term_term_pr_distr_and_inference-multinomial-min_df-1-size-150-max

December 8, 2020

autoreload modules and utilities

```
[1]: %load_ext autoreload
     %autoreload 2
```

import all neceesary libraries/packages

```
[2]: import joblib

     import numpy as np
     import pandas as pd

     from tqdm.notebook import tqdm
     import matplotlib.pyplot as plt

     from scipy.stats import entropy as calculate_entropy

     from sklearn.datasets import fetch_20newsgroups
     from sklearn.model_selection import StratifiedShuffleSplit

     from sklearn.pipeline import Pipeline
     from sklearn.pipeline import FeatureUnion
     from sklearn.naive_bayes import MultinomialNB, GaussianNB
     from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer

     from sklearn.metrics import classification_report, accuracy_score
     from sklearn.metrics import f1_score as calculate_f1_score
     from sklearn.model_selection import train_test_split, StratifiedKFold
```

Utility functions

```
[44]: ## utilities
      # from utils import clean_text

      import string
      from sklearn.base import TransformerMixin
```

```python
import nltk
from nltk import word_tokenize
from nltk.stem import WordNetLemmatizer


nltk.download('stopwords')
nltk.download('wordnet')


def clean_text(text: str, lemmatizer = lambda x: x) -> str:
    # removes upper cases
    text = text.lower().strip()

    # removes punctuation
    for char in string.punctuation:
        text = text.replace(char, " ")

    #lematize the words and join back into string text
    text = " ".join([lemmatizer(word) for word in word_tokenize(text)])
    return text


def data_isvalid(text, analyser, min_character_size, max_character_size):
    return min_character_size <= len(analyser(text)) <= max_character_size


def get_pipeline(vectorizer_type, classifier, use_t2pi, min_df=3,␣
 ↪stop_words=None, lemmatizer = lambda x: x):
    vectorizer = CountVectorizer if vectorizer_type == "count" else␣
 ↪TfidfVectorizer
    models = [
        ('clean_text', CleanTextTransformer(lemmatizer)),

        ("vectorizers", FeatureUnion([
            ('count_binary', CountVectorizer(stop_words=stop_words,␣
 ↪binary=True, min_df=min_df)),
            ("count", vectorizer(stop_words=stop_words, min_df=min_df))
        ])),
    ]

    if use_t2pi:
        models.append(('t2pi_transformer', T2PITransformer()))

    models.append(('classifier', classifier))
    return Pipeline(models)
```

```python
class CleanTextTransformer(TransformerMixin):
    def __init__(self, lemmatizer):
        self._lemmatizer = lemmatizer

    def fit(self, X, y=None, **fit_params):
        return self

    def transform(self, X, y=None, **fit_params):
        return np.vectorize(lambda x: clean_text(x, self._lemmatizer))(X)

    def __str__(self):
        return "CleanTextTransformer()"

    def __repr__(self):
        return self.__str__()


class T2PITransformer(TransformerMixin):
    @staticmethod
    def _max_weight(x, pbar, word_word_pr):
        pbar.update(1)
        return (word_word_pr.T * x).max(0)

    @staticmethod
    def _sum_weight(x, pbar, word_word_pr):
        pbar.update(1)
        return (word_word_pr.T * x).sum(0)

    def fit(self, X, y=None, **fit_params):
        X = X[:, :int(X.shape[1]/2)].toarray()

        print("creating term-term co-occurence pr matrix")
        terms = np.arange(X.shape[1])

        X = pd.DataFrame(X, columns=terms)
        self.word_word_pr_distr = pd.DataFrame(data=0.0, columns=terms,␣
↪index=terms)

        for term in tqdm(terms):
            self.word_word_pr_distr[term] = X[X[term] > 0].sum(0) / X.sum(0)

        return self

    def transform(self, X, y=None, **fit_params):
        X = X[:, int(X.shape[1]/2):].toarray()
        X = pd.DataFrame(X, columns=self.word_word_pr_distr.columns)
```

```python
        print("transforming ...")
        with tqdm(total=X.shape[0]) as pbar:
            X = X.apply(self._max_weight, axis=1, args=(pbar, self.
    ↪word_word_pr_distr))

        return X

    def __str__(self):
        return "T2PITransformer()"

    def __repr__(self):
        return self.__str__()
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\christian\AppData\Roaming\nltk_data…
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\christian\AppData\Roaming\nltk_data…
[nltk_data]   Package wordnet is already up-to-date!
```

1 Load Data

```python
[45]: # total number of samples needed
      randomize = False

      # retrieve dataset
      categories = ['rec.autos', 'talk.politics.mideast', 'alt.atheism', 'sci.space']

      all_docs = fetch_20newsgroups(subset='train', shuffle=randomize,
       ↪remove=('headers', 'footers', 'quotes'), categories=categories)
      categories = all_docs.target_names
```

```python
[46]: print(all_docs.data[0])
```

```
I think that domestication will change behavior to a large degree.
Domesticated animals exhibit behaviors not found in the wild.  I
don't think that they can be viewed as good representatives of the
wild animal kingdom, since they have been bred for thousands of years
to produce certain behaviors, etc.
```

1.0.1 Create Dataframe

```
[47]: data = pd.DataFrame(
          data={
              "text":all_docs.data,
              "label":all_docs.target
          }
      )

      data.head()
```

```
[47]:                                            text   label
      0  \n\nI think that domestication will change beh…      0
      1  \nI don't like this comment about "Typical" th…      3
      2  \n<apparently you're not a woman - my husband …      1
      3  While not exactly a service incident, I had a …      1
      4  \n\nI think I can. Largely as a result of effo…      2
```

1.0.2 Label Frequency

```
[48]: print(data["label"].value_counts())
      print()

      barlist = plt.bar(categories, data["label"].value_counts())

      plt.title("Frequency of documents")
      plt.xticks(categories, list(map(lambda x: x.split(".")[1], categories)))
      plt.ylabel('Number of documents')
      plt.xlabel('Sentiment expressed in Reviews')

      barlist[0].set_color('red')
      barlist[1].set_color('green')
      barlist[2].set_color('blue')
      barlist[3].set_color('grey')
      plt.show()
```

```
1    594
2    593
3    564
0    480
Name: label, dtype: int64
```

Frequency of documents
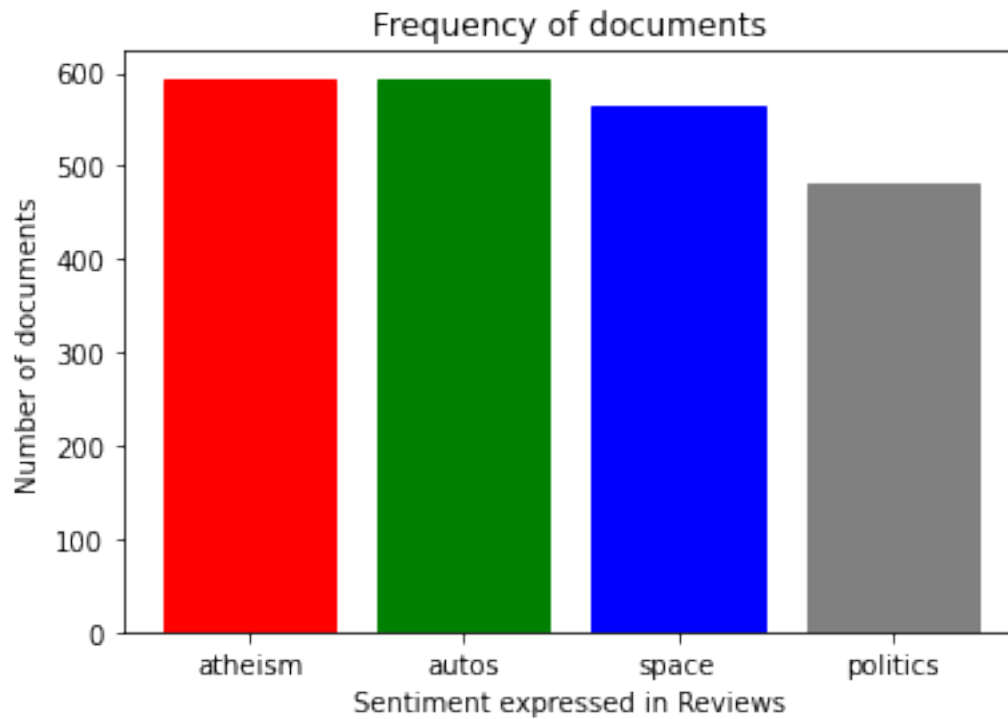
The Dataset labels needs to be balanced

1.0.3 Parameters

```
[49]: min_df = 1
      stop_words = "english"

      def get_classifier():
      #     return GaussianNB()
          return MultinomialNB()

      def get_lemmatizer():
      #     return lambda x: x
          return WordNetLemmatizer().lemmatize
```

2 Select Valid Data

```
[50]: max_size_per_class = 150

      # remove long text
      indices = data["text"].apply(data_isvalid, args=(lambda x: clean_text(x,␣
       ↪get_lemmatizer()), 128, 512))
      data = data[indices]
```

```
# make classes balanced
class_indices = []

for index in range(4):
    class_indices.append(np.where((data["label"] == index))[0])

size_per_class = min(max_size_per_class, min(map(len, class_indices)))
indices = np.concatenate([class_ids[:size_per_class] for class_ids in␣
 ↪class_indices])

data = data.iloc[indices]

data.head()
```

[50]:   text label
    0   \n\nI think that domestication will change beh…   0
    19   \n\n\tI agree, we spend too much energy on the…   0
    30   \n[rest deleted…]\n\nYou were a liberal arts…   0
    36   \nWorse? Maybe not, but it is definately a vi…   0
    50   \n\n Could you explain what any of the above p…   0

[51]: ```
print(data.iloc[0]["text"])
```

    I think that domestication will change behavior to a large degree.
    Domesticated animals exhibit behaviors not found in the wild.  I
    don't think that they can be viewed as good representatives of the
    wild animal kingdom, since they have been bred for thousands of years
    to produce certain behaviors, etc.

[52]: ```
print(data["label"].value_counts())
print()

barlist = plt.bar(categories, data["label"].value_counts())

plt.title("Frequency of documents")
plt.xticks(categories, list(map(lambda x: x.split(".")[1], categories)))
plt.ylabel('Number of documents')
plt.xlabel('Sentiment expressed in Reviews')

barlist[0].set_color('red')
barlist[1].set_color('green')
barlist[2].set_color('blue')
barlist[3].set_color('grey')
plt.show()
```

```
3    150
2    150
1    150
0    150
Name: label, dtype: int64
```
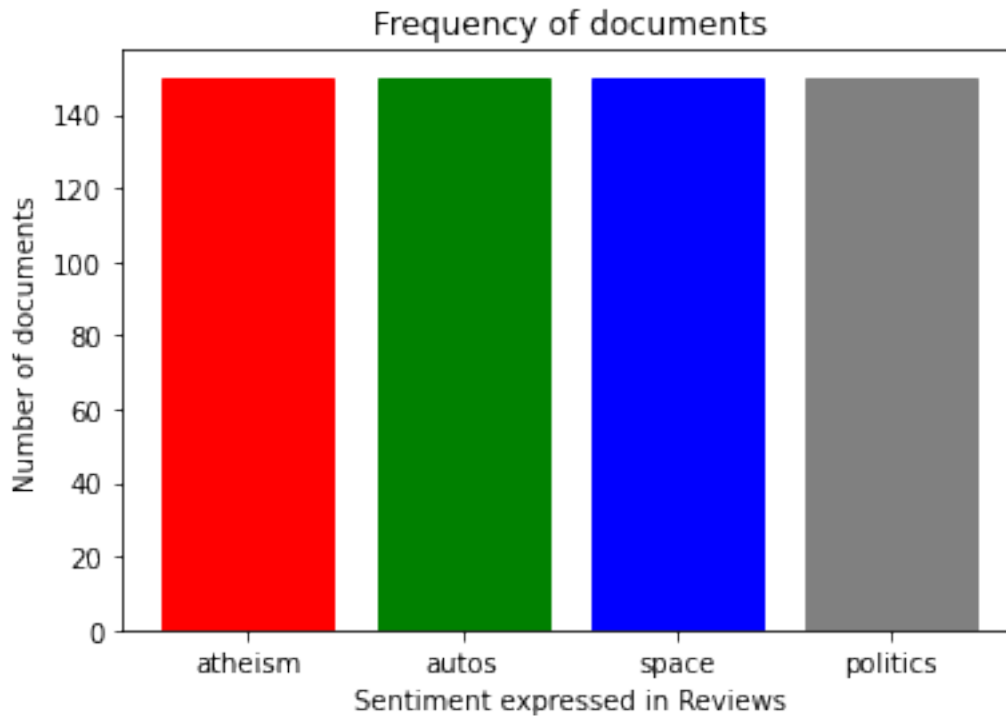

2.0.1 initialize input and output

```
[53]: X = data["text"]
      y = data['label']

      X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25,␣
       ↪random_state=42)
```

2.0.2 initialize recursive word infer model

```
[54]: # initialize model
      t2pi_model = get_pipeline("count", get_classifier(), use_t2pi=True,␣
       ↪min_df=min_df, stop_words=None, lemmatizer = get_lemmatizer())
      t2pi_model
```

```
[54]: Pipeline(steps=[('clean_text', CleanTextTransformer()),
                ('vectorizers',
                 FeatureUnion(transformer_list=[('count_binary',
                                                 CountVectorizer(binary=True)),
                                                ('count', CountVectorizer())])),
                ('t2pi_transformer', T2PITransformer()),
                ('classifier', MultinomialNB())])
```

```
[55]: # fit model
      t2pi_model.fit(X_train, y_train)
```

creating term-term co-occurence pr matrix

HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=4881.0),␣
 ↪HTML(value='')))

transforming …

HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=450.0),␣
 ↪HTML(value='')))


```
[55]: Pipeline(steps=[('clean_text', CleanTextTransformer()),
                ('vectorizers',
                 FeatureUnion(transformer_list=[('count_binary',
                                                 CountVectorizer(binary=True)),
                                                ('count', CountVectorizer())])),
                ('t2pi_transformer', T2PITransformer()),
                ('classifier', MultinomialNB())])
```

```
[56]: y_pred = t2pi_model.predict(X_test) #predict testing data

      print(classification_report(y_test, y_pred))
```

transforming …

HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=150.0),␣
 ↪HTML(value='')))

	precision	recall	f1-score	support
0	0.73	0.73	0.73	45
1	0.74	0.78	0.76	37
2	0.84	0.66	0.74	32
3	0.71	0.81	0.75	36
accuracy			0.75	150

```
      macro avg       0.76      0.74      0.75       150
   weighted avg       0.75      0.75      0.75       150
```

2.0.3 Initialize models

```python
[57]: # normal model
      count_model = get_pipeline("count", get_classifier(), use_t2pi=False,
       →min_df=min_df, stop_words=None, lemmatizer = get_lemmatizer())
      count_sw_model = get_pipeline("count", get_classifier(), use_t2pi=False,
       →min_df=min_df, stop_words=stop_words, lemmatizer = get_lemmatizer())

      tfidf_model = get_pipeline("tfidf", get_classifier(), use_t2pi=False,
       →min_df=min_df, stop_words=None, lemmatizer = get_lemmatizer())
      tfidf_sw_model = get_pipeline("tfidf", get_classifier(), use_t2pi=False,
       →min_df=min_df, stop_words=stop_words, lemmatizer = get_lemmatizer())

      # model
      t2pi_count_model = get_pipeline("count", get_classifier(), use_t2pi=True,
       →min_df=min_df, stop_words=None, lemmatizer = get_lemmatizer())
      t2pi_count_sw_model = get_pipeline("count", get_classifier(), use_t2pi=True,
       →min_df=min_df, stop_words=stop_words, lemmatizer = get_lemmatizer())

      t2pi_tfidf_model = get_pipeline("tfidf", get_classifier(), use_t2pi=True,
       →min_df=min_df, stop_words=None, lemmatizer = get_lemmatizer())
      t2pi_tfidf_sw_model = get_pipeline("tfidf", get_classifier(), use_t2pi=True,
       →min_df=min_df, stop_words=stop_words, lemmatizer = get_lemmatizer())

      models = {
          "count_model": count_model,
          "count_sw_model": count_model,
          "tfidf_model": tfidf_model,
          "tfidf_sw_model": tfidf_sw_model,
          "t2pi_count_model": t2pi_count_model,
          "t2pi_count_sw_model": t2pi_count_sw_model,
          "t2pi_tfidf_model": t2pi_tfidf_model,
          "t2pi_tfidf_sw_model": t2pi_tfidf_sw_model
      }
```

2.0.4 Running Cross validation on all Models

```python
[58]: split_size = 7
      skf = StratifiedKFold(n_splits=split_size, shuffle=True, random_state=100)

      index = 0
      macro_f1_scores, weighted_f1_scores, accuracies = [], [], []
```

```python
for train_index, test_index in skf.split(X, y):
    index += 1

    x_train_fold, x_test_fold = X.iloc[train_index], X.iloc[test_index]
    y_train_fold, y_test_fold = y.iloc[train_index], y.iloc[test_index]

    accuracies.append([])
    macro_f1_scores.append([])
    weighted_f1_scores.append([])

    for model_name, model in models.items():
        print(f'-> {index}. {model_name} \n{"="*100}\n')
        model.fit(x_train_fold, y_train_fold)
        y_pred = model.predict(x_test_fold)

        accuracy = accuracy_score(y_test_fold, y_pred)
        weighted_f1_score = calculate_f1_score(y_test_fold, y_pred,
 ↪average='weighted')
        macro_f1_score = calculate_f1_score(y_test_fold, y_pred,
 ↪average='macro')

        weighted_f1_scores[-1].append(weighted_f1_score)
        macro_f1_scores[-1].append(macro_f1_score)
        accuracies[-1].append(accuracy)
```

```
-> 1. count_model
================================================================================
===================

-> 1. count_sw_model
================================================================================
===================

-> 1. tfidf_model
================================================================================
===================

-> 1. tfidf_sw_model
================================================================================
===================

-> 1. t2pi_count_model
================================================================================
===================

creating term-term co-occurence pr matrix
```

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=5292.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=514.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=86.0),␣
 ↪HTML(value='')))
```

-> 1. t2pi_count_sw_model
==
====================

creating term-term co-occurence pr matrix

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=5034.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=514.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=86.0),␣
 ↪HTML(value='')))
```

-> 1. t2pi_tfidf_model
==
====================

creating term-term co-occurence pr matrix

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=5292.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=514.0),␣
 ↪HTML(value='')))
```

```
transforming …

HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=86.0),␣
 ↪HTML(value='')))


-> 1. t2pi_tfidf_sw_model
================================================================================
===================

creating term-term co-occurence pr matrix

HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=5034.0),␣
 ↪HTML(value='')))


transforming …

HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=514.0),␣
 ↪HTML(value='')))


transforming …

HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=86.0),␣
 ↪HTML(value='')))


-> 2. count_model
================================================================================
===================

-> 2. count_sw_model
================================================================================
===================

-> 2. tfidf_model
================================================================================
===================

-> 2. tfidf_sw_model
================================================================================
===================

-> 2. t2pi_count_model
================================================================================
===================

creating term-term co-occurence pr matrix
```

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=5356.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=514.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=86.0),␣
 ↪HTML(value='')))
```

```
-> 2. t2pi_count_sw_model
================================================================================
====================
```

creating term-term co-occurence pr matrix

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=5094.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=514.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=86.0),␣
 ↪HTML(value='')))
```

```
-> 2. t2pi_tfidf_model
================================================================================
====================
```

creating term-term co-occurence pr matrix

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=5356.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=514.0),␣
 ↪HTML(value='')))
```

```
transforming …

HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=86.0),␣
 ↪HTML(value='')))


-> 2. t2pi_tfidf_sw_model
================================================================================
===================

creating term-term co-occurence pr matrix

HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=5094.0),␣
 ↪HTML(value='')))


transforming …

HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=514.0),␣
 ↪HTML(value='')))


transforming …

HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=86.0),␣
 ↪HTML(value='')))


-> 3. count_model
================================================================================
===================

-> 3. count_sw_model
================================================================================
===================

-> 3. tfidf_model
================================================================================
===================

-> 3. tfidf_sw_model
================================================================================
===================

-> 3. t2pi_count_model
================================================================================
===================

creating term-term co-occurence pr matrix
```

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=5310.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=514.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=86.0),␣
 ↪HTML(value='')))
```

-> 3. t2pi_count_sw_model
==
====================

creating term-term co-occurence pr matrix

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=5051.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=514.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=86.0),␣
 ↪HTML(value='')))
```

-> 3. t2pi_tfidf_model
==
====================

creating term-term co-occurence pr matrix

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=5310.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=514.0),␣
 ↪HTML(value='')))
```

```
transforming …

HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=86.0),␣
 ↪HTML(value='')))


-> 3. t2pi_tfidf_sw_model
================================================================================
====================

creating term-term co-occurence pr matrix

HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=5051.0),␣
 ↪HTML(value='')))


transforming …

HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=514.0),␣
 ↪HTML(value='')))


transforming …

HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=86.0),␣
 ↪HTML(value='')))


-> 4. count_model
================================================================================
====================

-> 4. count_sw_model
================================================================================
====================

-> 4. tfidf_model
================================================================================
====================

-> 4. tfidf_sw_model
================================================================================
====================

-> 4. t2pi_count_model
================================================================================
====================

creating term-term co-occurence pr matrix
```

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=5268.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=514.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=86.0),␣
 ↪HTML(value='')))
```

```
-> 4. t2pi_count_sw_model
================================================================================
====================
```

creating term-term co-occurence pr matrix

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=5009.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=514.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=86.0),␣
 ↪HTML(value='')))
```

```
-> 4. t2pi_tfidf_model
================================================================================
====================
```

creating term-term co-occurence pr matrix

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=5268.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=514.0),␣
 ↪HTML(value='')))
```

```
transforming …

HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=86.0),␣
 ↪HTML(value='')))


-> 4. t2pi_tfidf_sw_model
================================================================================
===================

creating term-term co-occurence pr matrix

HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=5009.0),␣
 ↪HTML(value='')))


transforming …

HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=514.0),␣
 ↪HTML(value='')))


transforming …

HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=86.0),␣
 ↪HTML(value='')))


-> 5. count_model
================================================================================
===================

-> 5. count_sw_model
================================================================================
===================

-> 5. tfidf_model
================================================================================
===================

-> 5. tfidf_sw_model
================================================================================
===================

-> 5. t2pi_count_model
================================================================================
===================

creating term-term co-occurence pr matrix
```

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=5313.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=514.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=86.0),␣
 ↪HTML(value='')))
```

```
-> 5. t2pi_count_sw_model
================================================================================
===================
```

creating term-term co-occurence pr matrix

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=5054.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=514.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=86.0),␣
 ↪HTML(value='')))
```

```
-> 5. t2pi_tfidf_model
================================================================================
===================
```

creating term-term co-occurence pr matrix

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=5313.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=514.0),␣
 ↪HTML(value='')))
```

```
transforming …

HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=86.0),␣
 ↪HTML(value='')))


-> 5. t2pi_tfidf_sw_model
================================================================================
===================

creating term-term co-occurence pr matrix

HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=5054.0),␣
 ↪HTML(value='')))


transforming …

HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=514.0),␣
 ↪HTML(value='')))


transforming …

HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=86.0),␣
 ↪HTML(value='')))


-> 6. count_model
================================================================================
===================

-> 6. count_sw_model
================================================================================
===================

-> 6. tfidf_model
================================================================================
===================

-> 6. tfidf_sw_model
================================================================================
===================

-> 6. t2pi_count_model
================================================================================
===================

creating term-term co-occurence pr matrix
```

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=5290.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=515.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=85.0),␣
 ↪HTML(value='')))
```

```
-> 6. t2pi_count_sw_model
================================================================================
====================
```

creating term-term co-occurence pr matrix

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=5028.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=515.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=85.0),␣
 ↪HTML(value='')))
```

```
-> 6. t2pi_tfidf_model
================================================================================
====================
```

creating term-term co-occurence pr matrix

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=5290.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=515.0),␣
 ↪HTML(value='')))
```

```
transforming …

HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=85.0),␣
 ↪HTML(value='')))


-> 6. t2pi_tfidf_sw_model
================================================================================
===================

creating term-term co-occurence pr matrix

HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=5028.0),␣
 ↪HTML(value='')))


transforming …

HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=515.0),␣
 ↪HTML(value='')))


transforming …

HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=85.0),␣
 ↪HTML(value='')))


-> 7. count_model
================================================================================
===================

-> 7. count_sw_model
================================================================================
===================

-> 7. tfidf_model
================================================================================
===================

-> 7. tfidf_sw_model
================================================================================
===================

-> 7. t2pi_count_model
================================================================================
===================

creating term-term co-occurence pr matrix
```

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=5315.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=515.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=85.0),␣
 ↪HTML(value='')))
```

-> 7. t2pi_count_sw_model
==
====================

creating term-term co-occurence pr matrix

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=5054.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=515.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=85.0),␣
 ↪HTML(value='')))
```

-> 7. t2pi_tfidf_model
==
====================

creating term-term co-occurence pr matrix

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=5315.0),␣
 ↪HTML(value='')))
```

transforming …

```
HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=515.0),␣
 ↪HTML(value='')))
```

```

```
transforming …

HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=85.0),␣
 ↪HTML(value='')))

-> 7. t2pi_tfidf_sw_model
==
==================

creating term-term co-occurence pr matrix

HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=5054.0),␣
 ↪HTML(value='')))

transforming …

HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=515.0),␣
 ↪HTML(value='')))

transforming …

HBox(children=(HTML(value=''), FloatProgress(value=0.0, max=85.0),␣
 ↪HTML(value='')))
```

```python
[59]: model_names = list(models.keys())

      accuracy = pd.DataFrame(data=np.array(accuracies), columns=model_names)
      weighted_f1_score = pd.DataFrame(data=np.array(weighted_f1_scores),␣
       ↪columns=model_names)
      macro_f1_score = pd.DataFrame(data=np.array(macro_f1_scores),␣
       ↪columns=model_names)

      accuracy.loc["mean"] = accuracy.mean(0)
      weighted_f1_score.loc["mean"] = weighted_f1_score.mean(0)
      macro_f1_score.loc["mean"] = macro_f1_score.mean(0)
```

```python
[60]: accuracy.head(split_size+1)
```

```
[60]:      count_model   count_sw_model   tfidf_model   tfidf_sw_model  \
      0       0.825581         0.825581      0.813953         0.848837
      1       0.895349         0.895349      0.872093         0.837209
      2       0.779070         0.779070      0.779070         0.779070
      3       0.767442         0.767442      0.767442         0.779070
      4       0.709302         0.709302      0.744186         0.802326
      5       0.752941         0.752941      0.752941         0.800000
```

```
6        0.705882          0.705882          0.741176          0.741176
mean     0.776510          0.776510          0.781552          0.798241


         t2pi_count_model  t2pi_count_sw_model  t2pi_tfidf_model  \
0            0.837209              0.825581          0.813953
1            0.883721              0.860465          0.895349
2            0.790698              0.779070          0.767442
3            0.755814              0.767442          0.755814
4            0.720930              0.802326          0.732558
5            0.776471              0.811765          0.776471
6            0.705882              0.741176          0.729412
mean         0.781532              0.798261          0.781571


         t2pi_tfidf_sw_model
0                0.837209
1                0.848837
2                0.779070
3                0.755814
4                0.802326
5                0.858824
6                0.752941
mean             0.805003
```

[61]: `weighted_f1_score.head(split_size+1)`

```
[61]:        count_model  count_sw_model  tfidf_model  tfidf_sw_model  \
0            0.829217        0.829217     0.816980        0.852813
1            0.895619        0.895619     0.872093        0.835883
2            0.773977        0.773977     0.771974        0.776706
3            0.768272        0.768272     0.769793        0.778344
4            0.702862        0.702862     0.737807        0.800336
5            0.753815        0.753815     0.753815        0.797988
6            0.702123        0.702123     0.736641        0.738003
mean         0.775126        0.775126     0.779872        0.797153


         t2pi_count_model  t2pi_count_sw_model  t2pi_tfidf_model  \
0            0.840247              0.828200          0.814432
1            0.884741              0.861251          0.896098
2            0.786789              0.774098          0.761177
3            0.755761              0.766674          0.758045
4            0.716678              0.802326          0.727169
5            0.776150              0.810528          0.776209
6            0.702123              0.738579          0.730390
mean         0.780355              0.797379          0.780503


         t2pi_tfidf_sw_model
0                0.838563
```

```
1              0.849586
2              0.774004
3              0.754738
4              0.798506
5              0.858205
6              0.749615
mean           0.803317
```

[62]: `macro_f1_score.head(split_size+1)`

[62]:
```
      count_model  count_sw_model  tfidf_model  tfidf_sw_model  \
0        0.829387        0.829387     0.817038        0.853040
1        0.895588        0.895588     0.872596        0.835450
2        0.773776        0.773776     0.771673        0.776965
3        0.767857        0.767857     0.769970        0.777284
4        0.702137        0.702137     0.736923        0.799901
5        0.753049        0.753049     0.753049        0.797747
6        0.701978        0.701978     0.736119        0.737151
mean     0.774825        0.774825     0.779624        0.796791

      t2pi_count_model  t2pi_count_sw_model  t2pi_tfidf_model  \
0             0.840469             0.828432          0.815041
1             0.884679             0.860704          0.896307
2             0.786707             0.774478          0.760971
3             0.755357             0.765336          0.757970
4             0.715714             0.801948          0.726300
5             0.775649             0.810437          0.775483
6             0.701978             0.737734          0.729928
mean          0.780079             0.797010          0.780286

      t2pi_tfidf_sw_model
0                0.838372
1                0.849076
2                0.774167
3                0.753651
4                0.797857
5                0.857901
6                0.748902
mean             0.802847
```