Assignment 2

# Step 1) Data Selection

* **SQL queries:**

CREATE TABLE TX\_A2\_INITIAL AS

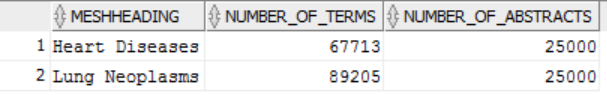
SELECT MESHHEADING, COUNT(DISTINCT TERM) AS NUMBER\_OF\_TERMS, COUNT(DISTINCT DMUSER.ML330\_SODA\_MSH.PMID) AS NUMBER\_OF\_ABSTRACTS

FROM DMUSER.ML330\_SODA\_MSH, DMUSER.ML330\_SODA\_NWD

WHERE DMUSER.ML330\_SODA\_MSH.PMID = DMUSER.ML330\_SODA\_NWD.PMID

GROUP BY MESHHEADING;

* **Result:**



For category ‘Heart Diseases’, there are 67713 distinct terms and 25000 abstracts.

For category ‘Lung Neoplasms’, there are 89205 distinct terms and 25000 abstracts.

There are some overlapping terms between the two categories.

# Step 2) Preprocessing

* How many words are there with no vocabulary changes or pruning?

SELECT COUNT(DISTINCT TERM)

FROM DMUSER.ML330\_SODA\_NWD;



Before any preprocessing, the total number of distinct terms of both categories is 126848.

* How many lower case words are there?

SELECT COUNT(DISTINCT TERM)

FROM DMUSER.ML330\_SODA\_NWD

WHERE TERM = LOWER(TERM);



There are 83383 terms in lowercase.

SELECT COUNT(DISTINCT LOWER(TERM))

FROM DMUSER.ML330\_SODA\_NWD;



The total number of distinct terms when converting to lowercase is 114881.

* How many words are there after removing stop words?

I imported stop word list from <http://www.lextek.com/manuals/onix/stopwords2.html>, and created a table named TX\_A2\_STOPWORDS. The list contains 571 words.

SELECT COUNT(DISTINCT LOWER(TERM))

FROM DMUSER.ML330\_SODA\_NWD, TX\_A2\_STOPWORDS

WHERE LOWER(TERM) = TX\_A2\_STOPWORDS.STOPWORD;



There are 490 words of the terms can be regarded as stop words. So after removing stop words, there are 114881-490=114391 distinct words in lower case.

* Select distinct terms in lowercase without stopwords

CREATE TABLE TX\_A2\_DISTINCT\_LOWER\_NOSTOP AS

SELECT DISTINCT LOWER(TERM) AS DISTINCTTERM

FROM DMUSER.ML330\_SODA\_NWD;

DELETE FROM TX\_A2\_DISTINCT\_LOWER\_NOSTOP

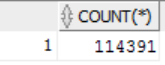
WHERE DISTINCTTERM IN (SELECT DISTINCTTERM

FROM TX\_A2\_DISTINCT\_LOWER\_NOSTOP, TX\_A2\_STOPWORDS

WHERE DISTINCTTERM = STOPWORD);

SELECT COUNT(\*)

FROM TX\_A2\_DISTINCT\_LOWER\_NOSTOP;



* Identify and remove terms that appear in few abstracts or in most abstracts

(1) Remove terms that appear less than 11 times.

CREATE TABLE TX\_A2\_DF AS

SELECT DISTINCTTERM, COUNT(DISTINCT DMUSER.ML330\_SODA\_NWD.PMID) AS ABSTRACTNUMBER

FROM TX\_A2\_DISTINCT\_LOWER\_NOSTOP, DMUSER.ML330\_SODA\_NWD

WHERE TX\_A2\_DISTINCT\_LOWER\_NOSTOP.DISTINCTTERM = LOWER(DMUSER.ML330\_SODA\_NWD.TERM)

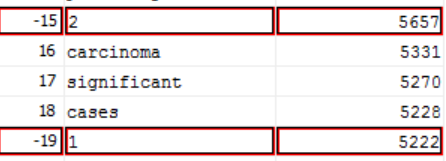
GROUP BY DISTINCTTERM

HAVING COUNT(DISTINCT DMUSER.ML330\_SODA\_NWD.PMID)>10

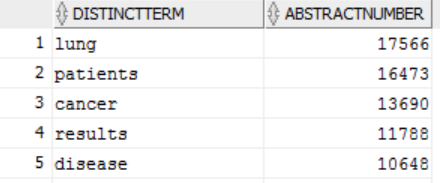
ORDER BY COUNT(DISTINCT DMUSER.ML330\_SODA\_NWD.PMID) DESC;

(2) Remove terms that are integers less than 50 and have appeared many times in different abstracts.

Remove some special meaningless words such as “’s”.

(3) The word that appears most among all the abstracts is ‘lung’. It appears 17566 times in different abstracts. However, there are 50,000 abstracts in all. So there is no need to remove words which appear so many times.



SELECT COUNT(\*)

FROM TX\_A2\_DF;



The number of terms after step 2 is 14531.

# Step 3) Transformations

**3.1) Term Frequency (tf) \* Inverse Document Frequency (idf)**

* SQL queries for computing tf\*idf:

CREATE TABLE TX\_A2\_TF AS

SELECT DISTINCTTERM, PMID, COUNT(LOWER(DMUSER.ML330\_SODA\_NWD.TERM)) AS TERMNUMBER

FROM DMUSER.ML330\_SODA\_NWD, TX\_A2\_DF

WHERE LOWER(DMUSER.ML330\_SODA\_NWD.TERM) = TX\_A2\_DF.DISTINCTTERM

GROUP BY DISTINCTTERM, PMID

ORDER BY COUNT(LOWER(DMUSER.ML330\_SODA\_NWD.TERM)) DESC;

SELECT COUNT(DISTINCT DISTINCTTERM)

FROM TX\_A2\_TF;

CREATE TABLE TX\_A2\_TFIDF AS

SELECT TX\_A2\_TF.DISTINCTTERM, PMID, TERMNUMBER \* (LOG(2,50000)-LOG(2,ABSTRACTNUMBER)+1) AS TFIDF

FROM TX\_A2\_DF, TX\_A2\_TF

WHERE TX\_A2\_TF.DISTINCTTERM = TX\_A2\_DF.DISTINCTTERM

ORDER BY TERMNUMBER \* (LOG(2,50000)-LOG(2,ABSTRACTNUMBER)+1) DESC;

SELECT COUNT(\*)

FROM TX\_A2\_TFIDF;

CREATE TABLE TX\_A2\_TFIDF\_TOP AS

SELECT DISTINCT DISTINCTTERM, MAX(TFIDF) AS MAXTFIDF

FROM TX\_A2\_TFIDF

GROUP BY DISTINCTTERM

ORDER BY MAX(TFIDF) DESC;

* Select top 100/200/500 terms with highest values of tf\*idf.

SELECT DISTINCTTERM

FROM TX\_A2\_TFIDF\_TOP

WHERE ROWNUM<=100; or 200 or 500

**3.2) Information Gain**

* Probability of Pr(t)

CREATE TABLE TX\_A2\_DF\_PR1 AS

SELECT DISTINCTTERM, ABSTRACTNUMBER/50000 AS PR1

FROM TX\_A2\_DF;

* Probability of Pr(Lung Neoplasms|t)

CREATE TABLE TX\_A2\_DF\_LUNGNUMBER AS

SELECT TX\_A2\_DF.DISTINCTTERM, COUNT(DISTINCT DMUSER.ML330\_SODA\_NWD.PMID) AS LUNGNUMBER

FROM TX\_A2\_DF, DMUSER.ML330\_SODA\_MSH, DMUSER.ML330\_SODA\_NWD

WHERE TX\_A2\_DF.DISTINCTTERM = LOWER(DMUSER.ML330\_SODA\_NWD.TERM) AND DMUSER.ML330\_SODA\_NWD.PMID = DMUSER.ML330\_SODA\_MSH.PMID

AND DMUSER.ML330\_SODA\_MSH.MESHHEADING = 'Lung Neoplasms'

GROUP BY TX\_A2\_DF.DISTINCTTERM

ORDER BY LUNGNUMBER DESC;

CREATE TABLE TX\_A2\_DF\_NUM\_PR2 AS

SELECT TX\_A2\_DF.DISTINCTTERM, TX\_A2\_DF\_LUNGNUMBER.LUNGNUMBER AS LUNGNUMBER

FROM TX\_A2\_DF\_LUNGNUMBER

RIGHT OUTER JOIN TX\_A2\_DF

ON TX\_A2\_DF.DISTINCTTERM = TX\_A2\_DF\_LUNGNUMBER.DISTINCTTERM;

* Right outer join is critical with null values for future computation of probability

CREATE TABLE TX\_A2\_DF\_PR2 AS

SELECT TX\_A2\_DF.DISTINCTTERM, LUNGNUMBER/ABSTRACTNUMBER AS PR2

FROM TX\_A2\_DF, TX\_A2\_DF\_NUM\_PR2

WHERE TX\_A2\_DF.DISTINCTTERM = TX\_A2\_DF\_NUM\_PR2.DISTINCTTERM;

* Probability of Pr(Heart Diseases|t)

CREATE TABLE TX\_A2\_DF\_HEARTNUMBER AS

SELECT TX\_A2\_DF.DISTINCTTERM, COUNT(DISTINCT DMUSER.ML330\_SODA\_NWD.PMID) AS HEARTNUMBER

FROM TX\_A2\_DF, DMUSER.ML330\_SODA\_MSH, DMUSER.ML330\_SODA\_NWD

WHERE TX\_A2\_DF.DISTINCTTERM = LOWER(DMUSER.ML330\_SODA\_NWD.TERM) AND DMUSER.ML330\_SODA\_NWD.PMID = DMUSER.ML330\_SODA\_MSH.PMID

AND DMUSER.ML330\_SODA\_MSH.MESHHEADING = 'Heart Diseases'

GROUP BY TX\_A2\_DF.DISTINCTTERM

ORDER BY HEARTNUMBER DESC;

CREATE TABLE TX\_A2\_DF\_NUM\_PR3 AS

SELECT TX\_A2\_DF.DISTINCTTERM, TX\_A2\_DF\_HEARTNUMBER.HEARTNUMBER AS HEARTNUMBER

FROM TX\_A2\_DF\_HEARTNUMBER

RIGHT OUTER JOIN TX\_A2\_DF

ON TX\_A2\_DF.DISTINCTTERM = TX\_A2\_DF\_HEARTNUMBER.DISTINCTTERM;

CREATE TABLE TX\_A2\_DF\_PR3 AS

SELECT TX\_A2\_DF.DISTINCTTERM, HEARTNUMBER/ABSTRACTNUMBER AS PR3

FROM TX\_A2\_DF, TX\_A2\_DF\_NUM\_PR3

WHERE TX\_A2\_DF.DISTINCTTERM = TX\_A2\_DF\_NUM\_PR3.DISTINCTTERM;

* Probability of Pr(not t)

CREATE TABLE TX\_A2\_DF\_PR4 AS

SELECT DISTINCTTERM, 1 - ABSTRACTNUMBER/50000 AS PR4

FROM TX\_A2\_DF;

* Probability of Pr(Lung Neoplasma|not t)

CREATE TABLE TX\_A2\_DF\_PR5 AS

SELECT TX\_A2\_DF.DISTINCTTERM, (25000-NVL(LUNGNUMBER,0))/(50000-ABSTRACTNUMBER) AS PR5

FROM TX\_A2\_DF, TX\_A2\_DF\_NUM\_PR2

WHERE TX\_A2\_DF.DISTINCTTERM = TX\_A2\_DF\_NUM\_PR2.DISTINCTTERM;

* Probability of Pr(Heart Diseases|not t)

CREATE TABLE TX\_A2\_DF\_PR6 AS

SELECT TX\_A2\_DF.DISTINCTTERM, (25000-NVL(HEARTNUMBER,0))/(50000-ABSTRACTNUMBER) AS PR6

FROM TX\_A2\_DF, TX\_A2\_DF\_NUM\_PR3

WHERE TX\_A2\_DF.DISTINCTTERM = TX\_A2\_DF\_NUM\_PR3.DISTINCTTERM;

* Compute IG of each term

CREATE TABLE TX\_A2\_IG AS

SELECT TX\_A2\_DF\_PR1.DISTINCTTERM, (PR1\*(NVL(PR2,0)\*LOG(2,NVL(PR2,1)) + NVL(PR3,0)\*LOG(2,NVL(PR3,1))) + PR4\*(PR5\*LOG(2,PR5) + PR6\*LOG(2,PR6))) AS IG

FROM TX\_A2\_DF\_PR1, TX\_A2\_DF\_PR2, TX\_A2\_DF\_PR3, TX\_A2\_DF\_PR4, TX\_A2\_DF\_PR5, TX\_A2\_DF\_PR6

WHERE TX\_A2\_DF\_PR1.DISTINCTTERM = TX\_A2\_DF\_PR2.DISTINCTTERM AND TX\_A2\_DF\_PR2.DISTINCTTERM = TX\_A2\_DF\_PR3.DISTINCTTERM

AND TX\_A2\_DF\_PR3.DISTINCTTERM = TX\_A2\_DF\_PR4.DISTINCTTERM AND TX\_A2\_DF\_PR4.DISTINCTTERM = TX\_A2\_DF\_PR5.DISTINCTTERM

AND TX\_A2\_DF\_PR5.DISTINCTTERM = TX\_A2\_DF\_PR6.DISTINCTTERM

ORDER BY IG DESC;

* Select top 100/200/500 terms with highest values of Information Gain.

SELECT DISTINCTTERM

FROM TX\_A2\_IG

WHERE ROWNUM<=100; or 200 or 500

**3.3) Create tables for classification**

Create 6 tables with 100, 200 and 500 terms for tf\*idf and for information gain respectively.

* Table1: Top 100 terms for tf\*idf

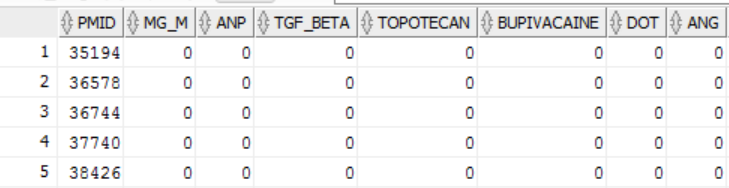
CREATE TABLE TX\_A2\_C1 AS

SELECT TX\_A2\_FEATURES1.\*, DMUSER.ML330\_SODA\_MSH.MESHHEADING

FROM TX\_A2\_FEATURES1

JOIN DMUSER.ML330\_SODA\_MSH

ON TX\_A2\_FEATURES1.PMID = DMUSER.ML330\_SODA\_MSH.PMID;



* Table2: Top 200 terms for tf\*idf

CREATE TABLE TX\_A2\_C2 AS

SELECT TX\_A2\_FEATURES2.\*, DMUSER.ML330\_SODA\_MSH.MESHHEADING

FROM TX\_A2\_FEATURES2

JOIN DMUSER.ML330\_SODA\_MSH

ON TX\_A2\_FEATURES2.PMID = DMUSER.ML330\_SODA\_MSH.PMID;

* Table3: Top 500 terms for tf\*idf

CREATE TABLE TX\_A2\_C3 AS

SELECT TX\_A2\_FEATURES3.\*, DMUSER.ML330\_SODA\_MSH.MESHHEADING

FROM TX\_A2\_FEATURES3

JOIN DMUSER.ML330\_SODA\_MSH

ON TX\_A2\_FEATURES3.PMID = DMUSER.ML330\_SODA\_MSH.PMID;

* Table4: Top 100 terms for Information Gain

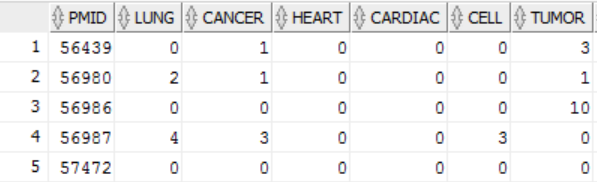
CREATE TABLE TX\_A2\_C4 AS

SELECT TX\_A2\_FEATURES4.\*, DMUSER.ML330\_SODA\_MSH.MESHHEADING

FROM TX\_A2\_FEATURES4

JOIN DMUSER.ML330\_SODA\_MSH

ON TX\_A2\_FEATURES4.PMID = DMUSER.ML330\_SODA\_MSH.PMID;



* Table5: Top 200 terms for Information Gain

CREATE TABLE TX\_A2\_C5 AS

SELECT TX\_A2\_FEATURES5.\*, DMUSER.ML330\_SODA\_MSH.MESHHEADING

FROM TX\_A2\_FEATURES5

JOIN DMUSER.ML330\_SODA\_MSH

ON TX\_A2\_FEATURES5.PMID = DMUSER.ML330\_SODA\_MSH.PMID;

* Table6: Top 500 terms for Information Gain

CREATE TABLE TX\_A2\_C6 AS

SELECT TX\_A2\_FEATURES6.\*, DMUSER.ML330\_SODA\_MSH.MESHHEADING

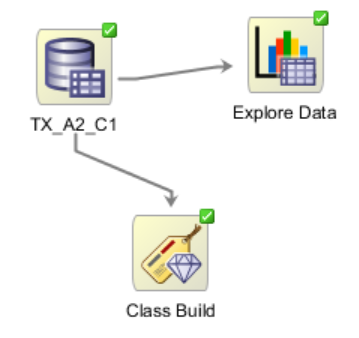
FROM TX\_A2\_FEATURES6

JOIN DMUSER.ML330\_SODA\_MSH

ON TX\_A2\_FEATURES6.PMID = DMUSER.ML330\_SODA\_MSH.PMID;

# Step 4) Data Mining

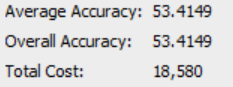
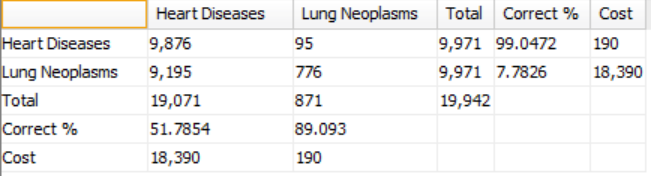
Use the oracle data miner to create two classifiers (a decision tree and naïve bayes) for the 6 tables of feature sets. The screenshot below is an example.



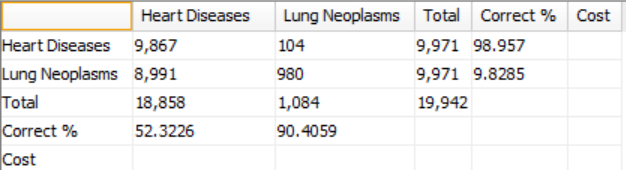
# Step 5) Interpretation

**1. Table of top 100 terms for tf\*idf**

* Accuracy and confusion matrix of decision trees

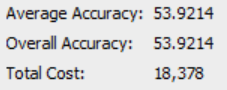
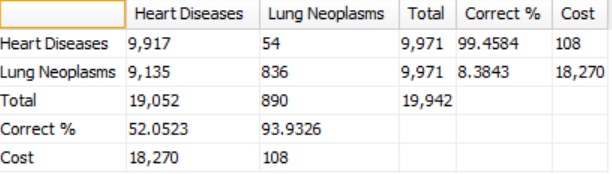
 

* Accuracy and confusion matrix of naïve bayes

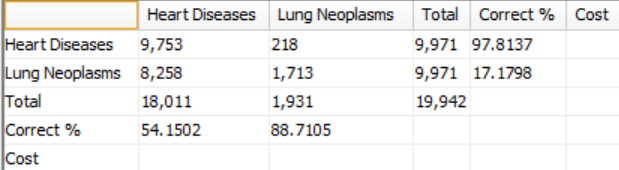
 

**2. Table of top 200 terms for tf\*idf**

* Accuracy and confusion matrix of decision trees

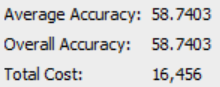
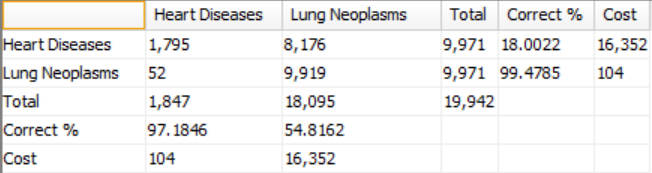
 

* Accuracy and confusion matrix of naïve bayes

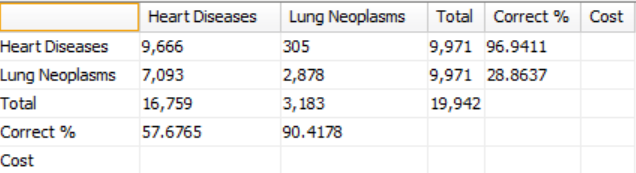
 

**3. Table of top 500 terms for tf\*idf**

* Accuracy and confusion matrix of decision trees

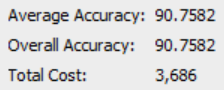
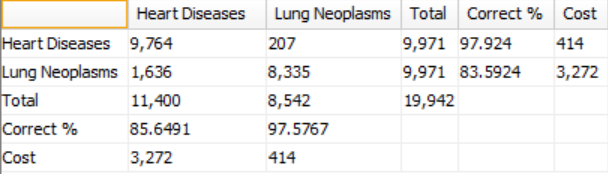
 

* Accuracy and confusion matrix of naïve bayes

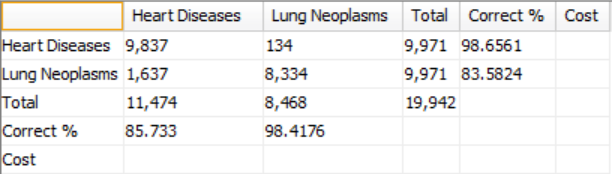
 

**4. Table of top 100 terms for Information Gain**

* Accuracy and confusion matrix of decision trees

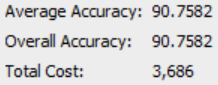
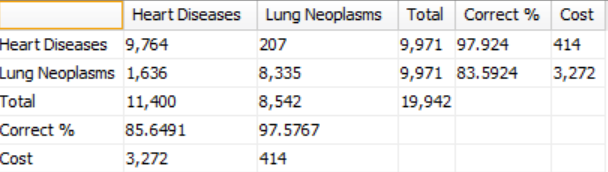
 

* Accuracy and confusion matrix of naïve bayes

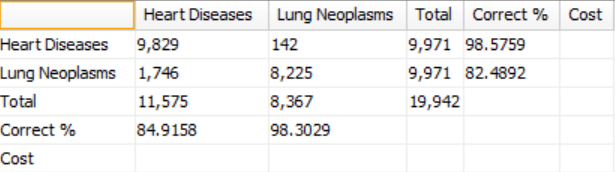
 

**5. Table of top 200 terms for Information Gain**

* Accuracy and confusion matrix of decision trees

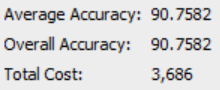
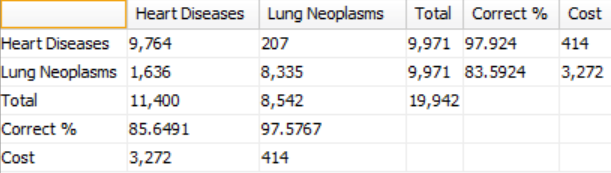
 

* Accuracy and confusion matrix of naïve bayes

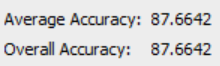
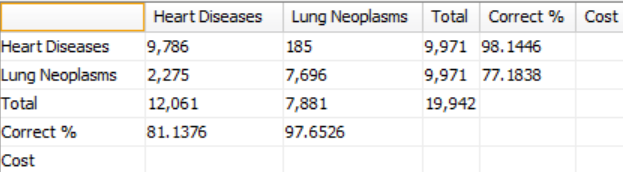
 

**6. Table of top 500 terms for Information Gain**

* Accuracy and confusion matrix of decision trees

* Accuracy and confusion matrix of naïve bayes

**7. Findings from the above results**

In general, performances of text classification of feature sets based on tf\*idf selection method are much worse than those of feature sets based on Information Gain. The overall accuracy of tf\*idf feature sets is about 50% to 65%, while the overall accuracy of Information Gain feature sets is about 90%. So information gain feature selection method is more effective for this text classification project.

Discrepancies of accuracy between decision trees and naïve bayes are not obvious for the same table (source data). For top 100/200/500 terms based on tf\*idf and top 100 terms based on information gain, the performances of naïve bayes are a little bit better than decision trees.

For tf\*idf feature sets, as the number of terms increases, the accuracy of both decision trees and naïve bayes increase. As can be seen in the confusion matrix, the accuracy of ‘Heart Diseases’ is much higher than ‘Lung Neoplasms’. However, there is a reverse for the accuracy of decision trees with 500 terms based on tf\*idf, which shows that the performance of identification of ‘Lung Neoplasms’ is much better than the other.

For information gain feature sets, as the number of terms increases, there is no difference between the accuracy of decision trees. Decision trees have the same performance on all the three tables based on information gain regardless of the number of terms. On the other hand, however, as the number of terms rises, the accuracy of naïve bayes declines for feature sets based on information gain.

As the number of terms increase, the model of decision trees become more sophisticated with more braches. After taking a look at the selected terms that played an important role for text classification, I find that top terms selected by information gain look more reasonable and anticipated, while top terms selected by tf\*idf seem surprised to me.