cb-02-szymanskir-tag_based_features_research

January 21, 2019

This notebook is devoted to the task of adding tag based features to the feature vectors of content based recommendation models.

1 Important

`make features` has to be run before running any cell in this notebook.

2 Imports

```
In [1]: import numpy as np
        import pandas as pd
        import seaborn as sns
```

```
In [2]: book_tags = pd.read_csv('../data/raw/book_tags.csv')
        tags_data = pd.read_csv('../data/raw/tags.csv')
        with open("../data/external/genres.txt") as file:
            goodreads_genres = [line.rstrip('\n') for line in file]
```

3 Visualization settings

```
In [3]: sns.set(context='paper', font_scale=1.2, style='ticks', palette='muted',
                rc={"axes.labelsize":16, "ytick.labelsize": 14, "xtick.labelsize":14,
                    "font.family": "sans-serif"})
```

3.1 Data description

```
In [4]: book_tags.head()
```

Out[4]:

	goodreads_book_id	tag_id	count
0	1	30574	167697
1	1	11305	37174
2	1	11557	34173
3	1	8717	12986
4	1	33114	12716

The data contains information about what tags were assigned to a specific book and how many times was it assigned - the `count` column in the above presented data frame.

```
In [5]: tags_data.tag_name
```

```
Out[5]: 0                                 -
        1                              --1-
        2                             --10-
        3                             --12-
        4                            --122-
        5                            --166-
        6                             --17-
        7                             --19-
        8                              --2-
        9                            --258-
        10                             --3-
        11                            --33-
        12                             --4-
        13                             --5-
        14                            --51-
        15                             --6-
        16                            --62-
        17                             --8-
        18                            --99-
        19      --available-at-raspberrys--
        20                          -2001--
        21                         -calif--
        22                           -d-c--
        23                            -dean
        24                        -england-
        25                         -fiction
        26                       -fictional
        27                      -fictitious
        28                       -football-
        29                          -george
                              ...
        34222
        34223
        34224
        34225                 --
        34226                      -
        34227          ---
        34228
        34229                  --
        34230
        34231              --
        34232
        34233
        34234                       -
        34235
        34236
        34237
        34238                        -
```

```
34239
34240
34241
34242
34243
34244                moonplus-reader
34245                       -
34246                       -
34247                        hildrens
34248
34249
34250
34251
Name: tag_name, Length: 34252, dtype: object
```

Unfortunately, some tags are defined in other languages than english and some tags contain no specific information as for example `--5-`. That is why only tags representing genres will be kept as book features. The considered set of features is presented in the cell below

```
In [6]: goodreads_genres[:20]
```

```
Out[6]: ['10th-century',
         '11th-century',
         '12th-century',
         '13th-century',
         '14th-century',
         '15th-century',
         '16th-century',
         '17th-century',
         '1864-shenandoah-campaign',
         '18th-century',
         '1917',
         '19th-century',
         '1st-grade',
         '20th-century',
         '21st-century',
         '2nd-grade',
         '40k',
         'abandoned',
         'abuse',
         'academia']
```

```
In [7]: len(goodreads_genres)
```

```
Out[7]: 1228
```

4 Usage of tags

```
In [8]: book_tags = book_tags[(book_tags['count'] > 0)]
```

```
In [9]: tag_usage = book_tags[['tag_id', 'count']].groupby(by='tag_id').agg(sum).reset_index()
```

```
In [10]: tag_usage['count'].describe().round()
```

```
Out[10]: count          34250.0
         mean            6098.0
         std           762731.0
         min              1.0
         25%              3.0
         50%             10.0
         75%             52.0
         max      140718761.0
         Name: count, dtype: float64
```

4.1 How to represent tags as features?

The question is how those tags should be converted to features. The following ideas are considered:

- append tags counts to existing feature vectors
- normalize the tags count in order to measure 'how much fictional' is the considered book

The problem of the first approach is that one book might have been assigned a 100 times and another one a 1000 times. For example the first one got the `comic-book` tag assigned a 100 times and the second one got tagged as `comic-book` 300 times. Now the first book seems like a pure `comic-book` but in terms of quantities the second book is 'more' `comic-book` than the first even though it is just partly a comic book.

The first step is to check the average amount of unique tags assigned to a single book.

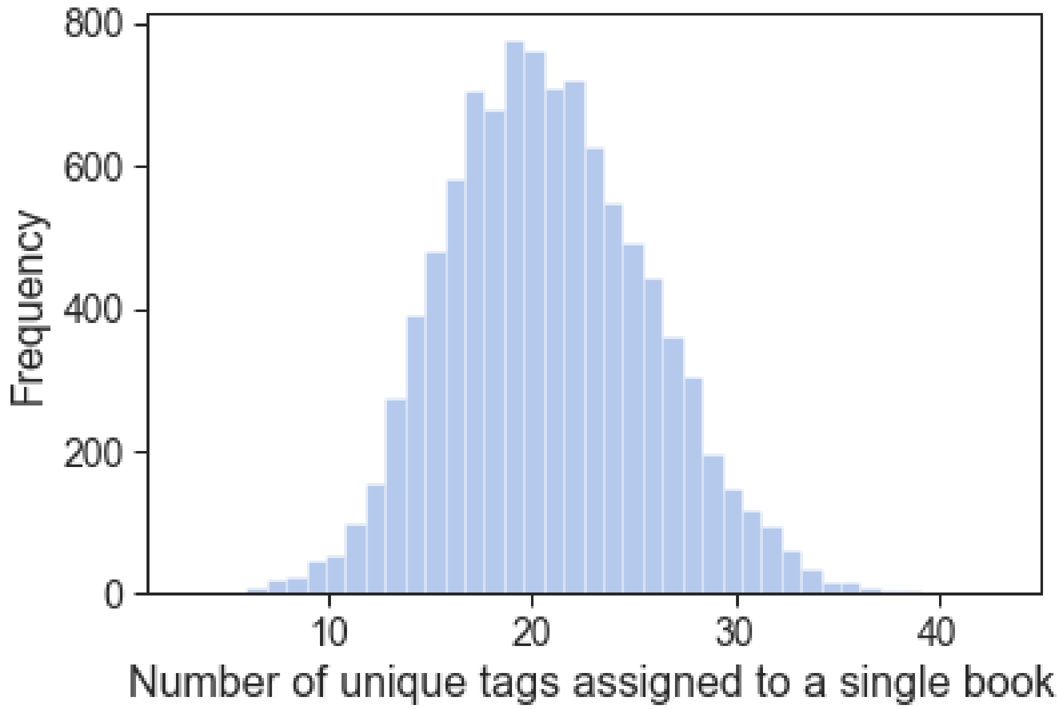
```
In [11]: book_tags_names = book_tags.merge(tags_data)
         book_tags_names = book_tags_names[book_tags_names.tag_name.isin(goodreads_genres)]
         tags_assigned_count = book_tags_names.groupby(
             'goodreads_book_id')['tag_id'].apply(np.unique).apply(len).reset_index()['tag_id']
```

```
In [12]: tags_assigned_count.describe()
```

```
Out[12]: count    10000.000000
         mean        20.712700
         std          5.206311
         min          3.000000
         25%         17.000000
         50%         20.000000
         75%         24.000000
         max         43.000000
         Name: tag_id, dtype: float64
```

```
In [13]: ax = sns.distplot(tags_assigned_count.values, kde=False, bins=len(
             np.unique(tags_assigned_count.values)))
         ax.set(xlabel='Number of unique tags assigned to a single book',
                ylabel='Frequency')
         # ax.get_figure().savefig('unique-tags-per-book.pdf', bbox_inches='tight')
```

```
Out[13]: [Text(0, 0.5, 'Frequency'),
          Text(0.5, 0, 'Number of unique tags assigned to a single book')]
```



On average a single book has 21 different tags assigned. This makes it an relevant feature as having 21 tags overall is not overspecific, but provides useful insights at the same time. Additionally, the small dimensionality allows omitting heavy computations.

4.2 Feature extraction result analysis

```
In [14]: tag_features = pd.read_csv('../features/tag_based_features.csv', index_col='book_id')
```

```
In [15]: tag_features.apply(sum, axis=1).head()
```

```
Out[15]: book_id
         1    1.0
         2    1.0
         3    1.0
         4    1.0
         5    1.0
         dtype: float64
```

```
In [16]: all(tag_features.apply(sum, axis=1).apply(round, 1) == 1)
```

```
Out[16]: True
```

All values sum up to 1 in each row which means that the tags count were normalized correctly. The reason why the sum was rounded up is because while extracting features computations were made on floating numbers which do not provide perfect accuracy.