BAHÇEŞEHİR UNIVERSITY

FACULTY OF ENGINEERING

DEPARTMENT OF COMPUTER ENGINEERING

CLASSIFICATION WITH FEATURE SELECTION BY USING MINIMUM REDUNDANCY MAXIMUM RELEVANCE AND K-NEAREST NEIGHBORS METHODS

Submitted by

ONUR ÖZÜDURU

Advisor

CEMAL OKAN ŞAKAR

İSTANBUL, JANUARY 2015

BAHÇEŞEHİR UNIVERSITY

FACULTY OF ENGINEERING

DEPARTMENT OF COMPUTER ENGINEERING

This project work submitted by Onur ÖZÜDURU has been done under my supervision. I hereby state that the work outlined in this report satisfies the requirements of the compulsory "Capstone Project" course, and Onur ÖZÜDURU can take the capstone project examination.

Signature and Date

…….

CEMAL OKAN ŞAKAR

Onur ÖZÜDURU has taken and passed the capstone project examination in our presence on January ……, 2015. We hereby certify that this project work fulfills all requirements of "Capstone Project" course.

THE EXAMINATION COMMITTEE

Committee Member	Signature
1. ……..	……..
2. ……..	……..
3. ……..	……..

I have reviewed this capstone project report submitted by Onur ÖZÜDURU. I hereby endorse the project work described here as a "Capstone Project."

Signature and Date

…….

Assist. Prof. K. Egemen Özden

Chairman of the Dept. of Computer Engineering

SUMMARY

Classification problem is one of the major problems in machine learning and k-Nearest Neighbors algorithm is a well known method for classification problems. In that project my main goal was combining with two popular methods that are minimum Redundancy Maximum Relevance, which is a feature selection method, and the k-Nearest Neighbors algorithm to solve classification problems with better accuracy. Idea behind the project is that selecting n features from features set with using minimum Redundancy Maximum Relevance method then applying k-Nearest Neighbors algorithm on data set which includes only with these selected n features. In addition to that idea it is repeated the process several times and voted each results, so the most voted results are determined as final result also it is generated new training sets from base set with replacement for each loop time to improve accuracy. I used discretized data set with nine classes and sixty samples (NCI cancer cell lines) for testing my approach. I used different distance methods and compare them to see which distance method gives more successful results. I added comparison tables on this report, as you can see results are promising. However there is performance problems on development, since there are many calculations it works slowly. For future studies performance problems can be handled with developing more effective code which uses mutual information.

ÖZET

Sınıflandırma problemi makine öğrenme alanında en bilinen problemlerden biridir ve k-Nearest Neighbors algoritması da sınıflandırma alanındaki tanınan algoritmlar arasındadır. Bu projede benim ana amacım özellik seçme metodlarından minimum Redundancy Maximum Relevance metodu ve k-Nearest Neighbors algoritması gibi iki popüler metodu sınıflandırma problemlerini daha doğru şekilde çözebilmek için birleştirmekti. Projedeki fikir; minimum Redundancy Maximum Relevance methodu ile özellik kümesinden n tane özellik seçip sadece bu seçilen özellikleri içeren veri kümeleri üzerinde k-Nearest Neighbors algorithmasını çalıştırmaktır. Bu fikre ek olarak süreci bir çok kez tekrar ederek en çok oy alan sonuç gerçek sonuç olarak sunulmaktadır, daha doğru sonuçlar elde edebilmek için. Bu projede discretized edilmiş, altmış örnekli (NCI kanser hücre şeritleri) veri setini kullandım. Ayrıca hangi uzaklık metodunun daha iyi sonuç verdiğini görmek için farklı uzaklık metodlarını kullandım. Eklediğim karşılaştırma tablolarında da görebileceğiniz gibi sonuçlar umut verici. Ancak çok fazla hesaplama olduğu için geliştirme aşamasında yavaşlık sorunları mevcuttur. Gelecek çalışmalarda bu performans sorunu bağıl verileri kullanarak daha verimli bir kod yazıp çözülebilir.

1 ACKNOWLEDGEMENTS

I want to thank my Project advisor Cemal Okan ŞAKAR on helping and guiding me on my project and I want to thank my friends and my family for their motivational talks.

Onur ÖZÜDURU

İstanbul, January 2015

TABLE OF CONTENTS

LIST OF FIGURES AND TABLES

2 ABBREVIATIONS

k-NN: k-Nearest Neighbors

mRMR: minimum Redundancy Maximum Relevance

NCI: National Cancer Institute

MI: Mutual Information

3 Introduction

In this study I combined feature selection approach with classification approach to get more stable results for classification. The proposed thesis is that if features in a data set can be eliminated by feature selection method then it will give more accurate results when applied classification method on the data set which has decreased number of feature by feature selection. My goal was getting straight class predictions from classification method with using that idea by using minimum Redundancy Maximum Relevance algorithm as feature selection method and using k-Nearest Neighbors algorithm as classification method.

I am interested in machine learning area. I took Special Topics at Software Engineering class which was about artificial intelligence as specialized topic at Bahcesehir University. My term project was about shortest path algorithms for that class. I told my project advisor C. Okan ŞAKAR about my interests and classes that I took at the past then he suggested to chose that subject. So, that is the reason why I have been selected this subject for my project topic.

The general approach -that combining feature selection and classification approaches together- which is used in that study is not a new approach. There are other studies which use that approach[1,2,3,4,5,6,7,8,9,10] in general however in a special look there are differences between all, about used algorithms, classification and feature selection methods, data sets and analysis (see Section 4.)

As I mentioned before my goal on this project (which will be called as *Voting mRMR* to be short) was combining minimum Redundancy Maximum Relevance algorithm and k-Nearest Neighbors algorithm to get accurate results for classification. The way in that study for combining these two algorithm is as follows assume that we have two data sets *D* which we know its classes, and *G* which we do not know its classes. To figure out classes of *G*, first mRMR algorithm applied for *N* times to get *K* selected features on *N* subsets $(S_1, S_2, \ldots, S_N)$ which are generated randomly with

replacement from a main dataset D $(\forall S_i \subset D.)$ After features had eliminated for each sub set S_i, let T_i denote S_i with K features that selected by mRMR, $T_1, T_2, \ldots, T_N$ are given as argument for training set and G is given as argument for sample set to k-NN algorithm for N times with same k value and distance method. It is recorded return values of k-NN for G every time and after that when that recorded answers had been voted, the algorithm in this study suggests classes that were most voted in the recorded answers, as result classes or classes of G.

I used mRMR approach to decrease number of features in the data set and applied k-NN algorithm on sub sets which has fewer features than the main data set, of the main data set. I worked on Matlab platform and I got Matlab source code and the data set that used in experiments from Mr. Peng's website which is about mRMR[11, 31]. I used *knnclassify* function which is a built-in function in Matlab, as k-NN algorithm. I tested the code for different distance methods and different number of features that selected by mRMR algorithm such as features that selected at least one time, features that selected more than 200 times and 50 features that is return value of mRMR. In experiments I used NCI data which is discretized as 3 states (-2, 0, 2) and included 9 cancers (classes), 60 samples, 9712 features [12]. First I divided that data set into two sub sets as training and sample. Then I applied voting mRMR for 1000 times. I used k values 1 to 9 and used different distance methods to see which is better. And I applied only k-NN on same subsets for understanding that voting mRMR does really give more accurate results. I faced performance problems that applying the voting mRMR for 1000 loop times takes about 5 hours. So it is obvious that there are implementation problems for effectiveness in this code. For future studies that problem might be handled by using better memory management approaches.

3.1 Thesis Outline

The outline of this thesis is as follows:

Section 3 Is a brief introduction about the project. The idea behind the project is briefly explained
and introduced here.

Section 4 presents related works about the approach which have influenced this project.

Section 5 introduces tools, methods and algorithms that are used in this project and explains how and why they used in this project.

Section 6 describes experimental results how the method on that project was applied on data set, which data set had been used. And commented results are given for this project.

Section 7 presents conclusions about work carried during and end of progress of project.

4 Related Works

The general approach -that combining feature selection and classification approaches together- which is used in that study is not a new approach. Even more there is a published paper that under the title "Feature Selection for Classification"[1] which was written by M. Dash and H. Liu. In addition same approach can be seen as the studies[2,3,4,5,6,7,8,9] specially about bioinformatics and text classification.

Of course there are differences between these studies that I mentioned above in special. They all used different combinations of feature selection and classification algorithms with each others for example the study[9] by Cai, Y., Huang, T., Hu, L., Shi, X., Xie, L., & Li, Y. was combined mRMR algorithm with Nearest Neighbor algorithm[10] and applied these approach on data set which is about amino acids.

In this study I applied mRMR as feature selection algorithm with k-NN as classification algorithm on NCI cancer data set, and implemented the code on Matlab platform. And I compared experimental results with different distance methods (see Section 5.3.1.1.)

5 Materials and Methods

The main goal of that study is improving results of k-Nearest Neighbors algorithm as it has been mentioned above sections. To do that I combined k-NN with mRMR algorithm why my thesis was if features of data could be eliminated according to their relevance before k-NN is applied then k-NN would give more accurate results. In general I propose that applying feature selection before classification will give more proper results for data sets which have large set of features.

I used Matlab platform and language while implementing Voting mRMR code (see Appendix B). I used Matlab version of minimum-redundancy maximum-relevance feature selection source code[13] and Mutual Information Toolbox[14] which both are uploaded by Hanchuan Peng from Matlab Central website[15]. For k-Nearest Neighbor algorithm I used built-in function in Matlab which is called as *knnclassify*. I tried to use different distance methods as possible for *knnclassify* function, such as euclidean distance, correlation distance, cosine distance, city-block distance. The code was written and tested on the computer which has system properties that are given at Appendix A.

5.1 MATLAB®

MATLAB® is the high-level language and interactive environment used by millions of engineers and scientists worldwide [16]. The MATLAB language provides native support for the vector and matrix operations that are fundamental to solving engineering and scientific problems, enabling fast development and execution. MATLAB add-on products provide built-in algorithms for signal processing and communications, image and video processing, control systems, and many other domains. By combining these algorithms with your own, you can build complex programs and applications [17].

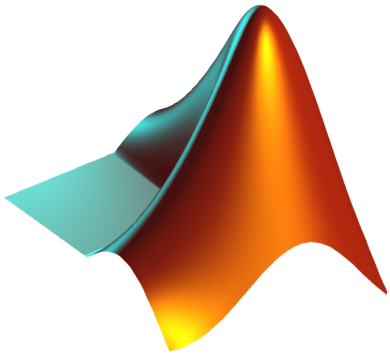

Figure 1. Matlab Logo

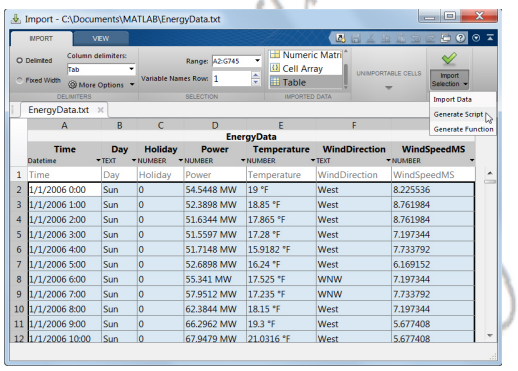

Figure 2. Data as seen on Matlab

I have chosen MATLAB platform and language why it is easy to implement algorithm and simple test platform for implemented codes.

5.2 Feature Selection

As machine learning aims to address larger, more complex tasks, the problem of focusing on the most relevant information in a potentially overwhelming quantity of data has become increasingly important. For instance, data mining of corporate or scientific records often involves dealing with both many features and many examples, and the Internet and World Wide Web have put a huge volume of low-quality information at the easy access of a learning system. Similar issues arise in the personalization of filtering systems for information retrieval, electronic mail, net-news, and the like[18].

The main goal of feature selection is to select a subset of d features from the given set of D measurements, $d < D$, without significantly degrading the performance of the recognition system. Assuming that a suitable criterion function has been chosen to evaluate the effectiveness of feature subsets, feature selection is reduced to a search problem that detects an optimal feature subset based on the selected measure [20].

To sum up feature selection is the problem of choosing a small subset of features that ideally is necessary and sufficient to describe the target concept[19].

5.2.1 mRMR Algorithm

In that study I applied mRMR method[31] to select features for classification. To more accurate results it is better to decrease number of feature before applying k-NN method that is why mRMR is used for eliminating features.

In minimum Redundancy Maximum Relevance feature selection, the goal is to select a feature subset set that best characterizes the statistical property of a target classification variable, subject to the constraint that these features are mutually as dissimilar to each other as possible, but marginally as similar to the classification variable as possible [21]. Mutual information (MI), which measures the mutual dependence of two variables, is used to quantify both relevance and redundancy in this method [22]. Mutual information is defined as the following formula.

$$I(x;y) = \iint p(x,y)\log\frac{p(x,y)}{p(x)p(y)}dxdy$$

where

x and y *are two random variables*.

$p(x), p(y), p(x,y)$ *probabilistic density functions*.

Let Ω denote the whole feature set, while Ωs denotes the already-selected feature set which contains m features and Ωt denotes the to-be-selected feature set which contains n features. Relevance D of the feature f in Ωt with the target c can be calculated by:

$$D = I(f,c) \quad (1)$$

And redundancy R of the feature f in Ωt with all the features in Ωs can be calculated by:

$$R = \frac{1}{m}\sum_{f_i \in \Omega_s} I(f,f_i) \quad (2)$$

To obtain the feature f j in Ωt with maximum relevance and minimum redundancy, Eqs. (1) and (2) are combined with the mRMR function:

$$\max_{f_j \in \Omega_t} \left[I(f_j, c) - \frac{1}{m} \sum_{f_i \in \Omega_s} I(f_j, f_i) \right] (j = 1,2,\ldots,n)$$

For a feature set with *N(=m + n)* features, the feature evaluation will continue N rounds. After these evaluations, we will get a feature set S by mRMR method:

$$S = \{f_1', f_2', \ldots, f_h' \ldots, f_N'\}$$

The feature index h indicates the importance of feature. The better a feature is, the smaller its index h will be [23].

5.3 Classification Problem

The term "classification" concerns any context in which some decision is taken or a forecast is made on the basis of currently available knowledge or information. A classification algorithm is an algorithm which permits us to repeatedly make a forecast on the basis of accumulated knowledge in new situations. Each new object is assigned to a class belonging to a predefined set of classes on the basis of observed values of suitably chosen features [24]. The essential characteristic of a classification problem is that the problem solver selects from a set of pre-enumerated solutions. This does not mean, of course, that the "right answer" is necessarily one of these solutions, just that the problem solver will only attempt to match the data against the known solutions, rather than construct a new one. Evidence can be uncertain and matches partial, so the output might be a ranked list of hypotheses [25].

5.3.1 k-NN Classification Algorithm

The k-nearest neighbor algorithm, which is most often used for classification, although it can also be used for estimation and prediction. k-Nearest neighbor is an example of instance-based learning, in which the training data set is stored, so that a classification for a new unclassified record may be found simply by comparing it to the most similar records in the training set [26].

In that study I used built in function at Matlab which calls as *knnclassify*. The syntax of that function is described as follows [27],

Class = knnclassify(*Sample*, *Training*, *Group*, *k*, *distance*)

where arguments are,

Sample: Matrix whose rows will be classified into groups. Sample must have the same number of columns as Training.

Training: Matrix used to group the rows in the matrix Sample. Training must have the same number of columns as Sample. Each row of Training belongs to the group whose value is the corresponding entry of Group.

Group: Vector whose distinct values define the grouping of the rows in Training.

k: The number of nearest neighbors used in the classification. Default is 1.

Distance: String specifying the distance metric. Choices are:

'euclidean' — Euclidean distance (default)

'cityblock' — Sum of absolute differences

'cosine' — One minus the cosine of the included angle between points (treated as vectors)

'correlation' — One minus the sample correlation between points (treated as sequences of values)

5.3.1.a Distance Methods for k-NN

Data analysts define distance metrics to measure similarity. A distance metric or distance function is a real-valued function d, such that for any coordinates x, y, and z [26]:

1. $d(x, y) \geq 0$, and $d(x, y) = 0$ if and only if $x = y$
2. $d(x, y) = d(y, x)$
3. $d(x, z) \leq d(x, y) + d(y, z)$

The most common distance function is Euclidean distance[26]. I used four different distance methods for comparing results and to see which distance method gives more accurate class predictions. In that study I did not use hamming distance which is also one of the arguments for *knnclassify* function because of data set must be in binary form to use that distance method.

Formulas for distance methods that were used in that study can be seen at the following sub sections.

5.3.1.a.1 Euclidean Distance

Formula for euclidean distance is [28,29]

$$d_E = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

5.3.1.a.2 Cosine Distance

Formula for cosine distance is[28,29]

$$d_C = 1 - \cos(\phi) \quad \textit{where} \quad \frac{\langle x, y \rangle}{\|x\|_2 \cdot \|y\|_2} = \cos(\phi)$$

5.3.1.a.3 Correlation Distance

Formula for correlation distance is[28,29]

$$d_{st} = 1 - \frac{(x_s - \bar{x}_s)(x_t - \bar{x}_t)'}{\sqrt{(x_s - \bar{x}_s)(x_s - \bar{x}_s)'}\sqrt{(x_t - \bar{x}_t)(x_t - \bar{x}_t)'}}$$

where

$$\bar{x}_s = \frac{1}{n}\sum_j x_{sj} \quad \text{and} \quad \bar{x}_t = \frac{1}{n}\sum_j x_{tj}$$

5.3.1.a.4 City-block Distance

Formula for city-block distance is[28,29]

$$d_{st} = \sum_{j=1}^{n} |x_{sj} - x_{tj}|$$

6 Experimental Results

The data set that was used for experiments is NCI data which is discretized as 3 states (-2, 0, 2) and included 9 cancers (classes), 60 samples, 9712 features [12]. It is available at Mr. Peng's web site under sample data sets section [30].

The code was tested with one thousand training subsets that was created with replacement from main data set and 1 to 9 k values for four different distance methods which are euclidean distance, city-block distance, cosine distance and correlation distance. 60 samples in the data set had been separated two parts as *sample set* which has 20 samples and *training set* which has 40 samples before it was given to the functions as arguments. Distribution of classes on the main data set (with 60 samples) was considered while the separation process.

The success rate was calculated with the following formula:

$$Success\ Rate(x) = \frac{100 * (\sum_{i=1}^{n} d(x_i, y_i))}{n} \quad where \quad d(a,b) = \begin{cases} 1, & a-b=0 \\ 0, & otherwise \end{cases}$$

x: *Result Vector of k - NN*
y: *Real Class Values Vector*
n: *Length of x*
* *Length of x* and *length of y must be equal*.

In that case *n* is equal to 20 since the *sample set* has 20 rows. As a result, the success rate is the percentage of number of successfully predicted class values by k-NN.

6.1 Self Comparative Results

Before to see compared results for voting mRMR, it is better to inspect results for only k-NN and k-NN with features that selected by mRMR for different distance methods. As it is mentioned before, there are two subsets: the sample set with 20 samples and the training set with 40 samples. For the results of the experiments which consist of k-NN applied on features that are selected by mRMR (more than 200 features, more than 1 features, voting mRMR,) the data set had been sent mRMR as argument for 1000 times for 50 features (k = 50) and for each time subsets were created with replacement to provide for randomize that is why we got slightly different 50 selected features for every time.

The following index scheme is an explanation for column names of comparison results for distance methods.

K: Value of the argument call as k for k-NN.

Average k-NN: Average success rate when k-NN applied for 1000 times on subsets that are generated with replacement, it is for seeing differences between k-NN and mRMR k-NN combinations.

>= 200 features: Applied k-NN for the features which are selected 200 times or more than 200 times by mRMR (these features are found by counting every features on 1000x50 mRMR result matrix.)

>= 1 features: Applied k-NN for the features which are selected at least one time by mRMR (these features are found by counting every features on 1000x50 mRMR result matrix.)

Voting mRMR: Applied k-NN for the features which are selected by mRMR when k value for mRMR equals 50.

6.1.1 Comparison Results for Euclidean Distance

K	Average k-NN	>= 200 features	>= 1 features	Voting mRMR
1	69.8250	65	75	75
2	69.8250	65	75	75
3	67.0850	70	65	80
4	68.6650	75	65	80
5	68.7300	75	70	85
6	69.1700	80	70	85
7	68.6950	80	60	85
8	68.2950	80	55	80
9	67.2350	80	50	80

Table 1. results for euclidean distance method.

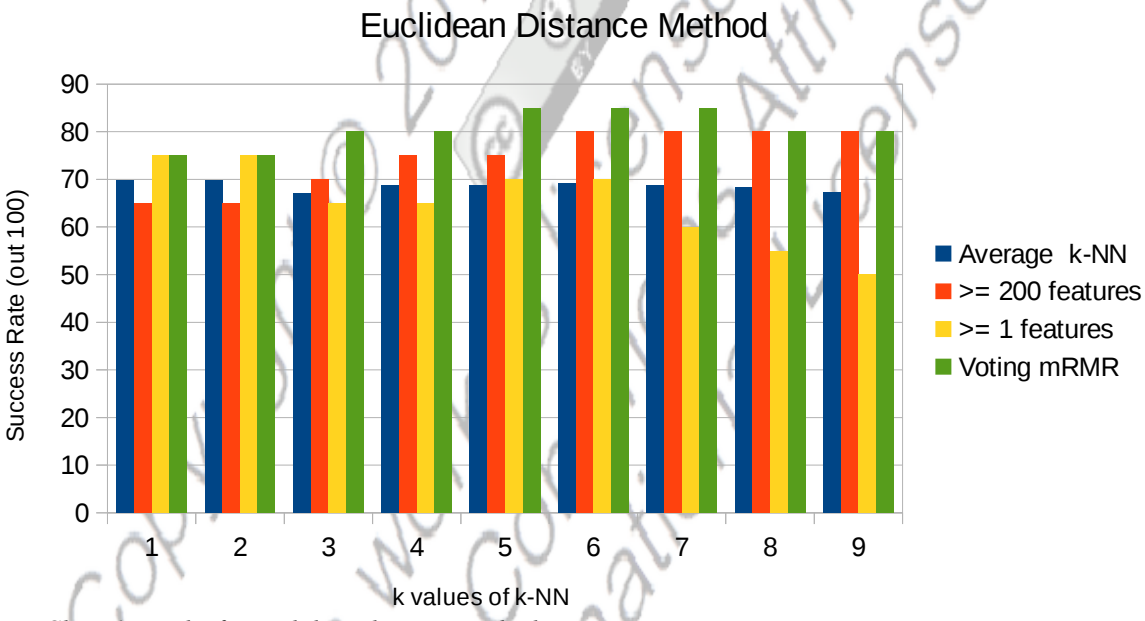

Chart 1. results for euclidean distance method.

Results for euclidean distance (*Table 1 and Chart 1*) are shown that voting mRMR method gives about 10% - 15% more success rate than using k-NN directly on the data. And we can see that success rate is decreased when k > 5 for data sets that consist of at least 1 selected feature by mRMR. Voting mRMR gives better performance when k = 5, 6, 7 and almost for all k values it is better than others. On the other hand there are interesting results which are when k = 1 or 2 there is no difference between voting mRMR and data sets that consist of at least 1 selected

feature by mRMR and data sets that consist of more than 200 features and voting mRMR gives same results when k = 8 or 9.

In conclusion it can be said that voting mRMR gives more accurate results for euclidean distance according to that result table and it works better when k = 5, 6, 7 for euclidean distance.

6.1.2 Comparison Results for Cosine Distance

K	Average k-NN	>= 200 features	>= 1 features	Voting mRMR
1	72.3050	75	75	80
2	72.3050	75	75	80
3	68.6400	75	65	80
4	69.4550	70	65	80
5	69.4300	75	70	85
6	69.7400	75	70	85
7	68.8400	80	70	85
8	68.0300	80	65	85
9	65.7200	80	60	85

Table 2. results for cosine distance method.

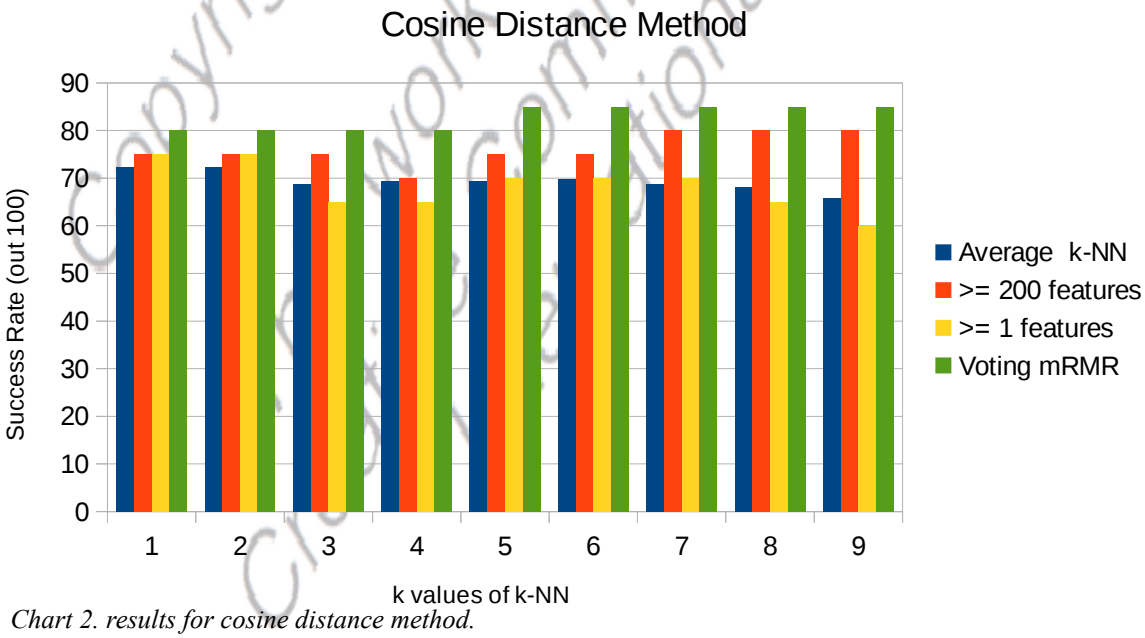

Chart 2. results for cosine distance method.

Results for cosine distance (*Table 2 and Chart 2*) are shown that voting mRMR method gives about 5% - 20% more success rate than using k-NN directly on the data. And we can see that there is no special behavior for different k values on data sets that consist of at least 1 selected feature by mRMR. Voting mRMR gives better performance when k = 5, 6, 7, 8, 9 and for all k values it is better than others.

In conclusion it can be said that voting mRMR gives more accurate results for cosine distance according to that result table and it works better when k = 5, 6, 7, 8, 9 for cosine distance.

6.1.3 Comparison Results for Correlation Distance

K	Average k-NN	>= 200 features	>= 1 features	Voting mRMR
1	71.5100	80	75	90
2	71.5100	80	75	90
3	67.2350	75	70	85
4	67.2700	65	65	80
5	66.5500	60	65	80
6	65.3950	65	65	80
7	64.7600	65	60	80
8	63.1200	60	55	75
9	60.5800	60	55	70

Table 3. results for correlation distance method.

Results for correlation distance (*Table 3 and Chart 3*) are shown that voting mRMR method gives about 10% - 20% more success rate than using k-NN directly on the data. And we can see that success rate is decreased when k > 2 for all cases. Voting mRMR gives better performance when k = 1 or 2 and for all k values it is better than others. On the other hand there are interesting results which are when k = 4, 5, 6, 7 voting mRMR gives same success rates that is 80%.

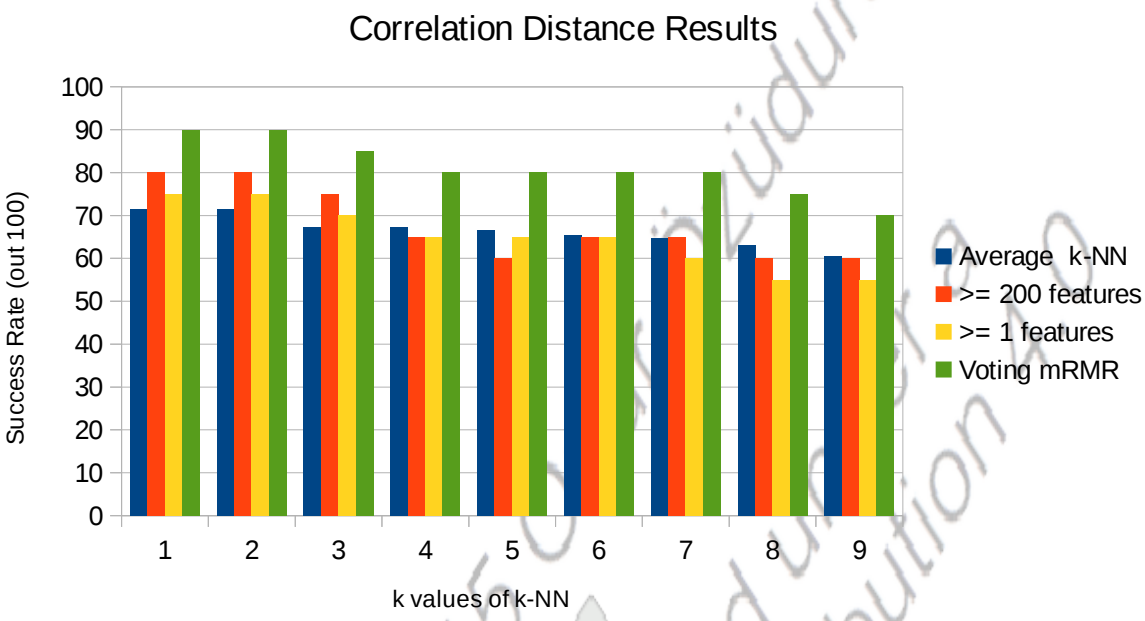

Chart 3. results for correlation distance method.

In conclusion it can be said that voting mRMR gives more accurate results for correlation distance according to that result table and it works better when k = 1 or 2 for correlation distance.

6.1.4 Comparison Results for City-block Distance

K	Average k-NN	>= 200 features	>= 1 features	Voting mRMR
1	75.6050	75	75	85
2	75.6050	75	75	85
3	72.6000	80	70	80
4	73.5900	80	70	85
5	73.4950	80	75	85
6	73.8050	80	70	85
7	73.1950	80	65	85
8	72.4100	85	60	85
9	70.8400	85	55	85

Table 4. results for city-block distance method.

Results for city-block distance (*Table 4 and Chart 4*) are shown that voting mRMR method gives about 5% - 15% more success rate than using k-NN directly on the data.

And we can see that success rate is decreased when k > 6 for data sets that consist of at least 1 selected feature by mRMR. Voting mRMR gives the same performance except when k = 3 and almost for all k values it is better than others. On the other hand there are interesting results which are when k = 1 or 2 there is no difference between data sets that consist of more than 200 features and data sets that consist of at least 1 selected feature by mRMR and data sets that consist of more than 200 features and voting mRMR gives same results when k = 8 or 9.

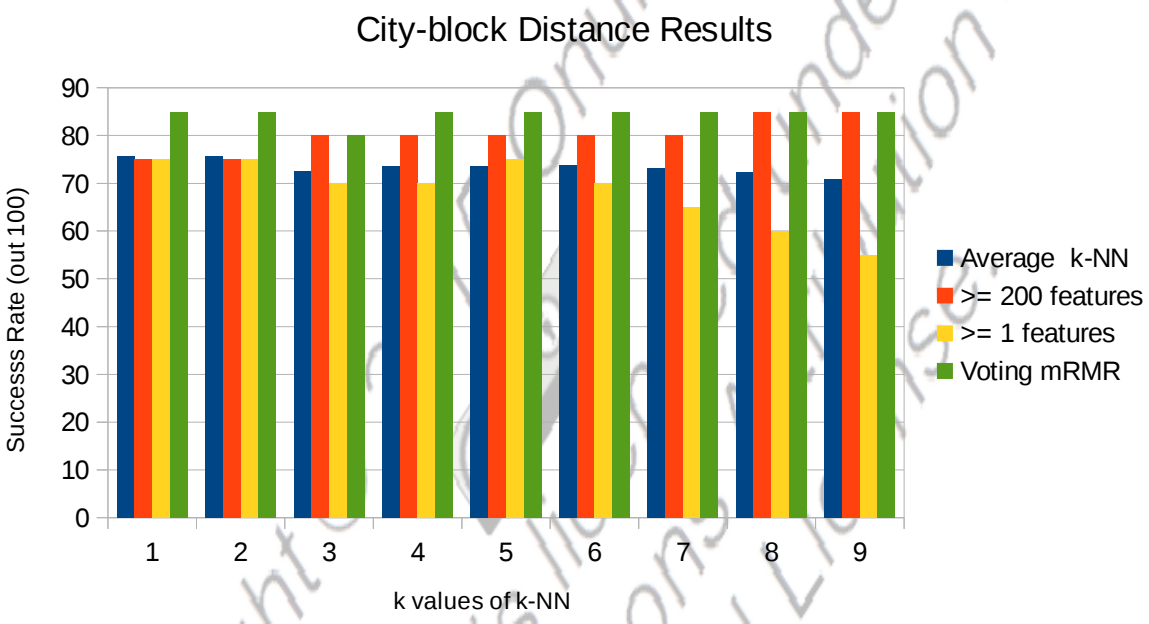

Chart 4. results for city-block distance method.

In conclusion it can be said that voting mRMR gives more accurate results for city-block distance according to that result table and it works very stable except when k = 3 for city-block distance.

6.1.5 Comparison Results for Voting mRMR

K	Euclidean Distance	Cosine Distance	Correlation Distance	City-block Distance
1	75	80	90	85
2	75	80	90	85
3	80	80	85	80
4	80	80	80	85
5	85	85	80	85
6	85	85	80	85
7	85	85	80	85
8	80	85	75	85
9	80	85	70	85

Table 5. results for voting mRMR method.

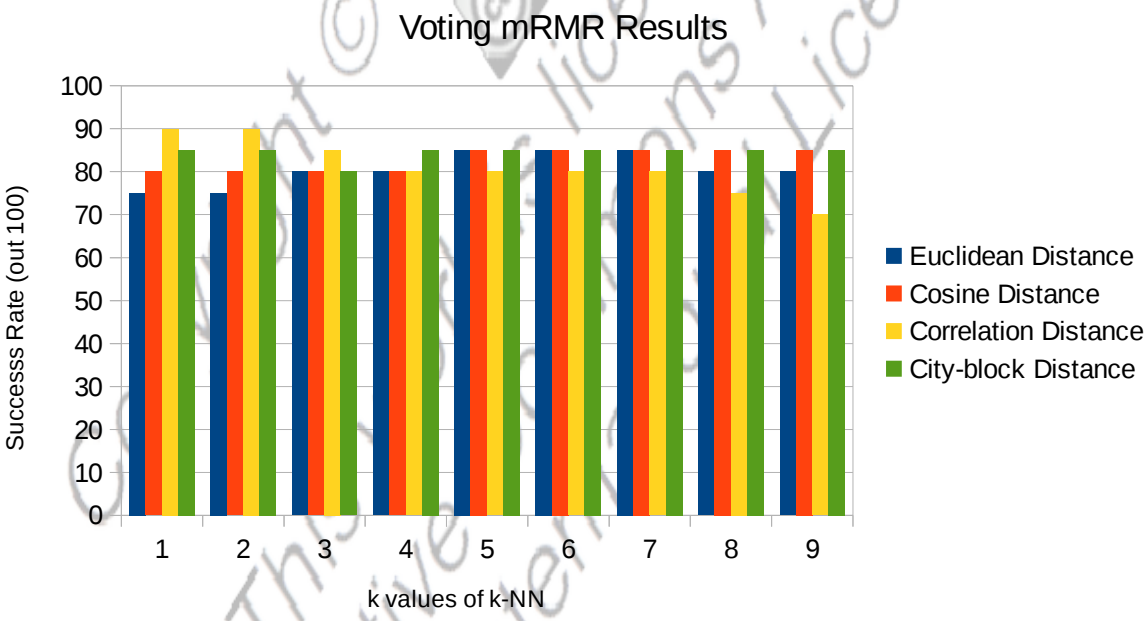

Chart 5. results for voting mRMR method.

When we compare results for euclidean, cosine, correlation and city-block distance methods with 1 to 9 k values on voting mRMR, it shows that when k = 1, 2 or 3 correlation distance method gives better performance but its success rate is decreased while k is increased. For euclidean distance and cosine distance methods it is better to

use k values that are greater than 4 however we cannot say that one is better than other when k = 5, 6 or 7.

I think the most interesting results here are belongs to city-block distance since it seems that changes on k values almost not affected to city-block distance method except the case when k = 3.

Therefore, while comparing different k values and distance methods for voting mRMR it can be said that for small k values (k < 4) it is better to use correlation distance method however for k > 4 city-block and cosine gives better performance also it must be denoted that changes on k values do not affected success rate for city-block distance method.

7 Conclusion

In conclusion combining minimum Redundancy Maximum Relevance algorithm and k-Nearest Neighbors algorithm to get accurate results for classification with voting method. It gives better success rates than using only k-NN algorithm, as it has been at experiments. And when comparing different distance methods and approaches using subsets that consist of at least 1 time selected features or more than 200 times selected features by mRMR is not a good idea. Using subsets with features that selected by mRMR and than voting k-NN algorithm's return values is a better approach. In addition to that for small k values (k = 1 or 2) on voting mRMR gives better results.

It must be considered that there are performance issues on implementation as I mentioned before applying voting mRMR takes too much time according to apply k-NN with same loop times. For future studies that situation can be handled with better memory management also voting mRMR algorithm can be tested with different datasets.

8 References

1. Dash, Manoranjan, and Huan Liu. "Feature selection for classification." Intelligent data analysis 1.3 (1997): 131-156.

2. Kwak, Nojun, and Chong-Ho Choi. "Input feature selection for classification problems." Neural Networks, IEEE Transactions on 13.1 (2002): 143-159.

3. Liu, Huiqing, Jinyan Li, and Limsoon Wong. "A comparative study on feature selection and classification methods using gene expression profiles and proteomic patterns." Genome Informatics Series (2002): 51-60.

4. Punch III, William F., et al. "Further Research on Feature Selection and Classification Using Genetic Algorithms." ICGA. 1993.

5. Mladeni'c, Dunja, and Marko Grobelnik. "Feature selection for classification based on text hierarchy." Text and the Web, Conference on Automated Learning and Discovery CONALD-98. 1998.

6. Forman, George. "An extensive empirical study of feature selection metrics for text classification." The Journal of machine learning research 3 (2003): 1289-1305.

7. Neumann, Julia, Christoph Schnörr, and Gabriele Steidl. "Combined SVM-based feature selection and classification." Machine Learning 61.1-3 (2005): 129-150.

8. Li, Tao, Chengliang Zhang, and Mitsunori Ogihara. "A comparative study of feature selection and multiclass classification methods for tissue classification based on gene expression." Bioinformatics 20.15 (2004): 2429-2437.

9. Cai, Yudong, et al. "Prediction of lysine ubiquitination with mRMR feature selection and analysis." Amino acids 42.4 (2012): 1387-1395.

10. Cover, Thomas, and Peter Hart. "Nearest neighbor pattern classification." Information Theory, IEEE Transactions on 13.1 (1967): 21-27.

11. http://penglab.janelia.org/proj/mRMR/

12.http://penglab.janelia.org/proj/mRMR/test_nci9_s3.csv

13.http://www.mathworks.com/matlabcentral/fileexchange/14916-minimum-redundancy-maximum-relevance-feature-selection

14.http://www.mathworks.com/matlabcentral/fileexchange/14888-mutual-information-computation

15.http://www.mathworks.com/matlabcentral/

16. http://www.mathworks.com/products/matlab/index.html

17. http://www.mathworks.com/products/matlab/features.html

18. Blum, Avrim L., and Pat Langley. "Selection of relevant features and examples in machine learning." Artificial intelligence 97.1 (1997): 245-271.

19.Kira, Kenji, and Larry A. Rendell. "The feature selection problem: Traditional methods and a new algorithm." AAAI. 1992.

20. Pudil, Pavel, Jana Novovičová, and Josef Kittler. "Floating search methods in feature selection." Pattern recognition letters 15.11 (1994): 1119-1125.

21. http://penglab.janelia.org/proj/mRMR/FAQ_mrmr.htm#Q1.1

22. Cai, Yudong, et al. "Prediction of lysine ubiquitination with mRMR feature selection and analysis." Amino acids 42.4 (2012): 1387-1395.

23. Cai, Yudong, et al. "Prediction of lysine ubiquitination with mRMR feature selection and analysis." Amino acids 42.4 (2012): 1387-1395.

24. Bazan, Jan G., et al. "Rough set algorithms in classification problem." Rough set methods and applications (2000): 49-88.

25. Clancey, William J. Classification problem solving. Department of Computer Science, Stanford University, 1984.

26. Larose, Daniel T. Discovering knowledge in data: an introduction to data mining. John Wiley & Sons, 2014.

27. http://www.mathworks.com/help/bioinfo/ref/knnclassify.html

28. http://www.mathworks.com/help/stats/pdist.html

29. Deza, Michel Marie, and Elena Deza. "Encyclopedia of distances." Encyclopedia of Distances (2009): 1-583.

30. http://penglab.janelia.org/proj/mRMR/#data

31. [TPAMI05] Hanchuan Peng, Fuhui Long, and Chris Ding, "Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 27, No. 8, pp.1226-1238, 2005.

9 Appendix A

CPU	Intel® Core™2 Duo CPU T6600 @ 2.20GHz × 2
GPU	AMD RV730
Memory	3 Gb
OS	Ubuntu 14.04 LTS
OS-type	32-bit

10 Appendix B

```
function res = voting_mrmr(mrmrIt, mrmrK, samp, train, groupT, knnK, DIS)

[sampM, sampN] = size(samp);
unique_classes = unique(groupT);
voteChart = zeros(sampM, length(unique_classes));

for it = 1:mrmrIt
   [mrmrMat, mrmrClasses] = mrmr_sample_creator(horzcat(groupT, train));
   knn_voter(mrmr_mid_d(mrmrMat, mrmrClasses, mrmrK));
end

res = get_voting_results(voteChart);

    % Creates matrices that is size of [# data rows x # data columns] with replacement.
    % Returns mixed data sample and its classes vector.
    function [sampleMat, classVec] = mrmr_sample_creator(data)
        [datM, datN] = size(data);
        [sampleMat,] = datasample(data, datM, 'Replace', true);
        classVec = sampleMat(:, 1);
        sampleMat = sampleMat(:, 2:datN);
    end

    % Modifies voteChart according to results of mrmr.
    function knn_voter(mrmrRes)
        S = [];
        T = [];
        % Picks only choosen features by mrmr from sample and training.
        for j = 1:mrmrK
            S = horzcat(S, samp(:, mrmrRes(j)));
            T = horzcat(T, train(:, mrmrRes(j)));
        end
        knnRes = knnclassify(S, T, groupT, knnK, DIS);
        for j = 1:sampM
            indexVote = find(unique_classes == knnRes(j));
            voteChart(j, indexVote) = voteChart(j, indexVote) + 1;
        end
    end

    % Calculates the classes that takes most of the votes and
    %  returns a result vector which includes classes of samp.
    function andTheWinnersAre = get_voting_results(finalChart)
        andTheWinnersAre = zeros(sampM, 1);
        for i = 1:sampM
            [val, ind] = max(finalChart(i, :));
            andTheWinnersAre(i) = unique_classes(ind);
        end
    end
end
```

11 Curriculum Vitae

Onur ÖZÜDURU

Address: ***

Mobile: ***

E-mail: onur.ozuduru { at } gmail { dot } com

Website: http://ozuduru.com

Career Objective

Participate in software developer companies.

Education

2010 – 2015 : Computer Engineering at Bahcesehir University

2012 – 2013 Spring Semester: Bahcesehir University Berlin Campus