1. **Data cleaning**

Before performing any data exploration and solution, we first clean the data by removing some abnormal values and correcting some typos and data inconsistency.

1. Remove 14 records with empty values:

E.g., #NAME? Female Fastion 繞頸上衣 Click 31/7/17

#ERROR! Female Fastion 極度乾燥外套 Impression 31/7/17

#NAME? Female Fastion 削肩泳衣 Impression 30/7/17

1. Correct typo in the sample file category label:

“Female Fastion” -> “Female Fashion”

1. Make the category labels of the test file consistent with those of the sample file:

“Female Clothes” -> “Female Fashion”

“Male Clothes” -> “Male Fashion”

1. **Solution intuition and basic data structure**

To recommend keywords to a product, we mainly consider two factors. One is the **relevance of the keyword to the product**. Because recommending a keyword “T-shirt” to a mobile phone product will be inappropriate, and a consumer searching for a mobile phone is unlikely to click a T-shirt product. We evaluate the relevance of a keyword *k* to a product *p* by calculating the **similarity** of the product *p* to all the products that used the keyword *k*. Because in the datasets, we only have the name of products, we use the classic [**cosine text similarity (using tf-idf weighting) metric**](https://www.cl.cam.ac.uk/teaching/1314/InfoRtrv/lecture4.pdf) to evaluate the similarity of the name of product *p* and the names of all the products that used keyword *k*. To calculate the cosine similarity score, we consider each keyword as a document that consists of all the product names that used that keyword. Then, a keyword document is transformed to a tf-idf weighted vector of the word terms of the document as , where

and is the frequency of a term *ti* in the document of keyword *k* . |*D*| is the total number of documents, and | is the number of documents that contains term *ti*.

We use the [jieba Chinese word segmentation package](https://github.com/fxsjy/jieba) to extract word terms from product names. The relevance score of a product *p* and a keyword *k* is evaluate as:

In order to find the keywords that are most relevant to a given product name efficiently, we build an [**inverted index**](http://www.dcs.bbk.ac.uk/~dell/teaching/cc/book/ditp/ditp_ch4.pdf) of the terms in product names to their relevant keywords (implemented in the inverted\_index.py file).

|  |  |  |
| --- | --- | --- |
| An example of the inverted index structure | | |
| word term | idf value | Related keywords as list (keyword, tf value) |
| 短袖  T恤  背心  蕾丝  简约  … … | 1.2  3.3  0.9  2.5  4.2  … … | (无袖,10), (闺蜜,3), (原宿风,9)  (一字领,22), (睡衣,18)  (无袖,18), (棉花糖,3), (多件,3),(迪士尼,3)  (无袖,20)  (上衣,23), (七分袖,11), (直筒裤,7)  … … |

The other factor that we consider is the **likelihood that a keyword can result in click events**. By exploring the sample data, I find that the most frequently used keywords do not necessarily cause more clicks. As shown in the figure below, the click rate and the keyword frequency are negatively correlated.

|  |  |  |
| --- | --- | --- |
|  |  |  |
| Female fashion category | Male fashion category | Moile&Gadgets category |

Therefore, the keyword recommendation needs to favor the less frequently used keywords that have high **click rate**. The click rate of a keyword *k* is calculated as:

In order to obtain the keyword frequency and click rate information quickly, we implement an index of keywords to their frequency and click rate (implemented in the keyword\_index.py file).

|  |  |  |
| --- | --- | --- |
| An example of the keyword frequency index structure | | |
| keyword | impression frequency | click frequency |
| 无袖  闺蜜  原宿风  背心  迪士尼  … … | 20  18  15  30  8  … … | 2  10  16  3  4  … … |

Based on these intuitions, we design two recommendation algorithms, which take both the relevance score and click rate score into consideration when recommending keywords to a product. The details of the proposed methods are introduced in the following sections.

1. **IR-based keyword recommendation**

The IR-based method recommends keywords to a product based on a composite score of the relevance score and click rate score. To be more specifically, for a product *p* and a keyword *k*, the composite score is calculated as follows:

where is the cosine similarity of the name of product *p* and all the product names that used keyword *k*. is the click rate of the keyword *k*. The parameter is a weight which takes value in the range (0,1) indicating how much the recommendation algorithm prefer the high-click rate keywords. The IR-based algorithm returns top-k keywords with the highest composite score.

The advantage of the IR-based method is that it can dynamically adapt to new data. When new log data come, it just need to insert the new records to the inverted index and keyword index. It will not affect the recommendation process. The disadvantage is that users need to tune the parameter based on intuition.

* 1. **Testing performance and parameter setting**

In order to evaluate the performance of the IR-based method, we propose to use the recall-based metric. To be more specific, we focus on a group of products and keywords that result in click event, and see if the recommendation of the algorithm contains the used keywords. We define the **recall metric** as the percentage of the products that the recommendation actually contains the used keywords. We also define a **weighted recall metric**, which is similar to the recall metric but the keywords are weighted by their click rate. This metric will evaluate how likely the algorithm will recommend the keywords with high click rate.

We perform 10-fold cross validation based on the click events of the sample data, and report the average recall and average weighted recall result in the figures below. Note that the algorithm is ran and tested for each category. From figure 1, we find the average recall value actually reduce with α parameter increase, but the average weighted recall value reaches the maximum when α=0.2. This may indicate 0.2 is a good choice of the α parameter. Figure 2 shows the recall metrics increase roughly linearly with the number of recommended keywords returned.

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
| Figure 1. Average recall and average weighted recall v.s. the α parameter | |

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
| Figure 2. Average recall and average weighted recall v.s. the top-k parameter | |

1. **ML-based keyword recommendation**

The IR-based method assumes that the recommended keyword is determined by its relevance score to the product name and its click rate, and coefficients (i.e., ) of the two factors are set based on intuition. In this project, we also tried an alternate machine learning based method. This method utilizes a logistic regression model to predict whether a click event will happen for a given product name and query keyword. The logistic regression model takes the relevance score of the product name and query keyword, the impression frequency of the keyword, the click frequency of the keyword, as well as the click rate as features, and automatically learn the coefficient of each individual features based on the sample data. Here we use the logistic regression model because the model can output the click probability for a given product name and keyword. The algorithm will recommend the top-k keywords with the highest click probability for a given product name.

The advantage of the ML-based method is that it automatically learn the coefficient of each feature to predict the click event. The disadvantage is that when many new data come, it may need to re-train the model.

**4.1 Testing performance and parameter setting**

In order to evaluate the performance of the proposed ML-based method, we proposed to use the recall and precision-based metrics. To be more specifically, we take a group of products and the corresponding query keywords and see how accurately the model can predict whether there will be a click event. We use the **F-score** as the evaluation metric.

We perform 10-fold cross validation based on the sample data, and report the F-score result in the figures below. Note that the algorithm is ran and tested for each category. From the currently result, the prediction model performance is not ideal. The precision and recall are both low. This indicates large space of improvement when more training data and/or more product features are available.

|  |  |
| --- | --- |
|  |  |
|  | |
| Figure 3. F-score vs regularization strength parameter | |