Data Mining
Final project report

Implementation of a single classification algorithm and several algorithms for feature reduction.

Przemysław Piórkowski, Michał Debski, Damian Hermanowski

June, 2015

Contents

Chapter 1

Introduction

The project objective is to implement single classification algorithm and several algorithms for feature reduction. In the addition, developed algorithms have to be evaluated against chosen data set and classification method.

Basing on the document provided by project's Supervisor and the initial research the Team decided to implement three types of the feature reduction algorithms: Document Frequency Thresholding (DFT), Mutual Information (MI) and χ^2 statistic (CHI). As a classification algorithm the k-nearest neighbors (kNN) method was chosen. The Team decided to use the C++ programming language for implementation purposes.

The source data set: SPAM E-mail Database, consists of 4601 instances of e-mails. Each e-mail is considered either as a spam or not. Classification is based on 57 continuous attributes denoting particular words and characters frequency as well as lengths of uninterrupted sequences of capital letters in a message. The data set does not have any missing attribute values.

Chapter 2

Algorithms implementation

DFT thresholding is the simplest technique for attribute reduction. Document frequency is the number of documents in which a term occurs. Our implementation counts occurences of not null values of each attribute. It utilizes the training set for this step. After that a vector of frequencies is created. An attribute which frequency is below given threshold is filtered out for further classification steps. The basic assumption is that rare terms are either non-informative for category prediction, or not influential in global performance.

MI tries to measure the influence of existence of particular attribute/term on a given class value. If one considers the twoway contingency table of a term t and a category c, where A is the number of times t and c co-occur, B is the number of time the t occurs without c, C is number of times c occurs without t, and N is the total number of documents, then the mutual information criterion between t and c is estimated using formula:

$$I(t, c) \approx \log \frac{A \times N}{(A + C) \times (A + B)}$$

I(t ,c) equal to zero means that attribute/term t and category c are independent. The goodness of aterm/attribute is scored in two ways: as a average of MIs for a given attribute (MI_AVG mode) and as a maximal MI for a given attribute (MI_MAX mode). The outcome of the algorithm is a reduction vector of attributes scores. An attribute with a score below a given threshold is filtered out.

$$I_{avg}(t) = \sum_{i=1}^{m} P_r(c_i) I(t, c_i)$$

$$I_{max}(t) = \max_{i=1}^{m}\{I(t, c_i)\}$$

The χ^2 statistic measures the lack of independence between an attribute t and a class c. Similarly to MI algorithm utilizes the twoway contingency table with an additional value D, which equals to case when neither c nor t occurs. The term-goodness is calculated with a given formula:

$$\chi^2(t, c) = \frac{N \times (AD - CB)^2}{(A + C) \times (B + D) \times (A + B) \times (C + D)}$$

Similarly to the MI method, the CHI statistic between the given attribute and classes may be computed as two scores: average value (CHI_AVG mode) and maximum value (CHI_MAX mode). The outcome of the algorithm is a vector of attributes scores. An attribute with a score below a given threshold is filtered out.

$$\chi^2_{avg}(t) = \sum_{i=1}^{m} P_r(c_i) \chi^2(t, c_i)$$

$$\chi^2_{max}(t) = \max_{i=1}^{m}\{\chi^2(t, c_i)\}$$

The project implements three feature reduction algorithms within 3 classes:

- CHIReduction
- DFTReduction
- MIReduction

The kNN algorithm is implemented in the **KNNClassifier** class. There were developed additional classes for data loading and data interpretation: **DataLoader**, **Data**. There is the **Logger** class for debugging and echoing purposes. There is the class supporting multithreading for parallelization capabilities: **ParallelExecutor** with a support of *AtomicIfPossible* and *AtomicInternal* classes.

The developed project application can be run with given parameters:

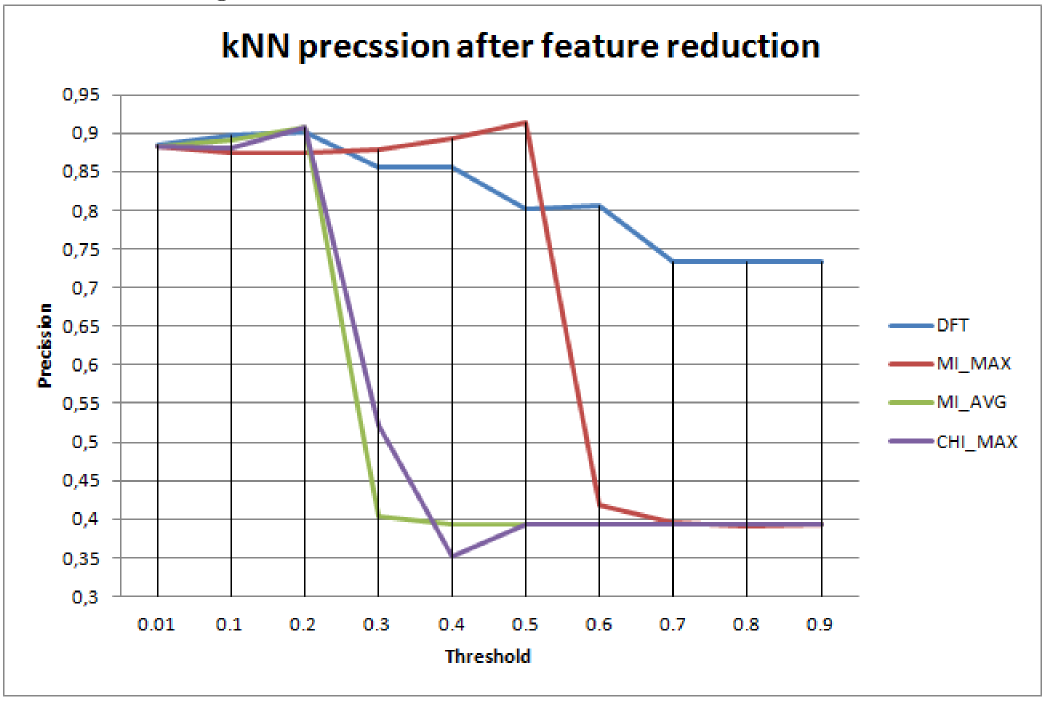
- `-ftrain trainFileName`
- `-ftest testFileName`

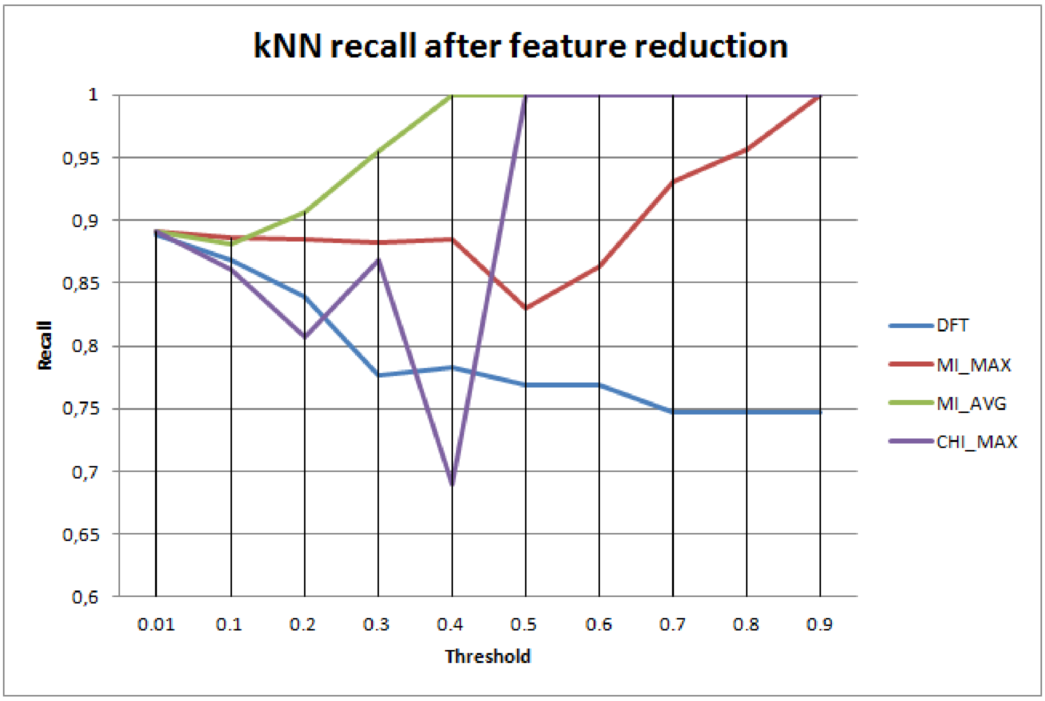
- `-h` - switch for reading headers of a source file
- `-cname className`
- `-k value`
- `-DFT dftThreshold`
- `-MI_MAX` - switches on MAX score calculation for MI algorithm
- `-MI_AVG` - switches on AVG score calculation for MI algorithm
- `-MI miThreshold`
- `-l verbosityLevel`
- `-CHI_MAX`- switches on MAX score calculation for CHI algorithm
- `-CHI_AVG`- switches on AVG score calculation for CHI algorithm
- `-CHI chiThreshold`

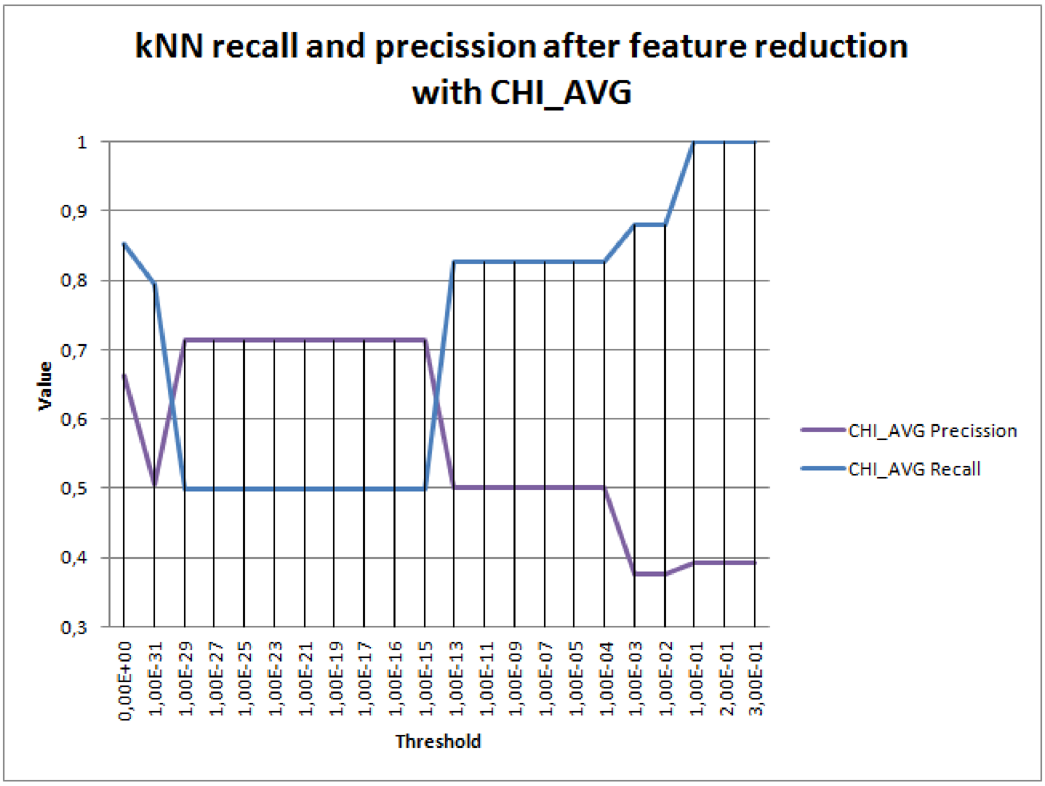
Chapter 3

Experiments

The source data set was arbitrary divided into 2 subsets: the training set and the test set. At the beggining classifier was used to build a model with the training set. kNN implementation has one control parameter - the number of neighbors voting. For evaluation purposes this k-value was set to 5. The influence of each feature reduction algorithm on the kNN classification method was measured with precission and recall values. These quality measures were compared with the results of classification without feature reduction against the test set.




kNN recall after feature reduction
1
0,95
0,9
0,85
0,8
0,75
0,7
0,65
0,6
Recall
0.01
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
Threshold
DFT
MI_MAX
MI_AVG
CHI_MAX


kNN recall and precission after feature reduction with CHI_AVG
1
0,9
0,8
0,7
0,6
0,5
0,4
0,3
Value
0,00E+00
1,00E-31
1,00E-29
1,00E-27
1,00E-25
1,00E-23
1,00E-21
1,00E-19
1,00E-17
1,00E-16
1,00E-15
1,00E-13
1,00E-11
1,00E-09
1,00E-07
1,00E-05
1,00E-04
1,00E-03
1,00E-02
1,00E-01
2,00E-01
3,00E-01
Threshold
CHI_AVG Precission
CHI_AVG Recall

Chapter 4

Conclusions

The results are interesting. The application of each feature reduction method (without CHI_AVG) on the given data set gave slight precission gain for lower threshold values. For MI_MAX method even applying threshold of value 0.50 did not decrease overall precission. MI_MAX, MI_AVG and CHI_MAX show meaningful precission drop for specific thresholds. For the DFT the fall of recall and precission with growing threshold is smooth - it is more stable algorithm. The recall curves for all methods but DFT are rising while precission is falling. Considering precisssion gains for lower threshold it has been shown that MI_MAX algorithm should be chosen. The DFT algorithm would be good for reducing the higher number of attributes.