# データサイエンス特論　プログラム課題　第1回(クラスタリング)

20NewsGroup データに対して、クラスタリング（20 クラス）を行い、授業資料で紹介した Purity 値(p.27)と V-measure(後述の定義を参照)で評価せよ。

締切は、二週間後【7 月 25 日(月)】の深夜(23:59)までで、Moodle LMS (https://lms.imc.tut.ac.jp/course/view.php?id=807)にアップロードすること。今回は、Python コード(または Jupyter notebook の.ipynb ファイル)、実行結果と考察を合わせ、提出すること。Jupyter notebook の場合、マークダウンに考察を記入したものを提出しても可とする。(注：実行結果は評価部分だけでよい)

『コメントとヒント』

(1) 20Newsgroup (https://kdd.ics.uci.edu/databases/20newsgroups/20newsgroups.html) は 20 グループの 20,000 メッセージからなる英語のテキストデータである。今回の課題では、原則、JSON で定義された 20newsgroup データ (https://github.com/selva86/datasets/blob/master/newsgroups.json) を使います。JSON データから、個々のニュースメッセージのテキストやカテゴリを取り出すプログラム例は、たとえば、以下のようにできます。

```
import json

# load JSON 20newsgroup data
with open("newsgroups.json") as fd:
    data = json.load(fd)

# content, target(=class ID), target_name(=newsgroup name)
content = data['content']
target = data['target']
target_names = data['target_names']

# get dictionary values for content, target, and target_names
content_value = content.values()
target_value = target.values()
target_name_value = target_names.values()

# extract lists for content, target, and target_names
```

```
content_value_list = list(content_value) # メッセージテキスト本体
target_value_list = list(target_value) # メッセージのカテゴリ ID
 target_namevalue_list = list(target_name_value)
num_docs = 11314 # (=len(content.keys())
```

この JSON データでのニュースメッセージの総数は 20,000 でなく、内容的な重複等を取り除いた **11,314** になります。カテゴリが 20 ある点は、オリジナルと変わりません。

(2) クラスタリングを行うためには、各ニュースメッセージテキストを固定長の文書ベクトルに変換する必要があります。この変換は、各自の裁量に任せます。以下では、代表例をいくつか紹介します。（ソースは 5 Simple Ways to Tokenize Text in Python (https://towardsdatascience.com/5-simple-ways-to-tokenize-text-in-python-92c6804edfc4)です）

（例１）NLTK パッケージを用いてトークナイズし、単語の語彙を決めてからベクトル化するアプローチ

（例２）scikit-learn (sklearn)パッケージの CountVectorizer と TfidfVectorizer を用いたベクトル化

（例３）gensim パッケージを用いたトークナイズとベクトル化

これらに加えて、以下の手法も参考までに紹介します。

（例４）訓練済みの Sentence Transformer モデルで入力テキストを固定長ベクトル(384 次元)に符号化（注：学習済みモデルを使うため、pytorch（ならびに sentence_transformers (https://www.sbert.net/))のインストールは必要ですが、CPU だけで動作します)。

(3) クラスタリングの手法も任意のもので OK ですが、以下ではもっともポピュラーな K-Means 法を紹介します。たとえば、以下のようなコードでクラスタリングを行い、クラスターID 列を生成します。

```
from sklearn import cluster
# assume fixed-legth vector data is kept in "vec_data"
k_means = cluster.KMeans(n_clusters=20)
k_means.fit(vec_data)
predicted = k_means.predict(vec_data)
```

(4)クラスタリングの評価は Purity と V-measure ですが、真値は(1)で述べた target_value_list にあるため、

truth = target_value_list

purity_score(truth, predicted)で計算した値を計算してください。関数purity_scoreは資料p.27を参考に自作ください。

一方、V-measureのほうは、sklearnの資料

定義自体は、ここ（https://scikit-learn.org/stable/modules/generated/sklearn.metrics.v_measure_score.html#sklearn.metrics.v_measure_score）にあります。また、ここ（https://scikit-learn.org/stable/auto_examples/text/plot_document_clustering.html#sphx-glr-auto-examples-text-plot-document-clustering-py）にプログラム例があります。

を参照ください。なお、こちらのURLにある資料には、20newsgroupの一部を用いて、K-meansでクラスタリングを行いV-measure等の複数の評価尺度で評価を行うところまでのサンプルプログラムが掲載されています。かなり参考になると思います。

(5) 考察部分には、Purityと V-measureから想像される20newsgroupデータを選択したクラスタリング手法のもとで、どうして、これらの値に帰着したかの分析、ならびに、入力の20newsgroupデータ（原則11314文書）から、今回のプログラム課題で、どこが計算時間的にボトルネックであったかの考察も加えてくれることを期待します。

(6) ２種類以上の文書のベクトル化、あるいは２種類以上のクラスタリング手法、あるいはクラスタリング手法に（クラスター数以外の）パラメータがある場合、２つ以上のパラメータでのクラスタリング結果を比較しても結構です。ただし、この(6)はオプションとします。すなわち、最低１種類の文書ベクトル化手法＋１種類のクラスタリング手法だけの結果で結構です。

今回のプログラム課題は、できるだけ自力でやってください。この授業にはTAがいないので、授業担当者にメールが集中する場合、回答できない場合もありますので、あらかじめご了承ください。