ADEO-Master 2 Text Mining

Final Exam

04-april-2019

1. Building a simple spam classifier.

1.2. Use the training set to create a decision tree spam classifier.

1.3. Apply your classifier to the test set.

1.4. Give the confusion matrix and the accuracy of the classifier

Classifier Model

J48 pruned tree

------------------

$1000 = '(-inf-0.5]'

| FREE = '(-inf-0.5]'

| | To = '(-inf-0.5]'

| | | mobile = '(-inf-0.5]': 0 (184.0/17.0)

| | | mobile = '(0.5-inf)': 1 (3.0)

| | To = '(0.5-inf)': 1 (3.0)

| FREE = '(0.5-inf)': 1 (4.0)

$1000 = '(0.5-inf)': 1 (6.0)

Number of Leaves : 5

Size of the tree : 9

Time taken to build model: 0.42 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0.1 seconds

=== Summary ===

Correctly Classified Instances 183 91.5 %

Incorrectly Classified Instances 17 8.5 %

Kappa statistic 0.6112

Mean absolute error 0.1543

Root mean squared error 0.2778

Relative absolute error 55.5468 %

Root relative squared error 74.8266 %

Total Number of Instances 200

=== Detailed Accuracy By Class ===

TP Rate FP Rate Precision Recall F-Measure MCC ROC Area PRC Area Class

1.000 0.515 0.908 1.000 0.952 0.663 0.742 0.908 0

0.485 0.000 1.000 0.485 0.653 0.663 0.742 0.570 1

Weighted Avg. 0.915 0.430 0.923 0.915 0.902 0.663 0.742 0.85

=== Confusion Matrix ===

a b <-- classified as

167 0 | a = 0

17 16 | b = 1

2 Processing corpuses

Let us go back to the file spamDataTxt.csv.

1. Create a corpus with the messages considered as spam.

2. Clean this corpus.

3. Show the corresponding wordcloud.

4. Repeat the previous questions with the set of all messages.

5. Can we notice differences between the two wordclouds ? If yes, can

these differences help to give us rules characterizing spam messages ?

All messages;



On the wordcloud plot of all messages, as we can see the featured words are respectively;

“call”, ”$1000”, ”claim”, ”please”, ”receive”, ”code”,”customerselected” etc.





On the wordcloud plot of spam messages looks pretty similar with previous one. But of course there are some key differences like “free”,”replyplease”,”cash”,”mobile” etc. So while we use these values, we can determine spams from messages.

6. Some methods use TF-IDF values to build spam classifier

1. How can we use intermediate results of the previous questions to compute these values ? (you don't have to do that, you have just to explain how to do it).

*Mathematically we can express* ***TF-IDF*** *as below*

Term Frequency :

*Let* ***freq(t,d****) be the count of the instance of the term* ***t*** *in document* ***d****.*

*Let* ***TF(t,d)*** *be proportion of the count of term* ***t*** *in document* ***d.***

Inverse Document Frequency:

*Let* ***N*** *be the count distinct documents in the corpus.*

*Let* ***count(t)*** *be the count of documents in the corpus in which the terms* ***t*** *is present.*

)

The Might TF-IDF

*Combine TF and IDF to enhance document term frequency metrices:*

*In R programming language:*

# the function for calculate relative term frequency (TF)

term.frequency <- function(row) {

row / sum(row)

}

# the function for calculate inverse document frequency (IDF)

inverse.doc.freq <- function(col) {

corpus.size <- length(col)

doc.count <- length(which(col>0))

log10(corpus.size /doc.count)

}

# the function for calculate TF\_IDF

tf.idf <- function(x,idf) {

x \* idf

}

Regarding the results of the previous questions, like “free”,”replyplease”,”cash”,”mobile” etc. Suppose that we have term count tables of a corpus consisting of only two documents, as listed on the right.

The calculation of tf–idf for the term " free " is performed as follows:

**Document 1**

|  |  |
| --- | --- |
| **Terms** | **Term Count** |
| free | 1 |
| replyplease | 1 |
| cash | 1 |
| mobile | 2 |

**Document 2**

|  |  |
| --- | --- |
| **Terms** | **Term Count** |
| free | 1 |
| claim | 1 |
| please | 2 |
| code | 1 |

An idf is constant per corpus, and accounts for the ratio of documents that include the word "free". In this case, we have a corpus of two documents and all of them include the word "free".

So tf–idf is zero for the word "free", which implies that the word is not very informative as it appears in all documents.

(b) What do you think about these methods ?

TFIDF, short for term frequency–inverse document frequency, is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus[[1]](#footnote-1) Regarding the explanation, to able to find spam words at the corpus and how many times repeating itself throughout document, it is really effective methods for our case.

3 Text Clustering applied to spam detection (R)

3.1. Explain why the messages' description used in this file can give more

accurate spam classifier than those of the previous questions ?

Because each variable at the file spamDataNum.csv (X1…X48) represents the frequency of a given word in the mail rather than raw data like previous file. So we can easily analyze words frequency and understand which words have relation with each one. And it helps to build more better models and it produces more accurate result for us.

3.4. Normalize the data and apply again kMeans. How do you explain the

difference between the two accuracies ?

After the normalization of data we obtained better accuracy than un-normalized data as below:

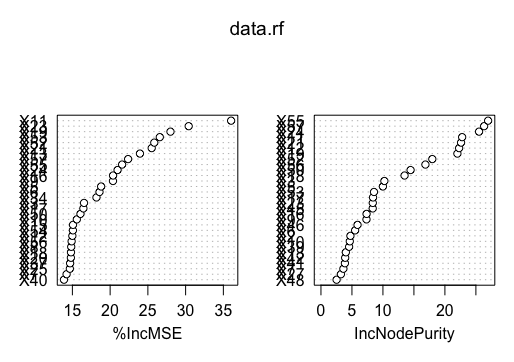
Unnormalized:

#[1] 0.6359487

Normalized:

# [1] 0.7998261

3.6 Compute and plot the importance values.



3.7. What are the most three important variables ?

%IncMSE is the most robust and informative measure. the higher number, the more important IncNodePurity relates to the loss function which by best splits are chosen. The loss function is mse for regression and gini-impurity for classification. More useful variables achieve higher increases in node purities, that is to find a split which has a high inter node 'variance' and a small intra node 'variance'. IncNodePurity is biased and should only be used if the extra computation time of calculating %IncMSE is unacceptable.

X55, X57, X56, X53 ..

1. Rajaraman, A.; Ullman, J.D. (2011). "Data Mining" (PDF). Mining of Massive Datasets. pp. 1–17. doi:10.1017/CBO9781139058452.002. ISBN 978-1-139-05845-2. [↑](#footnote-ref-1)