Semantic search engine using elastic search and Text analysis

Jayanth Prakash Kulkarni
School of Computer Science
Windsor, Ontario,Canada
kulka112@uwindsor.ca

ABSTRACT

A search engine which captures the semantic relations between research papers was developed. 3 datasets containing research papers with the title and venue of conference were considered and classification and clustering algorithms were learnt on this data and the results are discussed in detail here.

CCS CONCEPTS

• **Machine Learning** → **Classification**; • **Deep Learning** → *Doc2vec*; • **Pagerank**; • **Feature Selection** → Mutual information, Chisquared;

KEYWORDS

Classification,Kmeans,Support Vector Machines, Naive Bayes, Feature Selection

1 INTRODUCTION

Semantic search engine was developed by indexing the content of 3,079,007 research papers papers. Page rank for all the pages which had citations links were calculated. Further to find similar papers, doc2vec similarity score was used for this. To understand the working of the classical machine learning algorithms like Naive bayes, Support vector machines and logistic regression 3 data sets were analyzed in detail.

2 SEARCH ENGINE CONCEPTS

2.1 Search Engine

Search engine indexing is the process of a search engine collecting, parses and stores data for use by the search engine. The actual search engine index is the place where all the data the search engine has collected is stored. It is the search engine index that provides the results for search queries, and pages that are stored within the search engine index that appear on the search engine results page. Without a search engine index, the search engine would take considerable amounts of time and effort each time a search query was initiated, as the search engine would have to search not only every web page or piece of data that has to do with the particular keyword used in the search query, but every other piece of information it has access to, to ensure that it is not missing something that has something to do with the particular keyword. Search engine spiders, also called search engine crawlers, are how the search engine index gets its information, as well as keeping it up to date and free of spam. [1]

2.2 Document

The term document refers to a single page or item in a website search index. On a publishing website, for example, a document could be a single article.

2.3 Fields

In the world of search, the term fields refers to the various components or elements which form the schema of a document in a search engine index. Different document types have different fields.

3 RESOURCES USED

3.1 Flask

Flask is a micro web framework written in Python.It is classified as a microframework because it does not require particular tools or libraries.

3.2 Elastic Search

Elasticsearch is a search engine based on Lucene.It provides a distributed, multitenant-capable full-text search engine with an HTTP web interface and schema-free JSON documents.Elasticsearch is developed in Java and is released as open source under the terms of the Apache License. [2]

3.3 AMiner

Aminer Dataset contains citation data is extracted from DBLP, ACM, MAG (Microsoft Academic Graph), and other sources. The first version contains 629,814 papers and 632,752 citations. Each paper is associated with abstract, authors, year, venue, and title[3]

Field Name	Field Type	Description	Example
id	string	paper ID	013ea675-bb58-42f8-a423-f5534546b2b1
title	string	paper title	Prediction of consensus binding mode geometries for related chemical series of positive allosteric modulators of adenosine and muscarinic acetylcholine receptors
authors	list of strings	paper authors	["Leon A. Sakkal", "Kyle Z. Rajkowski", "Roger S. Armen"]
venue	string	paper venue	Journal of Computational Chemistry
year	int	published year	2017
n_citation	int	citation number	0
references	list of strings	citing papers' ID	["4f4f200c-0764-4fef-9718-b8bccf303dba", "aa699fbf-fabe-40e4-bd68-46eaf333f7b1"]
abtract	string	abstract	This paper studies ...

Figure 1: Aminer Dataset Structure

4 MACHINE LEARNING ALGORITHMS

4.1 Naive Bayes

The goal of any probabilistic classifier is, with features x_0 through x_n and classes c_0 through c_k, to determine the probability of the features occurring in each class, and to return the most likely class. Therefore, for each class, we want to be able to calculate P(c_i ||x_0, fi, x_n).

In order to do this, we use Bayes rule. Recall that Bayes rule is the following:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

baye's rules

In the context of classification, you can replace A with a class, c_i, and B with our set of features, x_0 through x_n. Since P(B) serves as normalization, and we are usually unable to calculate P(x_0,..., x_n), we can simply ignore that term, and instead just state that P(c_i | x_0,..., x_n) âĹİ P(x_0,..., x_n | c_i) * P(c_i), where α means "is proportional to". P(c_i) is simple to calculate; it is just the proportion of the dataset that falls in class i. P(x_0,..., x_n | c_i) is more difficult to compute. In order to simplify its computation, we make the assumption that x_0 through x_n are conditionally independent given c_i, which allows us to say that P(x_0,..., x_n | c_i) = P(x_0 | c_i) * P(x_1 | c_i) * ... * P(x_n | c_i). This assumption is most likely not true.hence the name naive Bayes classifier, but the classifier nonetheless performs well in most situations. Therefore, our final representation of class probability is the following:

$$\begin{aligned} P(c_i|x_0,\ldots,x_n) &\propto P(x_0,\ldots,x_n|c_i)P(c_i) \\ &\propto P(c_i)\prod_{j=1}^{n} P(x_j|c_i) \end{aligned}$$

Conditional relation

Calculating the individual P(x_j | c_i) terms will depend on what distribution your features follow. In the context of text classification, where features may be word counts, features may follow a multinomial distribution. In other cases, where features are continuous, they may follow a Gaussian distribution.

Note that there is very little explicit training in Naive Bayes compared to other common classification methods. The only work that must be done before prediction is finding the parameters for the featuresâĂŹ individual probability distributions, which can typically be done quickly and deterministically. This means that Naive Bayes classifiers can perform well even with high-dimensional data points and/or a large number of data points.

$$P(w|c) = \frac{\text{count}(w,c)+1}{\text{count}(c)+|V|+1},$$

smoothing function

Classification Now that we have a way to estimate the probability of a given data point falling in a certain class, we need to be able to use this to produce classifications. Naive Bayes handles this in a very simple manner; simply pick the c_i that has the largest probability given the data point's features.

This is referred to as the Maximum A Posteriori decision rule. This is because, referring back to our formulation of Bayes rule, we only use the P(B|A) and P(A) terms, which are the likelihood and prior terms, respectively. If we only used P(B|A), the likelihood, we would be using a Maximum Likelihood decision rule.[4]

	precision	recall	f1-score	support
icse	0.89	0.94	0.92	3328
sigmod	0.54	0.46	0.49	1384
vldb	0.54	0.55	0.54	1427
avg / total	0.73	0.74	0.73	6139

Results for Multinomial Naive Bayes

4.2 Support Vector Machines

What is Support Vector Machine? The objective of the support vector machine algorithm is to find a hyperplane in an N-dimensional space(N - the number of features) that distinctly classifies the data points. To separate the two classes of data points, there are many

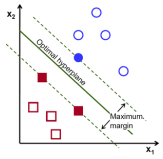


Figure 2: Support Vectors

possible hyperplanes that could be chosen. Our objective is to find a plane that has the maximum margin, i.e the maximum distance between data points of both classes. Maximizing the margin distance provides some reinforcement so that future data points can be classified with more confidence.

To separate the two classes of data points, there are many possible hyperplanes that could be chosen. Our objective is to find a plane that has the maximum margin, i.e the maximum distance between data points of both classes. Maximizing the margin distance provides some reinforcement so that future data points can be classified with more confidence.

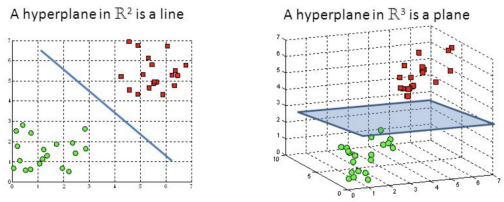


Figure 3: Hyper plane visualization

Hyperplanes in 2D and 3D feature space Hyperplanes are decision boundaries that help classify the data points. Data points falling on either side of the hyperplane can be attributed to different classes. Also, the dimension of the hyperplane depends upon the number of features. If the number of input features is 2, then the hyperplane is just a line. If the number of input features is 3, then the hyperplane becomes a two-dimensional plane. It becomes difficult to imagine when the number of features exceeds 3.[5] Support Vectors Support vectors are data points that are closer to the hyperplane and influence the position and orientation of the hyperplane. Using these support vectors, we maximize the margin of the classifier. Deleting the support vectors will change the position of the hyperplane. These are the points that help us build our SVM.

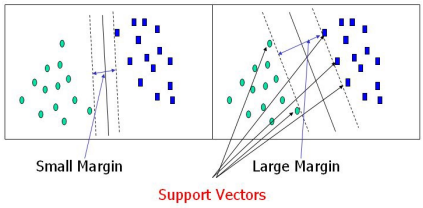


Figure 4: Support Vectors

$$c(x, y, f(x)) = \begin{cases} 0, & \text{if } y * f(x) \geq 1 \\ 1 - y * f(x), & \text{else} \end{cases}$$

Loss function.

	precision	recall	f1-score	support
icse	0.92	0.95	0.93	3328
sigmod	0.56	0.49	0.53	1384
vldb	0.56	0.58	0.57	1427
avg / total	0.75	0.76	0.76	6139

Results for Support Vector Machines with RBF kernel

4.3 K-means Clustering

clustering algorithm uses iterative refinement to produce a final result. The algorithm inputs are the number of clusters K and the data set. The data set is a collection of features for each data point. The algorithms starts with initial estimates for the K centroids, which can either be randomly generated or randomly selected from the data set. The algorithm then iterates between two steps: 1. Data assigment step: Each centroid defines one of the clusters. In this step, each data point is assigned to its nearest centroid, based on the squared Euclidean distance. More formally, if c_i is the collection of centroids in set C, then each data point x is assigned to a cluster based on

Loss function.

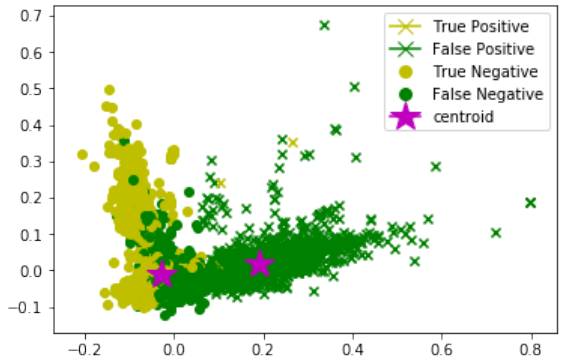
$$\underset{c_i \in C}{\operatorname{argmin}}\ dist(c_i,\ x)^2$$

where dist() is the standard (L2) Euclidean distance. Let the set of data point assignments for each i^{th} cluster centroid be S_i.

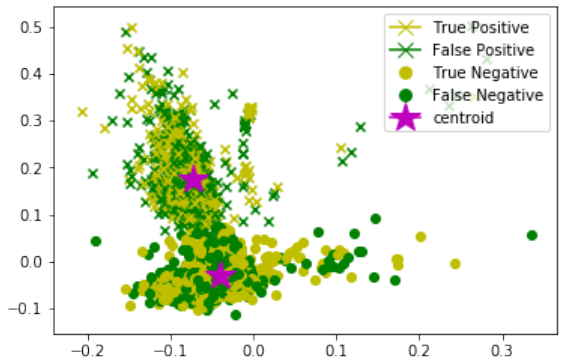
2. Centroid update step:

In this step, the centroids are recomputed. This is done by taking the mean of all data points assigned to that centroid's cluster.[6]

$$c_i = \frac{1}{|S_i|} \sum_{x_i \in S_i} x_i$$



ICSE and SIGMOD K-means, Purity = 0.52



VLDB and SIGMOD K-means, Purity = 0.4
as vldb and sigmod both have papers related to databases it is hard to get a good Purity score

4.4 Doc2Vec

Doc2Vec is a group of related models that are used to produce word embeddings. These models are shallow, two-layer neural networks that are trained to reconstruct linguistic contexts of words. Word2vec takes as its input a large corpus of text and produces a vector space, typically of several hundred dimensions, with each unique word in the corpus being assigned a corresponding vector in the space. Word vectors are positioned in the vector space such that words that share common contexts in the corpus are located in close proximity to one another in the space.

Word2vec was created by a team of researchers led by Tomas Mikolov at Google. The algorithm has been subsequently analysed and explained by other researchers.Embedding vectors created using the Word2vec algorithm have many advantages compared to earlier algorithms such as latent semantic analysis.[7]

4.5 Pagerank

PageRank is a link analysis algorithm and it assigns a numerical weighting to each element of a hyperlinked set of documents, such as the World Wide Web, with the purpose of "measuring" its relative

importance within the set. The algorithm may be applied to any collection of entities with reciprocal quotations and references. The numerical weight that it assigns to any given element E is referred to as the PageRank of E and denoted by PR(E) Other factors like Author Rank can contribute to the importance of an entity.

A PageRank results from a mathematical algorithm based on the webgraph, created by all World Wide Web pages as nodes and hyperlinks as edges, taking into consideration authority hubs such as cnn.com or usa.gov. The rank value indicates an importance of a particular page. A hyperlink to a page counts as a vote of support. The PageRank of a page is defined recursively and depends on the number and PageRank metric of all pages that link to it ("incoming links"). A page that is linked to by many pages with high PageRank receives a high rank itself.

Numerous academic papers concerning PageRank have been published since Page and Brin's original paper. In practice, the PageRank concept may be vulnerable to manipulation. Research has been conducted into identifying falsely influenced PageRank rankings. The goal is to find an effective means of ignoring links from documents with falsely influenced PageRank.[8]

$$PageRank\ of\ site = \sum \frac{PageRank\ of\ inbound\ link}{Number\ of\ links\ on\ that\ page}$$

OR

$$PR(u) = (1-d) + d \times \sum \frac{PR(v)}{N(v)}$$

Page Rank Formula

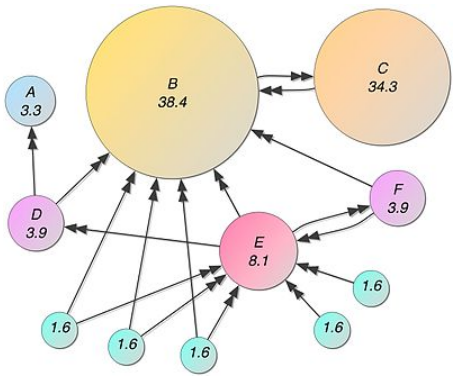


Figure 5: Page Rank Visualisation

5 DIMENSIONALITY REDUCTION

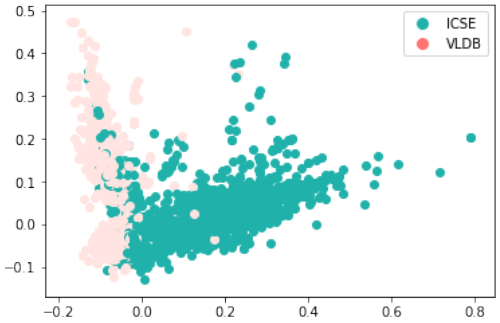
5.1 PCA

It does what it says on the tin. PCA finds the principal components of data.

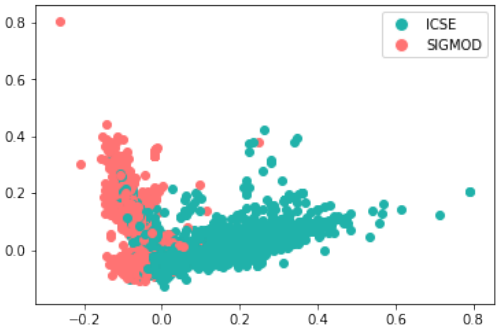
It is often useful to measure data in terms of its principal components rather than on a normal x-y axis. So what are principal components then? They're the underlying structure in the data. They are the directions where there is the most variance, the directions where the data is most spread out. This is easiest to explain by way of example. Here's some triangles in the shape of an oval:

Imagine that the triangles are points of data. To find the direction where there is most variance, find the straight line where the data is most spread out when projected onto it. A vertical straight line with the points projected on to it will look like this:

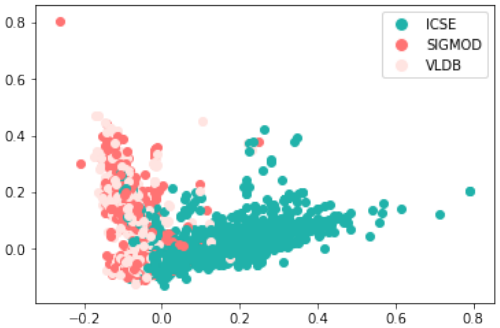
On this line the data is way more spread out, it has a large variance. In fact there isn't a straight line you can draw that has a larger variance than a horizontal one. A horizontal line is therefore the principal component in this example.



PCA dimensionality reduction to 2 dimensions ICSE and VLDB



PCA dimensionality reduction to 2 dimensions ICSE and SIGMOD



PCA dimensionality reduction to 2 dimensions VLDB and SIGMOD and ICSE

6 EVALUATION METRICS

6.1 Precision

Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. The question that this metric answer is of all passengers that labeled as survived, how many actually survived? High precision relates to the low false positive rate. We have got 0.788 precision which is pretty good.

Precision = TP/TP+FP

6.2 Recall (Sensitivity)

Recall is the ratio of correctly predicted positive observations to the all observations in actual class - yes. The question recall answers is: Of all the passengers that truly survived, how many did we label? We have got recall of 0.631 which is good for this model as itâĂŹs above 0.5.

Recall = TP/TP+FN

6.3 F1 score

F1 Score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account. Intuitively it is not as easy to understand as accuracy, but F1 is usually more useful than accuracy, especially if you have an uneven class distribution. Accuracy works best if false positives and false negatives have similar cost. If the cost of false positives and false negatives are very different, it's better to look at both Precision and Recall. In our case, F1 score is 0.701.

F1 Score = 2*(Recall * Precision) / (Recall + Precision)

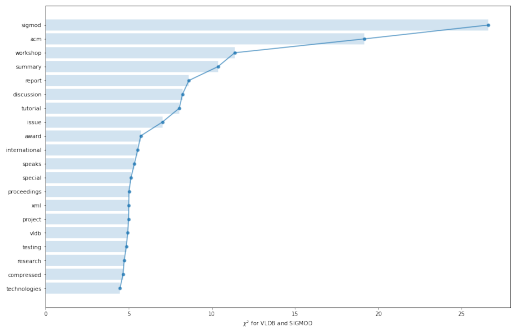
7 FEATURE SELECTION

7.1 Mutual information

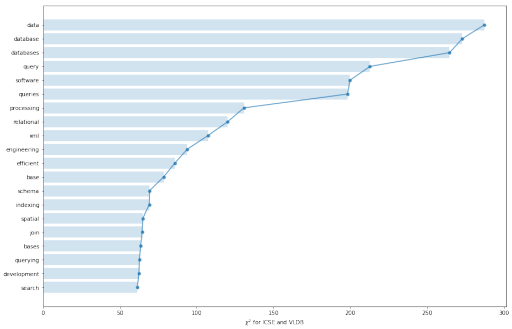
the mutual information of two discrete random variables X and Y can be defined $I(X;Y) = \int_{\mathcal{Y}} \int_{\mathcal{X}} p(x,y) \log \left(\frac{p(x,y)}{p(x)\,p(y)} \right) dx\,dy,$

where p(x,y) is the joint probability function of X and Y, and p(x) and p(y) p(y) are the marginal probability distribution functions of X and Y respectively.

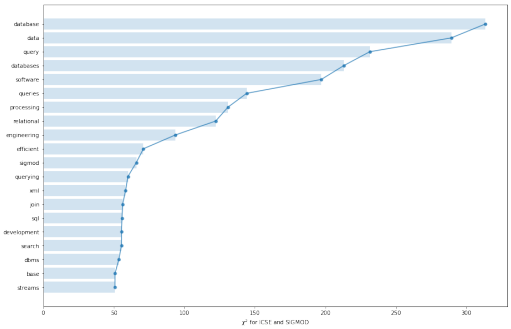
In the case of continuous random variables, the summation is replaced by a definite double integral[9]



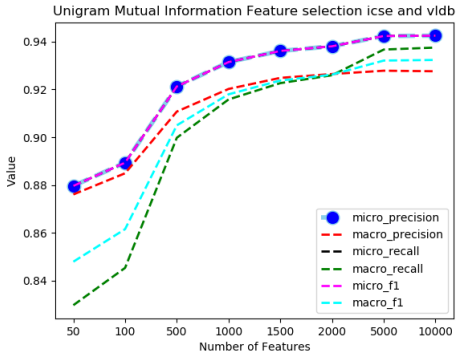
Most representative words of SIGMOD and VLDB



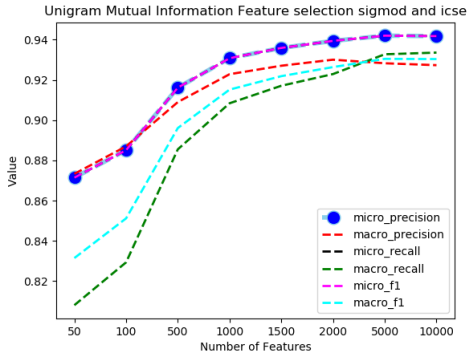
Most representative words of ICSE and VLDB



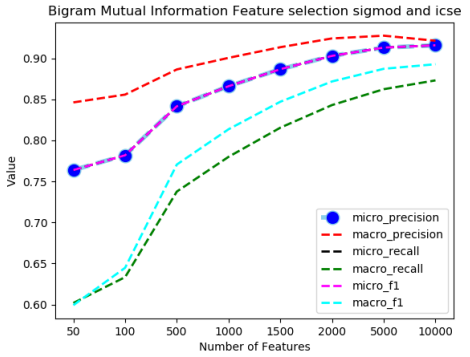
Most representative words of ICSE and SIGMOD



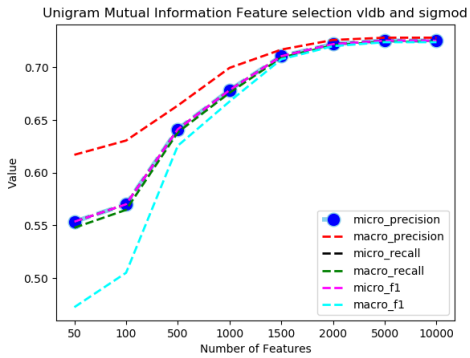
Unigram Mutual Information icse and vldb



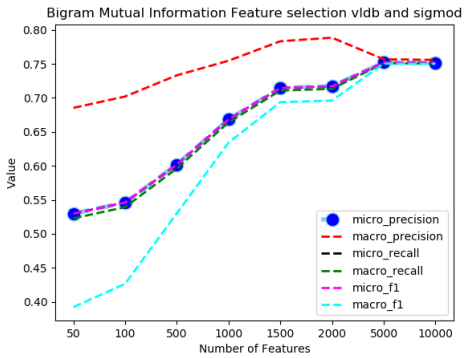
Unigram Mutual Information sigmod and icse



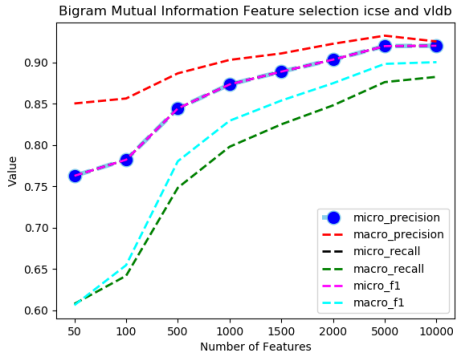
Bigram Mutual Information sigmod and icse



Unigram Mutual Information vldb and sigmod



Bigram Mutual Information vldb and sigmod



Bigram Mutual Information icse and vldb

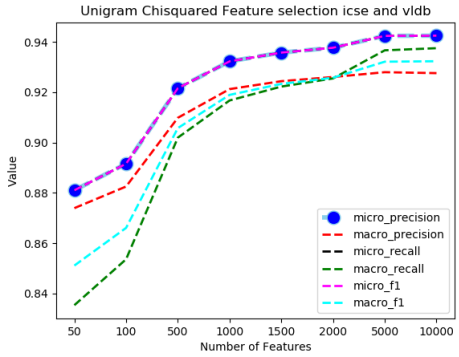
7.2 Chi Squared

Suppose that n observations in a random sample from a population are classified into k mutually exclusive classes with respective observed numbers xi (for i = 1,2,...,k), and a null hypothesis gives the probability pi that an observation falls into the $i^{th} class. So we have the expected numb$ np_i for all i, where

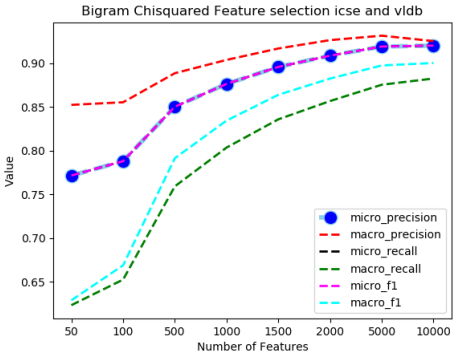
$$\sum_{i=1}^{k} p_i = 1$$

$$\sum_{i=1}^{k} m_i = n \sum_{i=1}^{k} p_i = \sum_{i=1}^{k} x_i$$

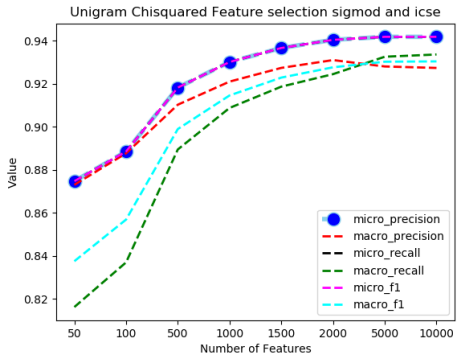
$$X^2 = \sum_{i=1}^{k} \frac{(x_i - m_i)^2}{m_i} = \sum_{i=1}^{k} \frac{x_i^2}{m_i} - n$$



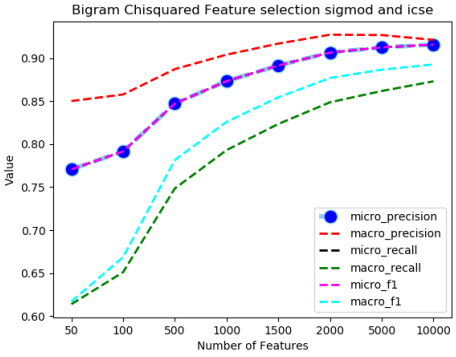
Unigram chisquared icse and vldb



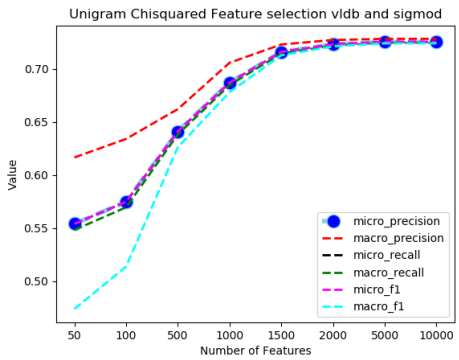
Bigram chisquared icse and vldb



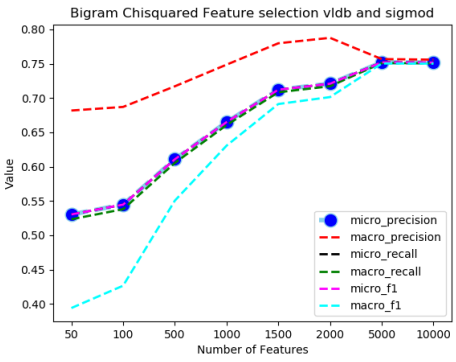
Unigram chisquared sigmod and icse



Bigram chisquared sigmod and icse



Unigram chisquared vldb and sigmod



Bigram chisquared vldb and sigmod

7.3 Why the precision recall and F1 scores are equal for micro averaging?

In order to calculate precision and recall, we need to know the amount of TP, FP and FN samples. How can you determine TP, FP and FN when you have a non-binary problem, i.e. more than just positive and negative as output? Imagine you have 3 classes (1,2,3) and each sample belongs to exactly one class. The following table shows the predicitons of our classifier for 9 test samples together with their correct labels.

Label	1	2	3	2	3	3	1	2	2
Prediction	2	2	1	2	1	3	2	3	2

Example

TP is the amount of samples that were predicted to have the correct label. In this example, TP = 4 (all green cells) FP is the amount of labels that got a "vote" but shouldn't. For example, in the first column, 1 should have been predicted, but 2 was predicted. So there is a false positive for class 2 in this case. On the other

hand, if the prediction is right (column 2), there is no FP counted. In this example, FP = 5 (all red cells) FN is the amount of labels that should have been predicted, but weren't. Look at the first column again. 1 should have been predicted, but wasn't. So there is a FN for class 1 in this case. As in the FP case, there is no FN counted if the prediction is correct (column 2). In this example, FP = 5 (all red cells)

In other words, if there is a false positive, there will always also be a false negative and vice versa, because always one class if predicted. If class A is predicted and the true label is B, then there is a FP for A and a FN for B. If the prediction is correct, i.e. class A is predicted and A is also the true label, denn there is neither a false positive nor a false negative but only a true positive. So there is no possibility that would increase only FP or FN but not both. That is why precision and recall are always the same when using the micro averaging scheme.The values of precision, recall and F1 score.
Precision P=44+5=49=0.4444
Recall R=44+5=49=0.4444
F1 score F1=249*4949+49=49249=49=0.4444
We can see that all metric values are identical.

I am also going to show how precision, recall and F1 score are calculated when using macro averaging instead of micro averaging. In this case, we first have to look at each class separately. Now we can treat every class as a binary label (class predicted yes/no).

In the previous example (see table above), each class has the following TP, FN, FP values and the following precision (P), recall (R) and F1 scores:
Class 1: TP = 0 / FN = 2 / FP = 2 => P = 0 / R = 0 / F1 = 0
Class 2: TP = 3 / FN = 1 / FP = 2 => P = 35 / R = 34 / F1 = 23
Class 3: TP = 1 / FN = 2 / FP = 1 => P = 12 / R = 13 / F1 = 25
All classes combined:
TP = 4 / FN = 5 / FP = 5
Precision (average over all classes): 0.36667
Recall (average over all classes): 0.36111
F1 (average over all classes): 0.35556.
These values differ from the micro averaging values

8 CONCLUSION AND FUTURE WORK

In conclusion, chi squared and mutual information are efficient statistical methods that we can use to reduce the dimension of the problem and still retain the accuracy in classification. For the further work, the front end of the search engine needs to be changed to Vue.js in order to make it more interactive.

REFERENCES

[1] *Brickmarketing* https://www.brickmarketing.com/define-search-engine-index.htm
[2] *Search DSL* https://www.elastic.co/
[3] *Aminer* https://aminer.org/citation
[4] *Naive Bayes* https://towardsdatascience.com/introduction-to-naive-bayes-classification-4cffabb1ae54
[5] svm https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47
[6] kmeans https://www.datascience.com/blog/k-means-clustering
[7] *Doc2vec Wikipedia.* https://en.wikipedia.org/wiki/Word2vec
[8] *Pagerank Wikipedia.* https://en.wikipedia.org/wiki/PageRank
[9] *Mutual information* https://en.wikipedia.org/wiki/Mutual_information