
TeraPad などで「文字コード」を「utf-8」に指定して、内容を書込み後に
一旦「user_simpledic.txt」として保存します。
その後、拡張子を「.txt」から「.csv」に変更して作成します。

このファイルは本稿では「text_mining_01.py」と同一の場所（フォルダー中）に置いています。

（２）実行結果

「ケース１」の場合・・「t=Tokenizer()#ケース１」を活かす

```
日経　　名詞,固有名詞,組織,*,*,*,日経,ニッケイ,ニッケイ
平均　　名詞,サ変接続,*,*,*,*,平均,ヘイキン,ヘイキン
先物　　名詞,一般,*,*,*,*,先物,サキモノ,サキモノ
や　　助詞,並立助詞,*,*,*,*,や,ヤ,ヤ
為替　　名詞,一般,*,*,*,*,為替,カワセ,カワセ
と　　助詞,並立助詞,*,*,*,*,と,ト,ト
ビット　　名詞,一般,*,*,*,*,ビット,ビット,ビット
コイン　　名詞,一般,*,*,*,*,コイン,コイン,コイン
の　　助詞,連体化,*,*,*,*,の,ノ,ノ
間　　名詞,一般,*,*,*,*,間,マ,マ
に　　助詞,格助詞,一般,*,*,*,に,ニ,ニ
関連　　名詞,サ変接続,*,*,*,*,関連,カンレン,カンレン
性　　名詞,接尾,一般,*,*,*,性,セイ,セイ
、　　記号,読点,*,*,*,*,、,、,、
つまり　　接続詞,*,*,*,*,*,つまり,ツマリ,ツマリ
相関　　名詞,サ変接続,*,*,*,*,相関,ソウカン,ソーカン
は　　助詞,係助詞,*,*,*,*,は,ハ,ワ
ある　　動詞,自立,*,*,五段・ラ行,基本形,ある,アル,アル
の　　名詞,非自立,一般,*,*,*,の,ノ,ノ
でしょ　　助動詞,*,*,*,特殊・デス,未然形,です,デショ,デショ
う　　助動詞,*,*,*,不変化型,基本形,う,ウ,ウ
か　　助詞,副助詞／並立助詞／終助詞,*,*,*,*,か,カ,カ
。　　記号,句点,*,*,*,*,。,。,。
```

「ケース２」の場合・・「t=Tokenizer("user_simpl・・・c="utf8")#ケース２」を活かす

```
日経平均先物　カスタム名詞,*,*,*,*,*,日経平均先物,ニッケイヘイキンサキモノ,ニッケイヘイキンサキモノ
や　　助詞,並立助詞,*,*,*,*,や,ヤ,ヤ
為替　　名詞,一般,*,*,*,*,為替,カワセ,カワセ
と　　助詞,並立助詞,*,*,*,*,と,ト,ト
ビットコイン　カスタム名詞,*,*,*,*,*,ビットコイン,ビットコイン,ビットコイン
の　　助詞,連体化,*,*,*,*,の,ノ,ノ
間　　名詞,一般,*,*,*,*,間,マ,マ
に　　助詞,格助詞,一般,*,*,*,に,ニ,ニ
関連性　　カスタム名詞,*,*,*,*,*,関連性,カンレンセイ,カンレンセイ
、　　記号,読点,*,*,*,*,、,、,、
つまり　　接続詞,*,*,*,*,*,つまり,ツマリ,ツマリ
相関　　名詞,サ変接続,*,*,*,*,相関,ソウカン,ソーカン
は　　助詞,係助詞,*,*,*,*,は,ハ,ワ
ある　　動詞,自立,*,*,五段・ラ行,基本形,ある,アル,アル
の　　名詞,非自立,一般,*,*,*,の,ノ,ノ
でしょ　　助動詞,*,*,*,特殊・デス,未然形,です,デショ,デショ
う　　助動詞,*,*,*,不変化型,基本形,う,ウ,ウ
か　　助詞,副助詞／並立助詞／終助詞,*,*,*,*,か,カ,カ
。　　記号,句点,*,*,*,*,。,。,。
```

（３）概要解説

```
# -*- coding: utf-8 -*-
```

Spyder で［ファイル］－［新規ファイル］を選択すると、作成される python プログラム・ファイルに自動的に書き込まれているコードです。
・プログラムの文字コードが「UTF-8」に指定されています。

```
"""
Created on Tue Mar 20 23:39:48 2018
「text_mining_01.py」
@author: amenbo
"""
```

「"""」と「"""」で囲まれた部分は、コメントとして処理されます。
・[text_mining_01.py] 部分は、小生が追記しましたが、他はファイル作成時に自動的に作成されたものです。

```
from janome.tokenizer import Tokenizer                    ①
#t=Tokenizer()#ケース１                                   ②-1
t=Tokenizer("user_simpledic.csv",udic_type="simpledic",udic_enc="utf8")#ケース２　②-2
```

① pip でインストール済の「janome」ライブラリから、「日本語形態素」分析用の「Tokenizer」パッケージをインポートします。
②-1（ケース 1）Tokenizer のオブジェクトを作成します
②-2（ケース 2）Tokenizer のオブジェクトを作成するのは同じですが、janome 備え付け辞書以外に、ユーザー辞書（簡易版、文字コードは utf-8）も使う事を指定しています。

```
'''
  utf8 で作ったユーザー辞書を使う場合
  terapad 等でテキスト「.txt」として作成してから、「.csv」に変える
'''
```

※これはコメントです

```
for token in t.tokenize('日経平均先物や為替とビットコインの間に関連性、つまり相関はあるのでしょうか。'):    ③
    print(token)                                                                                    ④
```

③ tokenizer()メソッドに「形態素」解析したい文字列「'*** '」を渡しています。
④ 「形態素」解析した結果をプリント（表示）します。

３．テキスト・ファイルのデータ解析（Python プログラム）

（１）プログラム例－２

[text_mining_02.py]

```
# -*- coding: utf-8 -*-
"""
Created on Thu Mar 22 22:19:31 2018
「text_mining_02.py」
@author: amenbo
"""
# python 解析器 janome をインポート - 1
from janome.tokenizer import Tokenizer

# 形態素解析用オブジェクトの生成 - 2
text = Tokenizer()

# txt ファイルからデータの読み込み - 3
text_file = open("text_1.txt")#ファイルは shift-jis　で動作
#text_file = open("text_1.txt",encoding="utf-8")#ファイルもＵＴＦ－８にしている　この場合も動作ＯＫ
bindata = text_file.read()
txt = bindata

print("¥n")
#print(txt) #動作ＯＫ

# txt から読み込んだデータを形態素解析 - 4
#lines = txt.split("¥r¥n")#なんか、うまく行かない
lines = txt.split("¥n")
for i in lines:
    print(i)
    print("¥n")
    text_c = text.tokenize(i)
    for j in text_c:
      print(j)
    print("¥n")

print("-----------------------------------------")
print("¥n")
```

```
# テキストを一行ごとに処理 - 5
word_dic = {}
#lines_1 = txt.split("¥r¥n")#何か、上手く分離できない
lines_1 = txt.split("¥n")
print(lines_1)
print("¥n")
for line in lines_1:
    malist = text.tokenize(line)
    #print(malist)
    for w in malist:
        word = w.surface
        #print(word)
        ps = w.part_of_speech # 品詞 - 6
        re=w.reading #  読み
        print("word:%s ¥t ps:%s ¥t re:%s" %(word,ps,re))
        if ps.find("名詞") < 0: continue # 名詞だけをカウント - 7
        if not word in word_dic:
            word_dic[word] = 0
        word_dic[word] += 1

print("-------------------------------------------")
print("word_dic  の中身 :  ¥n")
print(word_dic)
print("-------------------------------------------")
print("名刺の使用頻度 : ¥n")

# よく使われる単語を表示 - 8
keys = sorted(word_dic.items(), key=lambda x:x[1], reverse=True)
for word, cnt in keys[:50]:
    print("{0}({1}) ".format(word,cnt), end="")
```

[text_1.txt]

私の名前はアメンボです。
住まいは狛江市ですが国領駅の方が近いです。
今、テキストマイニングの練習をしています。
趣味は「地域活動」と MQL4 原語による「プログラミング」です。
貴方の名前は。
お住まいは何所ですか、最寄り駅は何所ですか、国領駅から近いですか？
お仕事と趣味を教えてください、ところで投資には興味がありますか。

作成方法；

・・本稿では、敢て「shift-jis」で作ってみました。
「utf-8」でも当然、問題ありません。

（2）実行結果

※ユーザー辞書を設定していないので、一部は不自然な形態素に分解されています。
例；「狛江　市」、「国領　駅」、「MQL 4」など

私の名前はアメンボです。

```
私          名詞,代名詞,一般,*,*,*,私,ワタシ,ワタシ
の          助詞,連体化,*,*,*,*,の,ノ,ノ
名前        名詞,一般,*,*,*,*,名前,ナマエ,ナマエ
は          助詞,係助詞,*,*,*,*,は,ハ,ワ
アメンボ    名詞,一般,*,*,*,*,アメンボ,アメンボ,アメンボ
です        助動詞,*,*,*,特殊・デス,基本形,です,デス,デス
。          記号,句点,*,*,*,*,。,。,。
```

住まいは狛江市ですが国領駅の方が近いです。

```
住まい      名詞,一般,*,*,*,*,住まい,スマイ,スマイ
は          助詞,係助詞,*,*,*,*,は,ハ,ワ
狛江        名詞,固有名詞,地域,一般,*,*,狛江,コマエ,コマエ
市          名詞,接尾,地域,*,*,*,市,シ,シ
です        助動詞,*,*,*,特殊・デス,基本形,です,デス,デス
が          助詞,接続助詞,*,*,*,*,が,ガ,ガ
国領        名詞,固有名詞,地域,一般,*,*,国領,コクリョウ,コクリョー
駅          名詞,接尾,地域,*,*,*,駅,エキ,エキ
の          助詞,連体化,*,*,*,*,の,ノ,ノ
方          名詞,非自立,一般,*,*,*,方,ホウ,ホー
が          助詞,格助詞,一般,*,*,*,が,ガ,ガ
近い        形容詞,自立,*,*,形容詞・アウオ段,基本形,近い,チカイ,チカイ
です        助動詞,*,*,*,特殊・デス,基本形,です,デス,デス
。          記号,句点,*,*,*,*,。,。,。
```

今、テキストマイニングの練習をしています。

```
今          名詞,副詞可能,*,*,*,*,今,イマ,イマ
、          記号,読点,*,*,*,*,、,、,、
テキスト    名詞,一般,*,*,*,*,テキスト,テキスト,テキスト
マイニング  名詞,サ変接続,*,*,*,*,マイニング,マイニング,マイニング
の          助詞,連体化,*,*,*,*,の,ノ,ノ
練習        名詞,サ変接続,*,*,*,*,練習,レンシュウ,レンシュー
を          助詞,格助詞,一般,*,*,*,を,ヲ,ヲ
し          動詞,自立,*,*,サ変・スル,連用形,する,シ,シ
```

```
て          助詞,接続助詞,*,*,*,*,て,テ,テ
い          動詞,非自立,*,*,一段,連用形,いる,イ,イ
ます        助動詞,*,*,*,特殊・マス,基本形,ます,マス,マス
。          記号,句点,*,*,*,*,。,。,。
```

趣味は「地域活動」と MQL4 原語による「プログラミング」です。

```
趣味        名詞,一般,*,*,*,*,趣味,シュミ,シュミ
は          助詞,係助詞,*,*,*,*,は,ハ,ワ
「          記号,括弧開,*,*,*,*,「,「,「
地域        名詞,一般,*,*,*,*,地域,チイキ,チイキ
活動        名詞,サ変接続,*,*,*,*,活動,カツドウ,カツドー
」          記号,括弧閉,*,*,*,*,」,」,」
と          助詞,格助詞,引用,*,*,*,と,ト,ト
MQL         名詞,一般,*,*,*,*,MQL,*,*
4           名詞,数,*,*,*,*,4,*,*
原語        名詞,一般,*,*,*,*,原語,ゲンゴ,ゲンゴ
による      助詞,格助詞,連語,*,*,*,による,ニヨル,ニヨル
「          記号,括弧開,*,*,*,*,「,「,「
プログラミング  名詞,サ変接続,*,*,*,*,プログラミング,プログラミング,プログラミング
」          記号,括弧閉,*,*,*,*,」,」,」
です        助動詞,*,*,*,特殊・デス,基本形,です,デス,デス
。          記号,句点,*,*,*,*,。,。,。
```

貴方の名前は。

```
貴方        名詞,代名詞,一般,*,*,*,貴方,アナタ,アナタ
の          助詞,連体化,*,*,*,*,の,ノ,ノ
名前        名詞,一般,*,*,*,*,名前,ナマエ,ナマエ
は          助詞,係助詞,*,*,*,*,は,ハ,ワ
。          記号,句点,*,*,*,*,。,。,。
```

お住まいは何所ですか、最寄り駅は何所ですか、国領駅から近いですか？

```
お          接頭詞,名詞接続,*,*,*,*,お,オ,オ
住まい      名詞,一般,*,*,*,*,住まい,スマイ,スマイ
は          助詞,係助詞,*,*,*,*,は,ハ,ワ
何          名詞,代名詞,一般,*,*,*,何,ナニ,ナニ
所          名詞,接尾,一般,*,*,*,所,ショ,ショ
です        助動詞,*,*,*,特殊・デス,基本形,です,デス,デス
か          助詞,副助詞／並立助詞／終助詞,*,*,*,*,か,カ,カ
、          記号,読点,*,*,*,*,、,、,、
最寄り駅    名詞,一般,*,*,*,*,最寄り駅,モヨリエキ,モヨリエキ
は          助詞,係助詞,*,*,*,*,は,ハ,ワ
何          名詞,代名詞,一般,*,*,*,何,ナニ,ナニ
所          名詞,接尾,一般,*,*,*,所,ショ,ショ
です        助動詞,*,*,*,特殊・デス,基本形,です,デス,デス
か          助詞,副助詞／並立助詞／終助詞,*,*,*,*,か,カ,カ
、          記号,読点,*,*,*,*,、,、,、
国領        名詞,固有名詞,地域,一般,*,*,国領,コクリョウ,コクリョー
駅          名詞,接尾,地域,*,*,*,駅,エキ,エキ
から        助詞,格助詞,一般,*,*,*,から,カラ,カラ
近い        形容詞,自立,*,*,形容詞・アウオ段,基本形,近い,チカイ,チカイ
です        助動詞,*,*,*,特殊・デス,基本形,です,デス,デス
か          助詞,副助詞／並立助詞／終助詞,*,*,*,*,か,カ,カ
？          記号,一般,*,*,*,*,？,？,？
```

お仕事と趣味を教えてください、ところで投資には興味がありますか。

```
お          接頭詞,名詞接続,*,*,*,*,お,オ,オ
仕事        名詞,サ変接続,*,*,*,*,仕事,シゴト,シゴト
と          助詞,並立助詞,*,*,*,*,と,ト,ト
趣味        名詞,一般,*,*,*,*,趣味,シュミ,シュミ
```

```
を　　助詞,格助詞,一般,*,*,*,を,ヲ,ヲ
教え　　動詞,自立,*,*,一段,連用形,教える,オシエ,オシエ
て　　助詞,接続助詞,*,*,*,*,て,テ,テ
ください　　動詞,非自立,*,*,五段・ラ行特殊,命令ｉ,くださる,クダサイ,クダサイ
、　　記号,読点,*,*,*,*,、,、,、
ところで　　接続詞,*,*,*,*,*,ところで,トコロデ,トコロデ
投資　　名詞,サ変接続,*,*,*,*,投資,トウシ,トーシ
に　　助詞,格助詞,一般,*,*,*,に,ニ,ニ
は　　助詞,係助詞,*,*,*,*,は,ハ,ワ
興味　　名詞,一般,*,*,*,*,興味,キョウミ,キョーミ
が　　助詞,格助詞,一般,*,*,*,が,ガ,ガ
あり　　動詞,自立,*,*,五段・ラ行,連用形,ある,アリ,アリ
ます　　助動詞,*,*,*,特殊・マス,基本形,ます,マス,マス
か　　助詞,副助詞／並立助詞／終助詞,*,*,*,*,か,カ,カ
。　　記号,句点,*,*,*,*,。,。,。
```

['私の名前はアメンボです。', '住まいは狛江市ですが国領駅の方が近いです。', '今、テキストマイニングの練習をしています。', '趣味は「地域活動」と MQL4 原語による「プログラミング」です。', '貴方の名前は。', 'お住まいは何所ですか、最寄り駅は何所ですか、国領駅から近いですか？', 'お仕事と趣味を教えてください、ところで投資には興味がありますか。']

```
word:私　ps:名詞,代名詞,一般,*　re:ワタシ
word:の　ps:助詞,連体化,*,*　re:ノ
word:名前　ps:名詞,一般,*,*　re:ナマエ
word:は　ps:助詞,係助詞,*,*　re:ハ
word:アメンボ　ps:名詞,一般,*,*　re:アメンボ
word:です　ps:助動詞,*,*,*　re:デス
word:。　ps:記号,句点,*,*　re:。
word:住まい　ps:名詞,一般,*,*　re:スマイ
word:は　ps:助詞,係助詞,*,*　re:ハ
word:狛江　ps:名詞,固有名詞,地域,一般　re:コマエ
word:市　ps:名詞,接尾,地域,*　re:シ
word:です　ps:助動詞,*,*,*　re:デス
word:が　ps:助詞,接続助詞,*,*　re:ガ
word:国領　ps:名詞,固有名詞,地域,一般　re:コクリョウ
word:駅　ps:名詞,接尾,地域,*　re:エキ
word:の　ps:助詞,連体化,*,*　re:ノ
word:方　ps:名詞,非自立,一般,*　re:ホウ
word:が　ps:助詞,格助詞,一般,*　re:ガ
word:近い　ps:形容詞,自立,*,*　re:チカイ
word:です　ps:助動詞,*,*,*　re:デス
word:。　ps:記号,句点,*,*　re:。
word:今　ps:名詞,副詞可能,*,*　re:イマ
word:、　ps:記号,読点,*,*　re:、
word:テキスト　ps:名詞,一般,*,*　re:テキスト
word:マイニング　ps:名詞,サ変接続,*,*　re:マイニング
word:の　ps:助詞,連体化,*,*　re:ノ
word:練習　ps:名詞,サ変接続,*,*　re:レンシュウ
word:を　ps:助詞,格助詞,一般,*　re:ヲ
word:し　ps:動詞,自立,*,*　re:シ
word:て　ps:助詞,接続助詞,*,*　re:テ
word:い　ps:動詞,非自立,*,*　re:イ
word:ます　ps:助動詞,*,*,*　re:マス
word:。　ps:記号,句点,*,*　re:。
word:趣味　ps:名詞,一般,*,*　re:シュミ
word:は　ps:助詞,係助詞,*,*　re:ハ
word:「　ps:記号,括弧開,*,*　re:「
word:地域　ps:名詞,一般,*,*　re:チイキ
word:活動　ps:名詞,サ変接続,*,*　re:カツドウ
word:」　ps:記号,括弧閉,*,*　re:」
word:と　ps:助詞,格助詞,引用,*　re:ト
word:MQL　ps:名詞,一般,*,*　re:*
word:4　ps:名詞,数,*,*　re:*
word:原語　ps:名詞,一般,*,*　re:ゲンゴ
word:による　ps:助詞,格助詞,連語,*　re:ニヨル
word:「　ps:記号,括弧開,*,*　re:「
```

```
word:プログラミング　　ps:名詞,サ変接続,*,*　　re:プログラミング
word:」　ps:記号,括弧閉,*,*　re:」
word:です　　ps:助動詞,*,*,*　　re:デス
word:。　ps:記号,句点,*,*　　re:。
word:貴方　　ps:名詞,代名詞,一般,*　　re:アナタ
word:の　ps:助詞,連体化,*,*　re:ノ
word:名前　　ps:名詞,一般,*,*　　re:ナマエ
word:は　ps:助詞,係助詞,*,*　re:ハ
word:。　ps:記号,句点,*,*　　re:。
word:お　ps:接頭詞,名詞接続,*,*　　re:オ
word:住まい　　ps:名詞,一般,*,*　　re:スマイ
word:は　ps:助詞,係助詞,*,*　re:ハ
word:何　ps:名詞,代名詞,一般,*　　re:ナニ
word:所　ps:名詞,接尾,一般,*　re:ショ
word:です　　ps:助動詞,*,*,*　　re:デス
word:か　ps:助詞,副助詞／並立助詞／終助詞,*,*　　re:カ
word:、　ps:記号,読点,*,*　　re:、
word:最寄り駅　　ps:名詞,一般,*,*　　re:モヨリエキ
word:は　ps:助詞,係助詞,*,*　re:ハ
word:何　ps:名詞,代名詞,一般,*　　re:ナニ
word:所　ps:名詞,接尾,一般,*　re:ショ
word:です　　ps:助動詞,*,*,*　　re:デス
word:か　ps:助詞,副助詞／並立助詞／終助詞,*,*　　re:カ
word:、　ps:記号,読点,*,*　　re:、
word:国領　　ps:名詞,固有名詞,地域,一般　　re:コクリョウ
word:駅　ps:名詞,接尾,地域,*　re:エキ
word:から　　ps:助詞,格助詞,一般,*　　re:カラ
word:近い　　ps:形容詞,自立,*,*　re:チカイ
word:です　　ps:助動詞,*,*,*　　re:デス
word:か　ps:助詞,副助詞／並立助詞／終助詞,*,*　　re:カ
word:？　ps:記号,一般,*,*　　re:？
word:お　ps:接頭詞,名詞接続,*,*　　re:オ
word:仕事　　ps:名詞,サ変接続,*,*　　re:シゴト
word:と　ps:助詞,並立助詞,*,*　　re:ト
word:趣味　　ps:名詞,一般,*,*　　re:シュミ
word:を　ps:助詞,格助詞,一般,*　　re:ヲ
word:教え　　ps:動詞,自立,*,*　　re:オシエ
word:て　ps:助詞,接続助詞,*,*　　re:テ
word:ください　　ps:動詞,非自立,*,*　re:クダサイ
word:、　ps:記号,読点,*,*　　re:、
word:ところで　　ps:接続詞,*,*,*　　re:トコロデ
word:投資　　ps:名詞,サ変接続,*,*　　re:トウシ
word:に　ps:助詞,格助詞,一般,*　　re:ニ
word:は　ps:助詞,係助詞,*,*　re:ハ
word:興味　　ps:名詞,一般,*,*　　re:キョウミ
word:が　ps:助詞,格助詞,一般,*　　re:ガ
word:あり　　ps:動詞,自立,*,*　　re:アリ
word:ます　　ps:動詞,*,*,*　　re:マス
word:か　ps:助詞,副助詞／並立助詞／終助詞,*,*　　re:カ
word:。　ps:記号,句点,*,*　　re:。
--------------------------------------------
word_dic　の中身：

{'私': 1, '名前': 2, 'アメンボ': 1, '住まい': 2, '狛江': 1, '市': 1, '国領': 2, '駅': 2, '方': 1, '今': 1,
'テキスト': 1, 'マイニング': 1, '練習': 1, '趣味': 2, '地域': 1, '活動': 1, 'MQL': 1, '4': 1, '原語': 1, '
プログラミング': 1, '貴方': 1, 'お': 2, '何': 2, '所': 2, '最寄り駅': 1, '仕事': 1, '投資': 1, '興味': 1}
--------------------------------------------
名刺の使用頻度：

名前(2) 住まい(2) 国領(2) 駅(2) 趣味(2) お(2) 何(2) 所(2) 私(1) アメンボ(1) 狛江(1) 市(1) 方(1) 今(1) テキ
スト(1) マイニング(1) 練習(1) 地域(1) 活動(1) MQL(1) 4(1) 原語(1) プログラミング(1) 貴方(1) 最寄り駅(1) 仕
事(1) 投資(1) 興味(1)
```

（３）概要解説

```
# -*- coding: utf-8 -*-
"""
Created on Thu Mar 22 22:19:31 2018
「text_mining_02.py」
@author: amenbo
"""
```

※例－１と同様に、上記部分はコメントとして処理されます。

```
# python 解析器 janome をインポート - 1
from janome.tokenizer import Tokenizer
# 形態素解析用オブジェクトの生成 - 2
text = Tokenizer()
```

※「janome」ライブラリから、「日本語形態素」分析用の「Tokenizer」パッケージを
　インポートしてから、Tokenizer のオブジェクトを作成します
　ただし、Tokenizer オブジェクト作成時に辞書等の指定は行っていません。

```
# txt ファイルからデータの読み込み - 3
text_file = open("text_1.txt")#ファイルは shift-jis　で動作　　①
#text_file = open("text_1.txt",encoding="utf-8")#ファイルもＵＴＦ－８にしている　この場合も動作ＯＫ
bindata = text_file.read()　　②
txt = bindata　　③

print("¥n")
#print(txt) #動作ＯＫ
```

※「例－１」の場合と異なり、「形態素」解析する対象は「text_1.txt」に記載されたテキストです。

手順；　①　先ず、テキストファイル「text_1.txt」を開き、
　　②　その内容を読み取って（.read()）
　　③　一度、テキスト変数（txt）として設定します・・小生のやり方ですが

```
# txt から読み込んだデータを形態素解析 - 4
#lines = txt.split("¥r¥n")#なんか、うまく行かない
lines = txt.split("¥n")                    ④
for i in lines:                            ⑤
    print(i)
    print("¥n")
    text_c = text.tokenize(i)
    for j in text_c:
      print(j)
    print("¥n")

print("-----------------------------------------")
print("¥n")
```

④ 改行コードを利用して、テキストを「１行」ごとに「リスト」形式に分離します
⑤リストの各要素（つまり１行）ごとに「形態素」解析していきます

```
# テキストを一行ごとに処理 - 5
word_dic = {}
#lines_1 = txt.split("¥r¥n")#何か、上手く分離できない
lines_1 = txt.split("¥n")                  ⑥
print(lines_1)                             ⑦
print("¥n")
for line in lines_1:                       ⑧
    malist = text.tokenize(line)
    #print(malist)
    for w in malist:                       ⑨
        word = w.surface
        #print(word)
        ps = w.part_of_speech # 品詞 - 6
        re=w.reading #  読み
        print("word:%s ¥t ps:%s ¥t re:%s" %(word,ps,re))
        if ps.find("名詞") < 0: continue # 名詞だけをカウント – 7          ⑩
        if not word in word_dic:
            word_dic[word] = 0
        word_dic[word] += 1
print("-----------------------------------------")
```

⑥「④」と同じ。改行コードを利用して、テキストを「１行」ごとに「リスト」形式に分離。
⑦上記「⑥」の内容をプリント
⑧１行ごとに形態素解析を行います
⑨上記「⑧」の処理として、

- [.surface] ＝ 名詞
- [.part_of_speech] ＝品詞
- [.reading] ＝読み

を検出して書き出しています。
⑩各種の品詞の中から、同じ「名詞」のみ、出現頻度をカウントしています。

```
print("word_dic  の中身：  ¥n")
print(word_dic)
print("----------------------------------------")
print("名刺の使用頻度： ¥n")
```

・word_dic（辞書）形式に保存されたデータを書出し。

```
# よく使われる単語を表示 - 8
keys = sorted(word_dic.items(), key=lambda x:x[1], reverse=True)
for word, cnt in keys[:50]:
    print("{0}({1}) ".format(word,cnt), end="")
```

・出現頻度の高い「名詞」の「上位５０番目まで」を書き出します。

※janome（日本語形態素解析用のライブラリ）は、導入が非常に簡単な割に高機能で使い易いです。（他にもライブラリは色々あるようですが）
アメンボは、「ほんの触り」程度に使っただけです、興味のある読者は色々と試してみることをお勧めします。

以　上