統計學習初論（105-2）

作業四

作業設計：盧信銘
國立台灣大學資管系

截止時間：2017 年 5 月 16 日上午 9 點

請至 RSAND 上批改，第一題範例命令為: sl_check_hw4q1 ./your_program，第二題範例命令為: sl_check_hw4q2 ./your_program。作業自己做。嚴禁抄襲。不接受紙本繳交，不接受遲交。請以英文或中文作答。

第一題

(50 points) The goal of sentence categorization is to assign a sentence to one of two or more predefined categories. The underlying problem is to detect sentences that mentioned about the effort of introducing new products or services in current fiscal year. We denoted this category as "S.PRODUCT."

The hw4ds1.rdata file contains a data frame named hw4ds1. The first column of hw4ds1 is annotated S.PRODUCT. The feature values (unigrams) are in the remaining columns. Each row in hw4ds1 represents a sentence. All feature values in columns 2 and beyond are term-frequency inverse document frequency (TF-IDF) values. A value of zero indicate the corresponding unigram did not appear in the sentence. A value larger than zero indicates that that particular unigram appeared in the sentence.

Also, there is another variable (list) named folds that specify how data should be partitioned for a five-fold cross validation. The variable folds is a list of five elements. Each element contains the row IDs in that particular fold.

Write a function named filter_chisq to perform feature selection based on Chi-squared statistics. This function should take the following input parameters (in this order):

1. dstrain: The training dataset for feature selection. The first column should be the outcome of the classification problem.
2. ypos: The label for a positive case in the first column. The default value is "pos."
3. min_count: The minimal count for a feature that can be considered in feature selection. A feature that has a count less than or equal to min_count should be excluded in the feature selection process. The default value is 5.

4. chi_threshold: The minimal value of chi squared statistics for a feature that should be returned. The default value is 10^{-5}.

The function should conduct feature selection based on the chi-square statistic. There are several variations in the literature about how the chi-square statistics should be computed. You should follow our discussion in slides so that your results are consistent with the correct answers. The feature value in columns 2 and beyond should be converted to binary values according to whether the original value is zero or not.

After processing all features in the dataset, you should return features with a Chi-square statistics larger than chi_threshold. The returned results should be sorted according to the chi-squared statistics in decreasing order.

The returned list should include the following components (in this order):

- colpos: the position of useful features, ordered by the chi-squared statistics from large to small.
- colname: the column names of useful features following the same order as the those first component.
- chistat: the Chi-squared statistics of useful features following the same order as the those in the first component.

Note that if all features have a Chi-squared statistics smaller than chi_threshold, you should still return a list containing above component. Each of the returned component should be NULL.

Sample input and output

```
#Sample 1
> load('hw4ds1.rdata')
> testfold = 1
> dstrain1 = hw4ds1[-folds[[testfold]],]
> out1=filter_chisq(dstrain1)
> print(head(out1$colpos, n=15))
 [1] 4111 4061 3150 4032 3510 2985 3783 3806 3416 2953 3937 1497 2999
3503 3636
> print(head(out1$colname, n=15))
 [1] "w4110_product"      "w4060_new"          "w3149_launch"
 [4] "w4031_development"  "w3509_expand"       "w2984_introduce"
 [7] "w3782_effort"       "w3805_growth"       "w3415_open"
[10] "w2952_family"       "w3936_technology"   "w1496_nuvigil"
[13] "w2998_introduction" "w3502_opportunity"  "w3635_achieve"
> print(head(out1$chistat, n=15))
 [1] 267.37745 200.98655  99.28851  54.68744  38.38120  38.29460
34.25346
 [8]  33.17687  28.95239  28.56158  27.35993  24.00468  24.00468
20.85789
[15]  19.79904


#Sample 2
```

```
> load('hw4ds1.rdata')
> testfold = 2
> dstrain1 = hw4ds1[-folds[[testfold]],]
> out1=filter_chisq(dstrain1)
> print(head(out1$colpos, n=15))
 [1] 4111 4061 3150 4032 3806 2999 3783 3510 3309 2985 3503 3718 3812
3644 2953
> print(head(out1$colname, n=15))
 [1] "w4110_product"      "w4060_new"          "w3149_launch"
 [4] "w4031_development"  "w3805_growth"       "w2998_introduction"
 [7] "w3782_effort"       "w3509_expand"       "w3308_enhance"
[10] "w2984_introduce"    "w3502_opportunity"  "w3717_develop"
[13] "w3811_improve"      "w3643_competitive"  "w2952_family"
> print(head(out1$chistat, n=15))
 [1] 289.14812 197.96719  92.13969  75.87712  42.41917  35.16189
34.94212
 [8]  30.90761  30.01064  29.65948  26.26970  26.26970  24.70598
20.62009
[15]  19.83428
```

Evaluation: All credits will be given based on the correctness of 10 testing cases. Correct output in a case is worth 5 points.

第二題

(50%) Following the first question, write a function named filter_ig to perform feature selection based on information gain. This function should take the following input parameters (in this order):

1. dstrain: The training dataset for feature selection. The first column should be the outcome of the classification problem.
2. ypos: The label for a positive case in the first column. The default value is "pos."
3. min_count: The minimal count for a feature that can be considered in feature selection. A feature that has a count less than or equal to min_count should be excluded in the feature selection process. The default value is 5.
4. ig_threshold: The minimal value of information gain that should be returned. The default value is 10^{-5}.

The function should conduct feature selection based on information gain. You should follow our discussion in slides so that your results are consistent with the correct answers. The feature value in columns 2 and beyond should be converted to binary values according to whether the original value is zero or not.
After processing all features in the dataset, you should return features with an information gain value larger than ig_threshold. The returned results should be sorted according to the information gain value in decreasing order.
The returned list should include the following components (in this order):

- colpos: the position of useful features, ordered by information gain from large to small.
- colname: the column names of useful features following the same order as the those first component.
- igvalue: the information gain values of useful features following the same order as the those in the first component.

Note that if all features have a information gain values smaller than ig_threshold, you should still return a list containing above components. Each of the returned component should be NULL.

When computing the entropy, you may encounter numerical problems because of zero or near zero probability values. Consider the case of computing the entropy of binary outcomes, the count of the two classes are 0 and 5, so we have a probability of 0 and 1 for the two outcomes. However, 0*log(0) gives NaN in R because log(0) is –Inf. To avoid this problem, you should avoid computing $p_i \log p_i$ for all $p_i < 10^{-6}$, and set the result (of $p_i \log p_i$) to 0 directly.

Sample input and output:

```
#Sample 1
> load('hw4ds1.rdata')
> testfold = 1
> dstrain1 = hw4ds1[-folds[[testfold]],]
>
> out1=filter_ig(dstrain1)
> print(head(out1$colpos, n=15))
 [1] 4111 4061 3150 4032 2985 3510 3783 3806 3416 2953 3937 1497 2999
3503 3636
> print(head(out1$colname, n=15))
 [1] "w4110_product"       "w4060_new"           "w3149_launch"
 [4] "w4031_development"   "w2984_introduce"     "w3509_expand"
 [7] "w3782_effort"        "w3805_growth"        "w3415_open"
[10] "w2952_family"        "w3936_technology"    "w1496_nuvigil"
[13] "w2998_introduction" "w3502_opportunity"   "w3635_achieve"
> print(head(out1$igvalue, n=15))
 [1] 0.24983760 0.17714788 0.08660127 0.04862331 0.03489284 0.03442428
 [7] 0.03087182 0.03009023 0.02940856 0.02677397 0.02563883 0.02514924
[13] 0.02514924 0.01987746 0.01911226


#Sample 2
> load('hw4ds1.rdata')
> testfold = 2
> dstrain1 = hw4ds1[-folds[[testfold]],]
>
> out1=filter_ig(dstrain1)
> print(head(out1$colpos, n=15))
 [1] 4111 4061 3150 4032 3806 2999 3783 3309 3510 2985 3503 3718 3812
3644 4088
> print(head(out1$colname, n=15))
 [1] "w4110_product"       "w4060_new"           "w3149_launch"
 [4] "w4031_development"   "w3805_growth"        "w2998_introduction"
 [7] "w3782_effort"        "w3308_enhance"       "w3509_expand"
```

```
 [10] "w2984_introduce"    "w3502_opportunity"  "w3717_develop"
 [13] "w3811_improve"      "w3643_competitive"  "w4087_market"
> print(head(out1$igvalue, n=15))
  [1] 0.27196282 0.17223363 0.07916467 0.06483458 0.03713128 0.03425038
  [7] 0.03118124 0.02989825 0.02779846 0.02732292 0.02405568 0.02405568
 [13] 0.02326669 0.01957476 0.01951049
```

Evaluation: All credits will be given based on the correctness of 10 testing cases. Correct output in a case is worth 5 points.