Recommendation System

Image to Number Conversion

Encoding the Data

- One-hot Encoding

- TF-IDF Encoding

TF-IDF Encoding

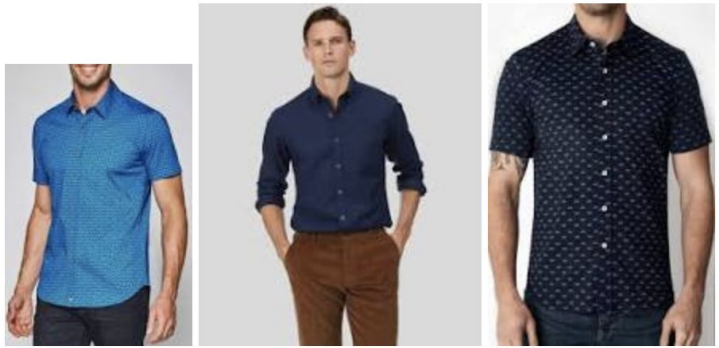
- *Term Frequency–Inverse Document Frequency* encoding
- Weigh a term according to the importance of the term within the document based on the number of times it appears in the document.
- TF (term frequency): the number of times it appears in a document.
  IDF (inverse document frequency): measurement of how significant that term is in the whole corpus.
- The more frequently the term appears, the larger its weight will be.
- Equation:

$$w_{x,y} = tf_{x,y} \times \log\left(\frac{N}{df_x}\right)$$

TF-IDF
Term x within document y

$tf_{x,y}$ = frequency of x in y
df_x = number of documents containing x
N = total number of documents

Example: TF-IDF Encoding



Step1: Describe the pictures

	id	description
0	1	blue short-sleeve shirts for man
1	2	black long-sleeve shirts for man
2	3	black short-sleeve shirts with dotted for man
3	4	blue t-shirts for woman
4	5	long-sleeve floral shirts for woman

Example: TF-IDF Encoding (Cont.)

Step 2: Use the pre-built TF-IDF vectorize in scikit-learn

Sample Code:

```
tfidf = TfidfVectorizer(analyzer='word', ngram_range=(1, 3), min_df=0, stop_words='english')
ds['description'] = ds['description'].fillna('')
tfidf_matrix = tfidf.fit_transform(ds['description'])
```

```
tfidf.get_feature_names()
```

```
['black',
 'black long',
 'black long sleeve',
 'black short',
 'black short sleeve',
 'blue',
 'blue shirts',
 'blue shirts woman',
 'blue short',
 'blue short sleeve',
 'dotted',
 'dotted man',
 'floral',
 'floral shirts',
 'floral shirts woman',
 'long',
 'long sleeve',
 'long sleeve floral',
 'long sleeve shirts',
 'man',
 'shirts',
 'shirts dotted',
 'shirts dotted man',
 'shirts man',
 'shirts woman',
 'short',
 'short sleeve',
 'short sleeve shirts',
 'sleeve',
 'sleeve floral',
 'sleeve floral shirts',
 'sleeve shirts',
 'sleeve shirts dotted',
 'sleeve shirts man',
 'woman']
```

Cosine-Similarity & Recommendation

Calculating the cosine-similarity using the TD-IDF matrix

```
array([[1.        , 0.36482714, 0.37679175, 0.15843418, 0.06641151],
       [0.36482714, 1.        , 0.22694714, 0.04003346, 0.22003889],
       [0.37679175, 0.22694714, 1.        , 0.03341733, 0.05416416],
       [0.15843418, 0.04003346, 0.03341733, 1.        , 0.24724761],
       [0.06641151, 0.22003889, 0.05416416, 0.24724761, 1.        ]])
```

Recommendation

Making recommendation by sorting the cosine-similarity

```
{1: [(0.38, 3), (0.36, 2), (0.16, 4), (0.07, 5)],
 2: [(0.36, 1), (0.23, 3), (0.22, 5), (0.04, 4)],
 3: [(0.38, 1), (0.23, 2), (0.05, 5), (0.03, 4)],
 4: [(0.25, 5), (0.16, 1), (0.04, 2), (0.03, 3)],
 5: [(0.25, 4), (0.22, 2), (0.07, 1), (0.05, 3)]}
```










Reference

https://heartbeat.fritz.ai/recommender-systems-with-python-part-i-content-based-filtering-5df4940bd831