International Journal of Computer Information Systems and Industrial Management Applications.
ISSN 2150-7988 Volume 14 (2022) pp. 226-238

Received: 10 January2021; Accepted: 20 March, 2021; Published: 22 April, 2022

A Cross-Entropy Based Feature Selection Method for Binary Valued Data Classification

Zhipeng Wang and Qiuming Zhu

Department Of Computer Science, College of Information Science and Technology
University of Nebraska at Omaha, Omaha, Nebraska 68182 USA
zhipengwang@unomaha.edu, qzhu@unomaha.edu

***Abstract*: Feature selection is a process of finding a meaningful subset of attributes from a given set of measurements for a purpose of revealing a coherent relation or causality in an event. The process is often indispensable to facilitate an effective pattern classification. It is usually a preprocessing step before constructing a machine learning model in big data analytics for improving the accuracy of predictive results. By selecting the most significant features, it could reduce the time of training and the complexity of the model, avoid data overfitting, and help user to better understand the source data and the modeling outcomes. Though features are commonly dealt with in continuous values, many features appear to be binary valued, i.e., either 1 or 0, in many real-world machine learning applications. Inspired by existing feature selection methods, a new framework called FMC_SELECTOR was presented in this paper which addresses specifically the selection of significant features of binary valued attributes from highly imbalanced large datasets. The FMC_SELECTOR combines the fisher linear discriminant analysis with a cross-entropy mechanism to create an integrated mapping function for evaluating each individual features from a given dataset. A new formulization called Mapping Based Cross-Entropy Evaluation (MCE) was derived for a quantitative ranking of the features. A Positive Case Prediction Score (PPS) is explored to verify the significance of the features selected in a classification process. The performance of FMC_SELECTOR is compared with two popular feature selection methods – the Univariate Importance (UI) and Recursive Feature Elimination (RFM), and shows a better performance on the datasets tested.**

***Keywords*:** Binary Features, Feature Selection, Cross Entropy, Pattern Classification, Model Verification**.**

I. Introduction

The features of an object, treated as individually measurable properties of the subject, are foremost essential and fundamental to a pattern recognition and machine learning process [6]. Many research works have recognized that by selecting the most important and significant features from the input dataset, it is possible to generate a better machine learning model and improve the overall accuracy of a classification. The feature selection process also has big effect in reducing the overfitting of dataset and increasing the precision of the model predictions [15].

Devijver and Kittler introduced the main concepts of feature selection in early 1980s [5]. They reviewed the heuristic methods for feature selection and used it for reducing the feature space of pattern classification. Later, Kenji and Rendall introduced a novel practical approach called Relief algorithm [2]. It was inspired by the instance-based learning and used two different ways to define the difference values of the nominal and numerical features. The general idea was to calculate the weight between two instances and compare them with a threshold to determine whether the selected feature is relevant or not. Exhaustive search was applied to go through every subset of the features by a given size and find the best value. In 1997, Blum and Langley gave a brief overview of feature selection techniques by providing important definitions and three categories of feature selection methods, namely the filter, wrapper, and embedded approaches [1].

Lately, a crow search algorithm was introduced by Dr. Askarzadeh for feature selection [30]. The algorithm was used to solve constrained engineering optimization problems by simulating the features that crows store their foods and retrieve it. In another paper, Xue et al. surveyed approaches on using evolutional computations for feature selection [31]. They concluded that some popular approach such as genetic algorithm (GA), genetic programming (GP), and particle swarm optimization (PSO) could be used to successfully improve the feature selection. However, they still pointed out some other issues, such as the scalability, effectiveness, and efficiency that were the important points for further improvement and could be addressed for potential future developments. Along that line, Anter and Ali designed a "hybrid crow" search optimization algorithm which integrated with chaos theory and fuzzy c-means algorithms for feature selection in medical domain [29]. Their general idea was to utilize the global optimization method and chaos theory to compensate for the lack of convergence of crow search algorithm (CSA) which transferred the random variables from Gaussian distribution to chaotic behavior. There was another paper which analyzed the clinic data of stroke patients by improving the feature selection method by Setyawati et al. [36]. They investigated a Fuzz Entropy method to generate the entropy values for each feature. By selecting a proper set of candidate features, they could achieve a 96% accuracy in diagnosis by only using 13 out of 23 features. Feature selection for breast cancer diagnosis, clinical symptoms of

Diabetic Retinopathy, and for the prediction of heart disease among smokers were reported in [22], [34] and [35].

In recent years, there are quite numbers of research focused on the verification of the existing feature selection methods. Post and his team [4] conducted experiments on the most popular feature selection algorithms by using 400 datasets. Surprisingly, only 41 percent of algorithms improved the pattern classification results by using the selected features and only 10 percent of them can significantly improve the classification results. Another main difficult problem for feature selection is the large search space which leads to O(2n) computational complexity in possible solutions for a dataset which has n features, where some features are inter-correlated somehow. A feature which is not strongly relevant to target might performed well when it combines with other features [27]. To address such situations, several papers proposed their idea based on combined existing knowledge or modified existing formulas to address the above concerns [15] [16].

In addition to the algorithmic approaches, researchers also implemented various mathematical models for feature selection by using differential evaluation criteria and measurements, for example, Fisher's Linear Discriminate (FLD) score, mutual information, ANOVA, chi-square, etc., [10].

While a number of traditional techniques were well developed, there were some public software libraries encapsulated the existing functions together for user to apply, for example, scikit-learn (SKLearn), the machine learning package in Python [13]. Even though these software packages provided much convenience for general public to use, the current state-of-art of the achievements of these library functions were still not satisfactory especially in terms of finding the most effective and accurate solutions to the general problems in many real world scenarios. Therefore many researchers keep working on developing new algorithms or improving the current methods for more efficient feature selection processes.

It is known that most real-world datasets contain three types of feature values: binary type, continuous type, categorized type. Many real world datasets for pattern recognition are binary valued, for example, in representing the 'yes' or 'no' answers to survey questions, 'positive' or 'negative' remarks in medical records, 'presence' or 'absence' of forensic evidences in cyber security, 'paid' or 'default' status for insurance or financial fraud detection, etc. Another source of binary features come from data preprocessing. During the data collection stage, it is sometimes preferred to conduct a pre-analysis process to convert non-binary valued features to binary ones that could help researchers to understand each individual feature better and provide a clearer picture for the collected dataset. It is possible to utilize the modified dataset for the rest process which could not only save the effort but also get a better machine learning model. In the other words, binary values are popular in many applications because they can more easily determine whether a feature meeting certain criteria or not and giving a definite answer.

In the context of pattern classification, these binary values are mostly denoted as either a number '0' or a '1'for the features in the data to be analyzed. Very often the values of these binary features are sparsely populated, i.e., the number of '0's far exceeds the number of '1's in multiple folds, or vice versa. Large sparse matrices are common in general for data analytics tasks. When the data is sparse the consequences would be: (1) lacking enough information to fit a discriminative or predictive model, thus losing the generalization capability of the model; or (2) missing the solution points for maximizing or minimizing a target function, thus causing dimensional distortion and inaccuracy of the outcome. There could be two main causes for a feature value to be valued '0': (1) genuine zeros where the '0' represents one meaningful value, such as that in a medical record where most test results or diagnoses are negative; (2) missing values where a '0' is used in place of 'null' for data that lacks a specification, which includes the cases that data is not collected, incomplete, or uncertain (e.g., unfilled fields in a survey or medical record, gaps in a sensory data inputs or record keeping, etc.). We will not try to distinguish the cases of above but consider them together as a presence of sparse data indistinctively in this paper for the simplicity of discussion.

The significance of a feature with respect to its classification capability and performance cannot be judged simply by the sparseness. A sparsely populated feature could still be discriminatively significant and useful in a model for data classification as long as the underlying information for the discrimination remains. A well populated (dense) feature is not necessary to be significant in terms of contributing to the classification accuracy if the critical piece of information for drawing the discriminative outcome is missing. For the former case, pattern classification systems can still in turn put the different pieces of the incomplete, imprecise, or uncertain information together on an integrative ground for properly classifying the samples to different class belongings.

In practice, many datasets with sparseness and binary valued features are also highly imbalanced, e.g., the amount of normal cases far outnumbers the amount of abnormal cases for medical diagnosis or for fraud detection. The high imbalance ratio of the dataset poses another critical challenge to the classification of samples because the large disproportionateness of the number of samples between the majority (often the negative) class and the minority (often the positive) class. It is more likely for the samples of already disadvantaged minority class, in terms of its relatively smaller number of membership in the dataset, to loss the necessary informative ground for the classification task due to the sparseness of the feature values. Moreover, most feature assessment and performance evaluation metrics tend to work on balanced datasets, thus would produce a biases outcome when the dataset is imbalanced. The inconsistency of the results between the balanced and imbalanced datasets makes the evaluation of the sparsely populated binary features more difficult.

The traditional ways of handling the imbalanced dataset in classification is to do oversampling (Mock the positive data) or data under-sampling (Reduce the size of negative data) to address such concerns. However, mocking the positive data can introduce the fake information into the dataset while reducing the negative data would cause the uncertainty and randomness for the data. Therefore both these operations suffer from the modification of the original data in a somehow

unjustifiable way though efforts were taken to make the modifications as statistically sound as possible.

How do we know if a sparsely populated binary feature still possesses the necessary information for classification on a given dataset? How to assess the usefulness of the feature in a highly imbalanced dataset with sparsely populated values and still get the most significant or dominant features be selected for a classification task? That is, how to identify the "discriminatively significant" features of a sparsely populated binary feature in terms of its production of a high classification rate and low false alarm rate for both positive class and negative class samples? Since most previous experiments in feature selection were conducted on features of continuously distributed and balanced dataset, the theory and idea from previous work might not be suitable for the type of imbalanced datasets containing mostly binary valued features. It is necessary and critic to address this specific issue by developing feature selection methods that perform well for this kind of datasets.

This paper is organized as follows: section II provides an overview of the technique foundation, and presents our new method namely the cross-entropy approach for binary valued feature selection. Section III describes our FMC_SELECTOR approach and its associated MCE and PPS algorithms for the feature selection and verification. Section IV describes our experiments and test results, and section V concludes with a summary.

II. Overview of Binary Valued Feature Selection

In many real world applications of machine learning, it is not uncommon that the raw datasets contains hundreds, even thousands of features. It would be very inefficient, and many times impractical to take account of all the features in building a predictive model. It is only possible to thoroughly analyze a portion of the features in many cases. On the other hands, only a subtle subset of the features has significant influence on the outcomes (the targets) of the effectiveness and accuracy of the final model in many applications. It is therefore necessary to know and acquire the subset from the overall features that are most critical to the building and validation of the model to be built. It is also necessary to identify the individual features that has significant importance to reveal the cause-and-effect relations of certain event or phenomena, for example, the single or a few of the crucial functional genes affecting a specific disease, the major factors of a medical complication, the critical causes of a social or political event, the influential elements on the sales of certain consumer products, etc. To address such concerns, applying feature selection would filter out non-essential features therefore simplify the machine learning model and reduce the construction time. Sparse binary data could have an immense effect on the ability to properly assess and evaluate the discriminative significances of these features, thus the selection of these features deserves special attention and treatment.

The general steps of feature selection method contain follows; (1) Applying an algorithm to all the features from given dataset; (2) Evaluating and selecting candidate feature into new feature set; (3) Continuing perform previous two steps until reaching the stopping condition; (4) Validating the feature selection method by evaluating the machine learning model which is generated from its return value. There are three theoretical foundations which can be applied for all the feature selection algorithms: filter method, wrapper method, and embedded method [7][8][28]. Most of these general steps and approaches are applicable to feature selection on binary valued datasets.

II.1. Filter Method

The filter method is a type of method that purely relies on mathematical formulation without utilizing the machine learning algorithm. The method selects features based on certain criterion. To define such criterion, most methods focus on relationships between feature and feature, relationships between feature and target, etc. The types of filter method can be adopted from information theory, correlation, distance, consistency, fuzzy-set and rough-set. There are two steps for a typical filter method: (1) applying math formula, and (2) making decision on the feature selection [17]. The first step is to apply math formula to each feature with required variable in that formula. After that, a list of values is represented as a specific characteristic for all the features. Then, the decision step will decide which feature is important or not by ranking the list of scores from either high to low or low to high. It is important to decide how many features need to be chosen. Most common algorithms in terms of filter method include: Information Gain [18], Fisher Score [19], ReliefF [20], etc.

II.2. Wrapper Method

Instead of developing a rigorous mathematical model, some researchers suggested a heuristics method by applying machine learning algorithms directly to a feature selection process [21]. The method utilizes the performance of classifier as an evaluation criterion on the selected feature set. By checking the machine learning outcomes of different feature set, it is possible to distinguish significant features by comparing the prediction result from the classification processes directly. It usually involves three steps. First step is to enter the original feature set into different types of machine learning algorithms. The second step is to remove the unimportant features based on the prediction results from the first step. The last step is to recursively repeat the previous steps until getting the optimized result.

The processes of a typical wrapper method can also be divided into three categories: (1) backward selection, (2) forward selection, and (3) stepwise selection. The backward selection refers to a process that starts with all the features. For each iteration, the algorithm removes one feature based on the lowest prediction value. By contrasting to the backward selection, the forward selection starts with zero feature and for each iteration, the algorithms select the feature with the highest prediction score. The stepwise selection is a hybrid method of forward and backward selection. For example, when adding a new feature to the existing feature set one would use a forward selection. It then performs a round of backward selection to remove the unrelated feature in existing candidate feature set. The method addresses the correlative relations among the features. Suchetha [21] suggested some classical wrapper methods, such as the Conditional Random Fields (CRF) and the Recursive Feature Elimination (RFM).

Some common machine learning algorithms for model construction are Naive Bayes (NB) [38], K-Nearest Neighbors (KNN) [22], Linear Regression, etc.

II.3. Embedded Method

The embedded method can be considered as a combination of the previous two methods – the filter and wrapper. There are many ways to implement an embedded method. For example, a mathematical formula can be integrated into a machine learning algorithm so that the feature selection process can be conducted during the construction of the machine learning model. The typical embedded methods are algorithms such as Random Forest and Extra Tree [14]. Also, the regularization approaches are popular, among them the LASSO (L1 regularization) and Ridge (L2 regularization) methods are the most common types [33].

Depends on different situations, it is sometimes hard to choose one type of feature selection method over the others in the real-world applications. However, there were several notable characteristics for each feature selection method. For example, the advantage of filter method is that the machine learning algorithm will not influence the decision. By comparing to wrapper method, the filter method is able to avoid overfitting where it is a possible situation that the input data are performed well in some algorithms but not well in other algorithms. Also, overfitting may happen due to the nature of algorithm itself when a procedure works well on one type of data but another type of data might not perform well by using the same procedure. However, since the filter method is independent from machine learning algorithm itself, the precision of the features selected might suffer. To address this problem, the wrapper method comes into play. Wrapper method can be considered as a heuristic method which tries all the possible combinations of sub-feature sets to generate the result. However, it can also be considered as a cheat method since it relies on testing all the combination which could lead to higher order of time consumption and the complexity of algorithm itself. Also, due to the nature of machine learning algorithm, it is necessary to decide which algorithms to be involved in the learning stages. The embedded method would take both advantages from previous methods. They can be faster than wrapper method and more accurate than filter method. However, it is unclear whether the result coming from one machine learning algorithm could perform well with other machine learning algorithms based on the different requirement criteria.

In this research, a filter-based method is adopted for selecting the most significant features for building a machine learning model for pattern classification on the dataset. Using filter method can minimize the influence from different machine learning algorithms and decrease the processing time. Also, the filter-based method would explore the intrinsic characteristic of binary features which could be meaningful for this research.

III. Cross-Entropy Measurement of Binary Valued Features

It is well known that entropy is used to measure how much information contains in or can be obtained from a given variable, as it was introduced by Claude Shannon [12]. In machine learning area, Cross-entropy is used to measure the distribution of two variables for revealing their coherent relations [6]. Although there are various feature selection methods, the cross-entropy theory was not used for this purpose yet.

III.1. Cross-Entropy

Cross-Entropy is a computational mechanism that calculates the difference between two probability distributions over a discrete random variable *x*, say *p* and *q*, such that

$$H(p, q) = -\sum_{x \in X} p(x) \log q(x).$$

In the continuous case of *x*, it is expressed as

$$H(p, q) = -\int_x P(x) \log Q(x) \mathrm{d}x, \text{ where } p = \frac{dP}{dx} \text{ and } q = \frac{dQ}{dx}.$$

Hooper [11] suggested that since the formula itself is non-symmetric therefore it is important to identify *p* and *q* properly. In machine learning field, cross-entropy is commonly used to measure how good the given classification model is. In this research and experimental cases, *p* denotes as target while *q* denotes the attributes toward the target, such as the features in a dataset. If *p* and *q* are the same, this formula calculates the entropy of the variable itself.

III.2. KL Divergence

While the cross-entropy calculates the total entropy between the distributions, a closely related measurement of it is called the Kullback-Leibler (*KL*) divergence that calculates the relative entropy between two probability distributions [9], such that

$$KL(p, q) = \sum_{x \in X} p(x) \log \left(\frac{q(x)}{p(x)}\right).$$

In the continuous case of *x*, it is

$$KL(p, q) = -\int_x P(x) \log \left(\frac{Q(x)}{P(x)}\right) dx .$$

The KL divergence quantifies how much one distribution differs from the other. It concerns with what information content is present if *p* could be generated from *q*. In other words, the KL divergence measures the average number of extra bits required to represent a message with *q* instead of *p*, not the total number of bits of *p* and *q* [6]. It is also noticed that KL-Divergence can never be negative. If *p* and *q* are in the same value, then $\frac{q(x)}{p(x)} = 1$, and since $\log(\frac{q(x)}{p(x)}) = 0$ the $KL(p, q) = 0$, which means that there is no divergence existing between *p* and *q*. If *p* and *q* are not the same, the KL Divergence will represent the divergence of them in terms of the entropy encoded as the minimum average lossless size of the bits of the two distributions [32].

Since the KL divergence requires the computation on $\frac{q(x)}{p(x)}$, it poses a more strict requirement on the completeness of the distributions of the $p(x)$ and $q(x)$ which may not always satisfied in our research because of the sparsity of the binary valued dataset. Therefore the cross-entropy *H(p, q)* was applied in this research with the treatment of the missing values of *x* ignored (setting $p(x) = 0$ or $q(x) = 1$) when computing for $p(x)$ and $q(x)$ in this research.

III.3. Methodologies Involved in this Research

As we already mentioned, a filter-based method is adopted as a mathematical formulation of the process. Applying the cross-entropy as a filter for selecting a set of binary valued features from given datasets, a features having similar distributions with the target attribute are identified as significant features. Since the cross-entropy measures the differences of the two distribution, it also indicates the similarities between the two distributions in the same way. A computational framework called Feature Mapping based Cross-Entropy Selector (FMC_SELECTOR) was thus developed in this research that centers on the use of a Mapping-based Cross-Entropy (MCE) evaluation method on the binary valued features for comparing and identifying a set of significant features. After getting the returning set from MCE, a following up measurement called Positive Prediction Score (PPS) is introduced to verify the selected feature set for classification performance on the given dataset. We describe the framework and its associated computational scheme in the next section.

IV. The FMC_SELECTOR

The general framework of FMC_SELECTOR is shown in Figure 1 below. It consists of three major functional blocks: (1) *Feature Pre-processing* which is further divided into three steps: a. Data cleaning process, b. Removing correlated features, and c. Forming independent candidate set; (2) *Applying MCE algorithm to select significant features*; and (3) *Applying PPS algorithm to verify the selected feature set*. The functional blocks of the framework are described in the following subsections.

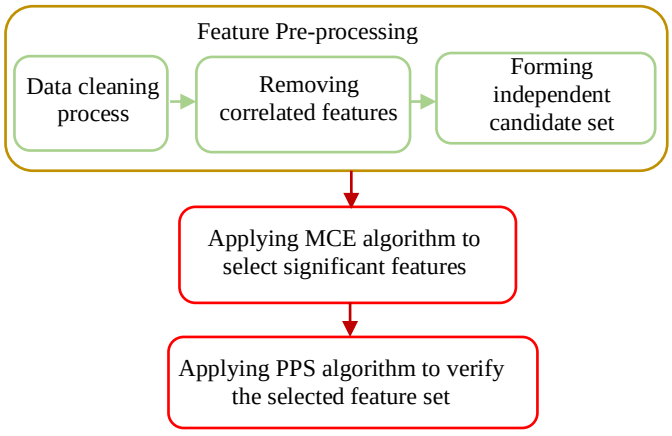


Figure 1. FMC_SELECTOR functional blacks and processing flowchart

IV.1. Feature Pre-processing

The first step in the function black of feature pre-processing is to clean the dataset by removing noisy features and convert non-binary features to binary. Noisy features in this research refers to those features that have a large percentage of missing values or a constant value over all samples. Feature binarization in this research refers to convert the value which are either continuous or categorized to binary. In the datasets for our experimentation, most features are already in binary values.

The second step of the feature pre-processing is to remove the correlated features. Correlated features can be considered as a redundancy of the information, therefore only one feature will be selected from each of the highly correlated feature groups in this research. There are two issues raised from the correlated features when constructing machine learning model. The first issue is multicollinearity. Badr [25] indicated that it happens when one predictor variable in a multiple regression model can be linearly predicted from the others with a high degree of accuracy. This could lead to the misunderstanding and inaccuracy of the machine learning results. Although he suggested to use the decision tree algorithm to avoid this problem, there are some other algorithms that can be applied. The Linear Regression, Naïve Bayes, and Support Vector Machine are used in our experiments for the verification of the features selected. For the second issue of the correlated features, Tolosi and Lengauer indicated that some classical feature selection ideas such as penalized logistic regression or random forest would become unstable in the presence of high feature correlations [26]. It is because parts of the correlated features can be considered as a redundant which will add the unnecessary weights on these parts of the features in the classification processes while the redundant features add no additional information to the dataset. It will make the same efforts as the multicollinearity when building the machine learning model. Therefore, removing it can enhance the accuracy of the machine learning model.

During this pre-processing step, Spearman's rank correlation coefficient was calculated for each pair of the features to identify the correlated feature groups among the features. The formulation of the Spearman's rank correlation coefficient is shown as follows:

$$\rho = 1 - \frac{6\sum d_i^2}{n(n^2-1)};$$

where ρ = the Spearman's rank correlation coefficient,

$d_i = R(x_i) - R(y_i)$ is the difference between the two ranks of each observation - here they are the individual sample values of the binary features x and y,

and

n = number of observations – is the number of samples which equals to the number of rows as the dataset organized in a way such that each row represents the values of a sample over all the features and each column represents the values of all the samples on each particular feature.

After the second step, a set of correlated feature pairs is extract from the original set. The rest of features are then placed into a candidate feature set for selection. Thus, the third step of the pre-processing is to choose the one and the only one most representative feature from feature pairs. It is done by simply applying the Fisher's linear discriminant function (LDF) to each individual feature in the group and select one of them by comparing the LDF scores. By projecting multi-dimensional object to one-dimension, the LDF it able to distinguish the different strengths of the features in terms of their discriminative property, it means that

the feature is more representative for its use in classification. The formula of Fisher's LDF is shown as follows:

$$Q_k = \frac{|m_{1k} - m_k| + |m_{2k} - m_k|}{\frac{1}{n_1}\sum_{j=1}^{n_1} |x_{1k}^j - m_{1k}| + \frac{1}{n_2}\sum_{j=1}^{n_2} |x_{2k}^j - m_{2k}|}$$

Where k is the k^{th} feature of the dataset,

$m_{1k} = \frac{1}{n_1}\sum_{j=1}^{n_1} x_{1k}^j$ - the mean value of feature k for class 1 (target is negative)

$m_{2k} = \frac{1}{n_2}\sum_{j=1}^{n_2} x_{2k}^j$, - the mean value of feature k for class 2 (target is positive)

$m_k = \frac{1}{n_1+n_2}(\sum_{j=1}^{n_2} x_{2k}^j + \sum_{j=1}^{n_1} x_{1k}^j)$ - the mean value of feature k for classes 1 and 2 together

n_1, n_2 – the number of samples for class 1 and class 2, respectively

x_{1k}^j - the value of the feature k on the j^{th} sample of class 1

x_{2k}^j - the value of the feature k on the j^{th} sample of class 2

By the end of the above processing, two feature sets are created. One is a non-selected feature set that contains the non-representative correlated features listed in groups along with the noisy features. The other set is the candidate features that contains both the representative features from the correlated groups and the individual features that do not correlate with any other features. The independent set is to be processed by the MCE algorithm described in the next section. Once the MCE algorithm is applied to select the significant features from the independent feature set, the results will be store in a new feature set called selected feature set.

IV.2. Applying MCE Algorithm to Select Significant Features

As in the common practice, the dataset for feature selection contains multiple columns of attributes that we call them features. Each row of the dataset represents a sample of the data of different feature values. Apart from the features, another single column is labeled as 'target' which indicates what the class or category the sample belongs. In the experiments of this research, the target is also binary valued, that is, a binary classification problem is addressed. We will call the samples with the target value '1' as "positive" samples, and the samples with the target value '0' as "negative" samples.

The computational procedure of the Mapping Based Cross-Entropy Evaluation (MCE) is formulated as follows:

1. First, we calculate the estimated probability distributions of the samples with respect to the positive (value '1') and negative (value '0') labels in the "target", that is

$$p(1) = \frac{Total\ number\ of\ samples\ that\ have\ the\ target\ value\ '1'}{Total\ number\ of\ samples\ in\ the\ dataset},$$

$$p(0) = \frac{Total\ number\ of\ samples\ that\ have\ the\ target\ value\ '0'}{Total\ number\ of\ samples\ in\ the\ dataset}.$$

2. Second, calculate the "Positive Consistency Rate (PCR)" and "Negative Consistency Rate (NCR)" for each individual feature q with respect to the "target," such that

$$PCR(q) = \frac{Total\ number\ of\ positive\ samples\ having\ the\ value\ '1'\ in\ feature\ q}{Total\ number\ of\ positive\ samples\ in\ the\ dataset}$$

$$NCR(q) = \frac{Total\ number\ of\ negative\ samples\ having\ the\ value\ '0'\ in\ feature\ q}{Total\ number\ of\ negative\ samples\ in\ the\ dataset}$$

3. Applying the above four values, the MCE value of a feature q is calculated as

$$MCE(q) = -(p(1)\log(PCR(q)) + p(0)\log(NCR(q)))$$

From the definition of the values involved in the calculation of *MCE(p)*, it is seen that a feature with higher value of *MCE(p)* means the higher difference between the distributions of the feature values and the target values. In contrast, the lower *MCE(p)* value means the distribution of the feature values is more align to the distribution of the target. In other words, the feature is more relevant to the target, thus significant for the classification of the samples with respect to the target.

For a given feature, it is critical to figure out the portion of information real useful to evaluate its relevance with the target. In this research, the $PCR(q)$ and $NCR(q)$ are chosen as the measurement of the relevance where the feature and target have the same value.

By the accomplishment of this computational step, the features with high $MCE(q)$ values are selected from candidate feature set and placed into the selected feature set while the features with low $MCE(q)$ values are put back into the non-selected feature set. In the next step, the experiments of verification steps are conducted to make comparison between the non-selected feature set and selected feature set so as to validate the significance of the features selected.

IV.3. Applying PPS Algorithm to Verify the Selected Feature Set

To evaluate the performance of the selected features in the building of machine learning model from the imbalanced dataset in its applications, we developed and used a Positive-case Prediction Score (PPS). It was noted that the datasets in our experimentation are highly imbalanced and there are no rebalancing operations involved in the entire process of FMC_SELECTOR. It is therefore to evaluate the verification outcomes with respect to the performance (classification rate) on both the positive class samples and the negative class samples without a bias. The PPS aims to evaluate the performance of the machine learning model in detecting positive cases in the datasets, while also counts the negative cases indirectly, that is, a non-biased measurement for the classification performance.

The PPS score in this research is calculated based on the general concept of the confusion matrix for evaluating the classification outcomes or the classifier performances. A confusion matrix is a 2D tabular representation to record the number of correct and wrong predictions from a classification process. For binary classifications, the confusion matrix consists of the following four elements:

- TP - True Positive, which means the actual result is positive and the machine's prediction is positive.

- FP - False Positive, which means the actual result is negative and the machine predicted positive.
- TN - True Negative, which means the result is actually negative and the machine predicts negative.
- FN - False Negative, which means the actual result is positive, but the machine predicted negative.

It is known that the common classifier performance evaluators such as the F1 score and the Matthews Correlation Coefficient (MCC) are biased measurements with respect to imbalanced datasets [39]. These measurements are therefore not suitable for an accurate evaluation of the discriminative performance of the features selected in the verification stage of this research because both the experimental datasets of this research are both highly imbalanced.

The PPS takes the "True Positive Rate (*TPR*)" as a gain and counts the "False Positive Rate (*FPR*)" as a penalty to calculate the discriminant outcome measurement for a feature q in a way such that $PPS(q) = TPR(q) - FPR(q)$, where the *TPR* and *FPR* are two combinational rates defined on the outcomes represented by the general confusion matrix of a classification process (a machine learning model) instead of on the labels of the target. That is

$$TPR(q) = \frac{Total\ number\ of\ samples\ classified\ as\ positive\ while\ their\ target\ value\ is\ '1'}{Total\ number\ of\ positive\ samples\ in\ the\ dataset}$$

and

$$FPR(q) = \frac{Total\ number\ of\ samples\ classified\ as\ positive\ while\ their\ target\ value\ is\ '0'}{Total\ number\ of\ negative\ samples\ in\ the\ dataset}$$

Note that for a given dataset, $TPR(q) = 1 - FNR(q)$ and $FPR(q) = 1 - TNR(q)$, where

$$FNR(q) = \frac{Total\ number\ of\ samples\ classified\ as\ negative\ while\ their\ target\ value\ is\ '1'}{Total\ number\ of\ positive\ samples\ in\ the\ dataset}$$

and

$$TNR(q) = \frac{Total\ number\ of\ samples\ classified\ as\ negative\ while\ their\ target\ value\ is\ '0'}{Total\ number\ of\ negative\ samples\ in\ the\ dataset}$$

By taking the advantage of the difference between the *TPR* and *TNR*, the *PPS* overcomes the bias of the simple classification rate on either the positive samples or negative samples. That is, it addresses the issue of imbalance of the dataset and makes an evaluation of the classifier performance on a balanced basis for both the positive and the negative classes with respect to the highly imbalanced presence of the samples in the dataset. The PPS values in range from -1 to +1, such that -1 indicates a total miss of classification with the selected features for any samples and +1 means a 100% correct classification for all the samples in the dataset, while a zero values represents a 50% accuracy for samples of both classes.

It was known that different machine learning and classification algorithms have different functional characteristics and performance outcomes. Choosing multiple classification algorithms in the feature verification stage of our feature selection process and comparing their overall performance would generate a more reliable and justifiable result, therefore, helping evaluating the effectiveness of the FMC_SELECTOR method. Thus, three popular classification algorithms, the K-Nearest Neighbor (KNN) [22], Linear Regression [23], and Naive Bayes [24][38] were selected for evaluating the feature selection results in our experiments. These algorithms are chosen mainly due to their simplicity and effectiveness proven by other researchers in machine learning practices.

V. Experiments and Results

V.1. Results and Analyses on 'Data Colorectal 08 to 13 Weighted' Dataset

The first data set we experimented with is called the "Data Colorectal 08 to 13 Weighted" which is a dataset queried from the Healthcare Cost and Utilization Project National Inpatient Sample (HCUP-NIS) database for adult patients with a diagnosis of colorectal cancer who underwent colorectal resection [37]. The dataset contains a total of 77,603 instances (the rows) but only 4.55% of them are positive cases i.e., the dataset is very highly imbalanced. Moreover, for most of the cases there are only 1 or 2 non-zero values among the 29 major attributes (the columns) of this study. That is the data is also very highly sparse (sparsity = 90.35%).

In the pre-processing step, the correlation computation function corr() in the SKLearn package with Spearman rank correlation method [3] was utilized to obtain the correlation coefficients for each pair of the features. The result is shown in figure 2 below where the darkness of the cells in the plot represents the relative values of the coefficients with white being high (most correlated) and black being low (less correlated) for the correlation. As we can see from the figure that the features of this dataset have an overall low correlation with each other.

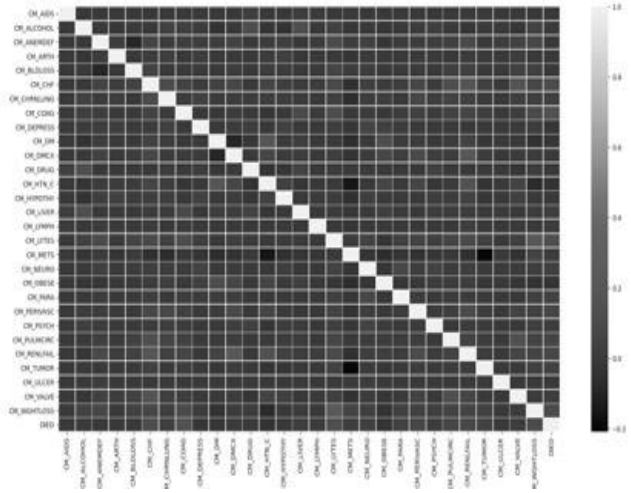


Figure 2. Correlation map for features in "Data Colorectal 08 to 13 Weighted" dataset

During the feature selection process, all the samples in the dataset are shuffled randomly for both the training set and testing set. After applying our MCE algorithm directly on the features of the dataset, all the features are ranked according to the $MCE(q)$ values. Table 1 below shows the top 9 features selected by the MCE algorithm on the dataset, with an order of from most significant to less significant (numbering 1 to 9) according to the $MCE(q)$ values (due to the nature of the cross-entropy computation, lower values of $MCE(q)$ means the higher level of significance of the feature q.).

Rank	**Feature**	MCE value
1	CM_COAG	0.128306
2	CM_CHF	0.153914
3	CM_LYTES	0.156126
4	CM_WGHTLOSS	0.159834
5	CM_PULMCIRC	0.161825
6	CM_RENLFAIL	0.165845
7	CM_PERIVASC	0.166367
8	CM_PARA	0.167449
9	CM_TUMOR	0.180618

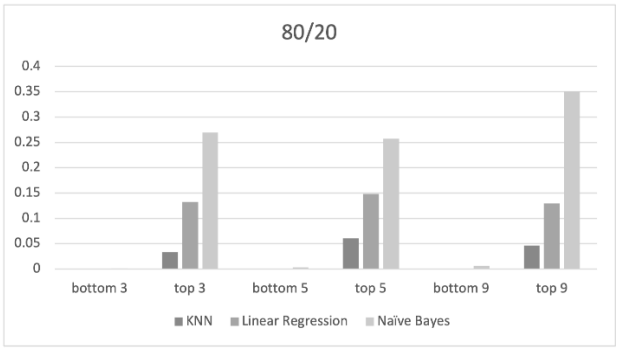
Table 1. Feature selection result and MCE values for the experiment dataset

The above results were compared with those obtained by two benchmarks, the Univariate Feature Selection method, and the Recursive Feature Elimination method [13]. The results are listed in the table 2 below. The table shows that the MCE algorithm selected almost the same group of features as the benchmark methods but in different orders.

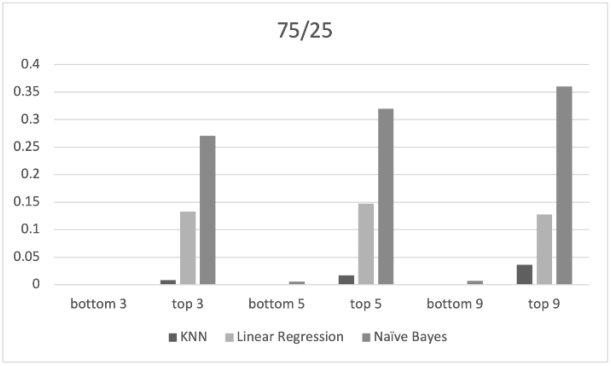
Rank	MCE Algorithm	Univariate Selection	Recursive Feature Elimination
1	CM_COAG	CM_COAG	CM_CHF
2	CM_CHF	CM_LYTES	CM_COAG
3	CM_LYTES	CM_CHF	CM_LYTES
4	CM_WGHTLOSS	CM_WGHTLOSS	CM_PARA
5	CM_PULMCIRC	CM_PERIVASC	CM_PERIVASC
6	CM_RENLFAIL	CM_PULMCIRC	CM_PULMCIRC
7	CM_PERIVASC	CM_RENLFAIL	CM_RENLFAIL
8	CM_PARA	CM_TUMOR	CM_ULCER
9	CM_TUMOR	CM_PARA	CM_WGHTLOSS

Table 2. Comparison of features selected by the MCE algorithm with benchmark methods

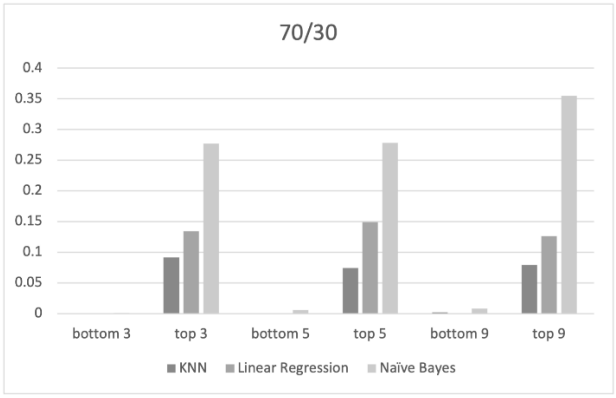
To validity the significance of the selected features, we applied the PPS approach to the MCE results. Three different pattern classification methods were used in PPS as discussed in last section, and were applied to different groups of combinations of the selected features versus non-selected features from the MCE algorithm. The splitting ratios of 80/20, 75/25, and 70/30, respectively, were used for setting up the training set and testing set on the dataset. Figure 3 shows the PPS collections as the test results in terms of the classification rates in comparison with the use of (1) the top 3 features selected by our MCE algorithm versus the bottom 3 features, (2) the top 5 features versus the bottom 5 features, and (3) the top 9 features versus the bottom 9 features. The results of these test cases were shown from left to right on each of the plots with the different classification algorithms and under the different settings of the training and test sets, respectively.



(a) For 80 to 20 split of training and test data



(b) For 75 to 25 split of training and test data



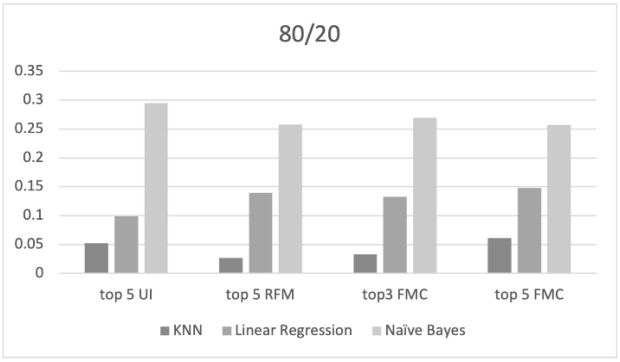
(c) For 70 to 30 split of training and test data

Figure 3. Comparison of PPS for feature groups selected (top3 or 5) by the MCE algorithm with those not selected (bottom 3, 5, or 9) features on the "Data Colorectal 08 to 13 Weighted" dataset

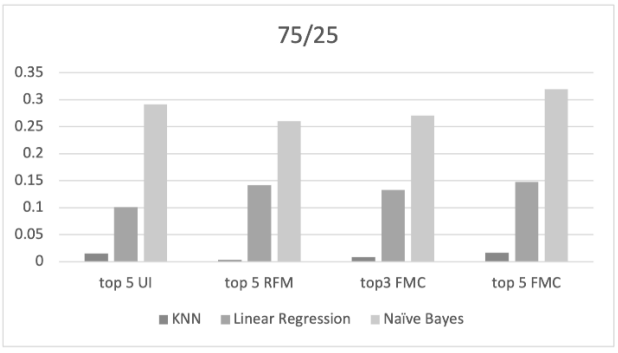
From the figure 3, it is seen that the features selected by the MCE algorithm performed much better than the non-selected features. The result verifies that the FMC_SELECTOR framework was able to select the significant features and helped to generate better machine learning models for the given dataset.

In addition to the comparison of the performance of the features among those selected and not selected by our FMC_SELECTOR as shown in the figure 3 above, tests for the comparison of our method (marked FMC) with the two benchmarks feature selection methods, namely the Univariate Feature Selection (marked as UI) and the Recursive Feature Elimination (marked as RFM) were conducted with the PPS results shown in figure 4. Note that though our FMC method selected the same top 9 features from the dataset as that did by the UI and RFM methods, the order of these top features were different from each other, as shown in table 2 above. We thus selected the top 3 features and top 5 features of the FMC

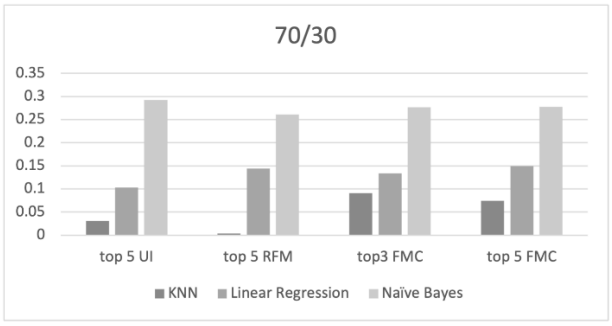
method as representatives to compare the performance with the top 5 features from the UI and RFM methods. The PPS for these teat cases are shown in plots with 80 to 20, 75 to 25, and 70 to 30 splitting of the samples in the datasets shown in the (a), (b), and (c) of figure 4, respectively. As it can be seen from the figure that the overall performance of the FMC_SELECTOR results, though somehow mixed, is mostly comparable with the other two benchmarks, with better results shown in some of the test cases.



(a) For 80 to 20 split of training and test data



(b) For 75 to 25 split of training and test data



(c) For 70 to 30 split of training and test data

Figure 4. Comparison of PPSs for different groups of the top 3 or 5 features selected by the MCE algorithm with the benchmark methods on the "Data Colorectal 08 to 13 Weighted" dataset.

V.2. Results and Analyses on 'Kddcup99_csv' Dataset

"Kddcup99_csv" is another dataset we used for testing and verifying our MCE algorithm and the FMC_SELECTOR framework. The "Kddcup99_csv" was first posted and used in the Third International Knowledge Discovery and Data Mining Tools Competition. There were 32 features and 1 target with 97,277 positive samples and 2,204 negative samples used in our research experiment. Doing the same as for the last experiment dataset, the correlation coefficients of the features of this dataset were obtained first in the pre-processing step by applying the Spearman rank correlation method. The results are shown in figure 5 below. Criterion for determining whether two features are correlated in this research is set for the correlation coefficient corr(x, y) being either >= 0.6 or <= -0.6. Five correlated feature groups were identified in this dataset from the computation results, as shown in table 3 below. In table 3, the most representative features from each correlated feature group are highlighted in bold. Those features are add to the candidate feature set while the others were placed into the non-selected feature set. Features with high coefficient values, with constant values for all samples, and without any true positive samples, such as the "wrong_fragment," "Inum_outbound_cmds," and "is_host_login," were removed from the candidate list therefore not consider in the further steps of the feature selection process.

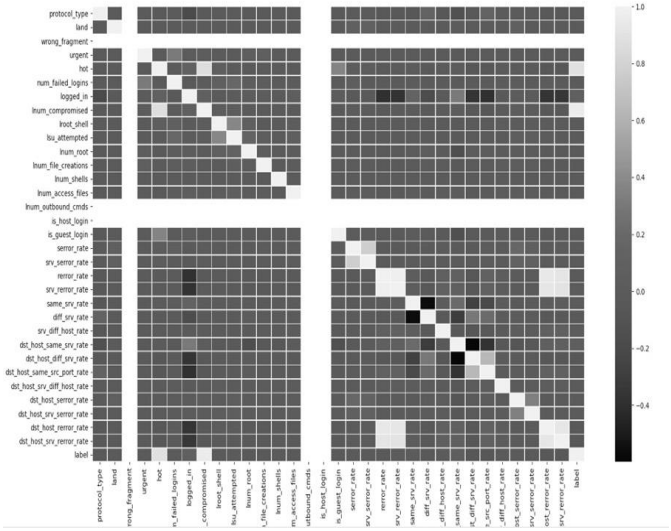


Figure 5. Correlation map for features in 'Kddcup99_csv' dataset

Correlated feature group 1	hot	**Inum_compromised**		
Correlated feature group 2	**Srv_serror_rate**	serror_rate		
Correlated feature group 3	Rerror_rate	Srv_rerror_rate	**Dst_host_error_rate**	Dst_host_srv_serror_rate
Correlated feature group 4	**Same_srv_rate**	Diff_srv_rate		
Correlated feature group 5	**Dst_host_same_srv_rate**	Dst_host_diff_srv_rate	Dst_host_same_src_port_rate	

Table 3. Correlated feature groups identified from the correlation coefficient computation. The most representative features from each group are highlighted in bold.

The *MCE*(q) computations were carried out on each feature q in the candidate set after the pre-processing. Table 4 below shows the top 9 features selected by the MCE algorithm on the "Kddcup99_csv" dataset, in ranks according to the *MCE*(*q*) values.

Rank	**Feature**	MCE value
1	lnum_compromised	0.002077997
2	logged_in	0.111462721
3	srv_serror_rate	0.11376247
4	dst_host_same_srv_rate	0.117019266
5	serror_rate	0.118823295
6	same_srv_rate	0.121410364
7	srv_diff_host_rate	0.150033709
8	dst_host_rerror_rate	0.229146162
9	dst_host_srv_rerror_rate	0.230147076

Table 4. Feature selection result for "Kddcup99-csv" dataset

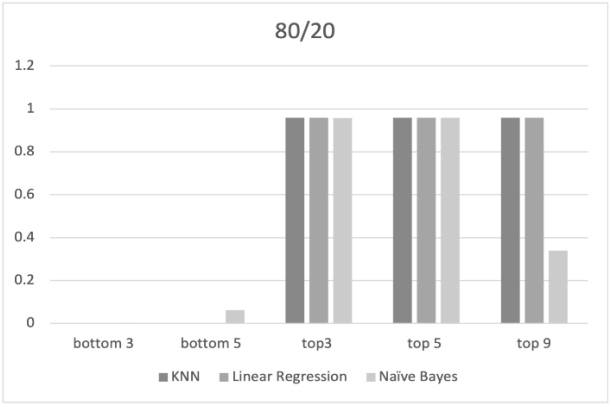
Table 5 below compares the top 9 features selected by the MCE algorithm with those obtained by the two benchmark methods, the Univariate Feature Selection Method and the Recursive Feature Elimination Method, respectively.

	MCE Algorithm	Univariate Selection	Recursive Feature Elimination
1	lnum_compromised	urgent	num_failed_logins
2	logged_in	lroot_shell	lnum_compromised
3	srv_serror_rate	num_failed_logins	lroot_shell
4	dst_host_same_srv_rate	lnum_compromised	lsu_attempted
5	serror_rate	is_guest_login	lnum_shells
6	same_srv_rate	serror_rate	srv_serror_rate
7	srv_diff_host_rate	dst_host_same_src_port_rate	diff_srv_rate
8	dst_host_rerror_rate	dst_host_serror_rate	dst_host_same_src_port_rate
9	dst_host_srv_rerror_rate	dst_host_srv_diff_host_rate	dst_host_rerror_rate

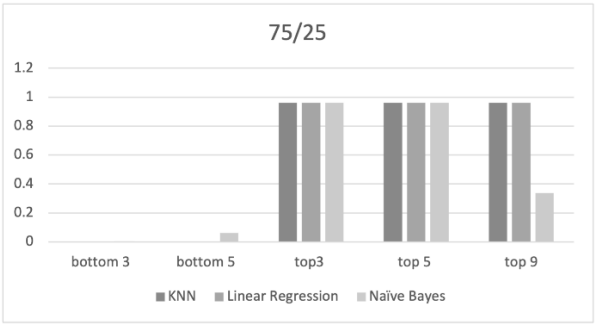
Table 5. Features selected on "Kddcup99_csv" by MCE algorithm in comparison with benchmark methods

Again we see that top 9 features selected by the MCE algorithm have a large percentage of overlap with those features selected by the comparing benchmark methods, but in different rank positions.

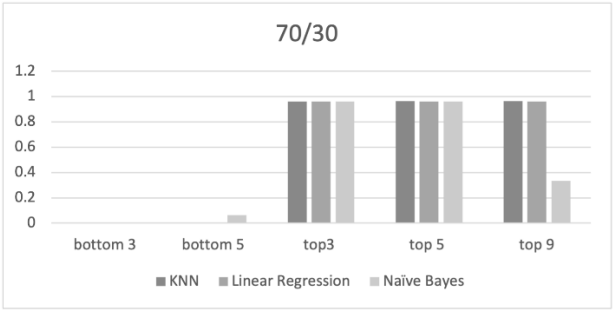
Validation of the significance of the selected features with respect to the non-selected features was conducted first by applying the PPS evaluations. We compared the top 3, top 5, and top 9 features selected by the MCE algorithm with the bottom 3 and bottom 5 of non-selected features in the experimentation. The results are shown in Figure 6 where the PPSs for the non-selected features are plotted by the left and the PPSs for the selected feature are plotted by the right of each plot. The verification is again done with the tests on applying the three classification models: KNN, Linear Regression, and Naïve Bayes, and with the dataset splitting of 80 to 20, 75 to 25, and 70 to 30, respectively. It is obvious from the figure that the classification performance with the use of the selected features significantly outperforms the results from the non-selected features.



(a) For 80 to 20 split of training and test data



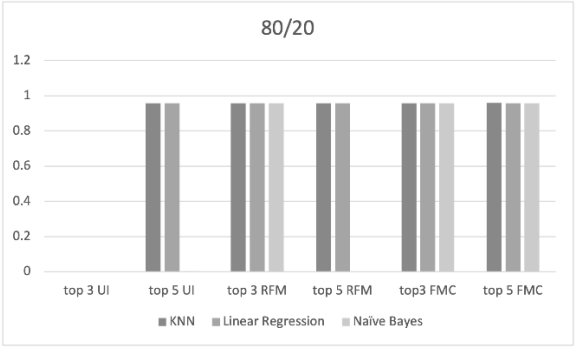
(b) For 75 to 25 split of training and test data



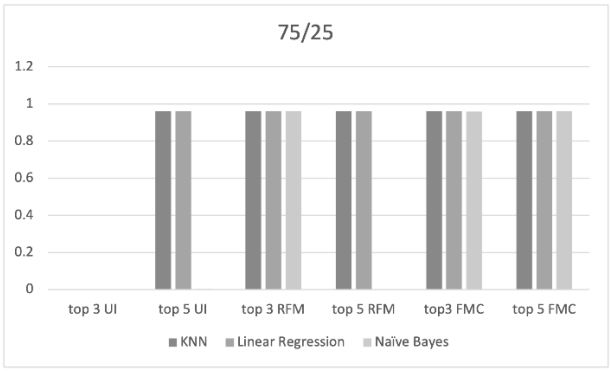
(c) For 70 to 30 split of training and test data

Figure 6. Comparison of PPS for feature groups selected (top3 or 5) by the MCE algorithm with those not selected (bottom 3, 5, or 9) features on the "Kddcup99_csv" dataset

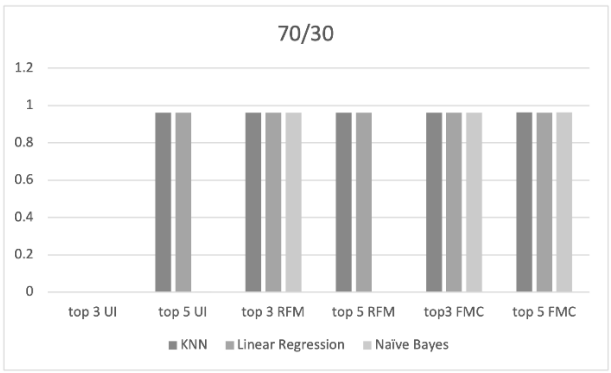
Verification experiment was also conducted to compare the PPS scores of the MCE algorithm with the two benchmark methods on the top ranked features selected. The PPS data in Figure 7 shows that the features selected from the MCE algorithm performed very compatible with those from the two-benchmark methods, though not appearing to be in a clear advantage over the other two on the "Kddcup99_csv" dataset.



(a) For 80 to 20 split of training and test data



(b) For 75 to 25 split of training and test data



(c) For 70 to 30 split of training and test data

Figure 7. Comparison of PPSs for different groups of the top 3 or 5 features selected by the MCE algorithm with the benchmark methods on the "Kddcup99_csv" dataset.

The verification results with the use of the two experimental datasets do indicate that the FMC_SELECTOR framework for feature selection is proven to be a viable method and in right direction for ranking binary valued feature in terms of their significance for improving model accuracy selection of classifications on the datasets, though further research and development are needed to improve its performance.

VI. Summary

The need to know the cause and effect relations of data entities in many real world applications calls for computational mechanisms dedicated to the identification of the dominant features in a dataset in addition to the high accuracy of pattern classifications and model predictions. Sparsely populated and binary valued features present a challenge to the accurate computation of the discriminative significances of these features, especially for the class sample size highly imbalanced datasets. There was a lack of previous research on specific methods working with binary valued features and imbalanced datasets. The research presented in this paper thus addressed this concern and introduced a new method that was focused on an quantitative measurement of the discriminative significances of the features so as to provide a rank and a selection of the features that can be used further to improve the accuracy of the machine learning models to be built on the binary features of the dataset.

The cross-entropy based formulation of the MCE computational scheme was successfully applied within the framework of FMC_SELECTOR to establish the coherent relations between the features of pattern classes and the target attributes under the conditions of the imbalance distribution and sparsity of the binary valued features. An evaluation mechanism named PPS was introduced in this research to analyze the performance of machine learning models generated by the selected features and to verify the significance of these features selected by the MCE algorithm. The verification process of the feature selection focused on the classification performance with PPS applied as a balanced measurement on both the positive class and negative class samples to overcome the biases of some other common classifier evaluators. The comparison and analysis of the PPS outcomes with the features selected by the benchmark methods show that the FMC_SELECTOR was effective in terms of to identify the most significant features in the given experiment datasets.

References

[1] L. Blum and P. Langley, "Selection of relevant features and examples in machine learning", *Artificial Intelligence*, Volume 97, Issue 1, pp. 245-271, 1997.

[2] K. Kira and L. A. Rendell, "A Practical Approach to Feature Selection," *Proceedings of the Ninth International Workshop on Machine Learning* (ML92), San Francisco, USA, pp. 249 – 256, 1992.

[3] J. L. Myers and A. D. Well, *Research Design and Statistical Analysis* (2nd ed.), Lawrence Erlbaum Associates Publishers, ISBN 978-0-8058-4037-7, 2003.

[4] M. J. Post, P. Putten, and J. N. Rijin, "Does Feature Selection Improve Classification? A Large-Scale Experiment in OpenML", *Proceedings of Advances in Intelligent Data Analysis XV*, Germany, pp. 158-170, 2016.

[5] P. A. Devijver and J. Kittler, *Pattern Recognition: A Statistical Approach*, *Prentice/Hall International*, ISBN: 0136542360, 1982.

[6] M. Bishop, *Pattern Recognition and Machine Learning*, Springer, ISBN-0387310738, 2006.

[7] K. Chotchantarakun and O. Sornil, "An Adaptive Multi-levels Sequential Feature Selection," *International Journal of Computer Information Systems and Industrial Management Applications*, ISSN 2150-7988 Volume 13,) pp. 010-019, 2021.

[8] A. M P Canuto and K. M O Vale, and Antonino Feitosa, "A Reinforcement-based Mechanism to Select Features for Classifiers in Ensemble Systems," *International Journal of Computer Information Systems and Industrial Management Applications*, ISSN 2150-7988 Volume 3, pp. 324 -335, 2011.

[9] J. Brownlee, "A Gentle Introduction to Cross-Entropy for Machine Learning," https://machinelearningmastery.com/cross-entropy-for-machine-learning/ (as of September 21, 2021).

[10] S. K. Gajawada, "ANOVA for Feature Selection in Machine Learning," https://towardsdatascience.com/anova-for-feature-selection-in-machine-learning-d9305e228476 (as of September 21, 2021).

[11] T. Hopper, "Cross Entropy and KL Divergence", https://tdhopper.com/blog/cross-entropy-and-kl-divergence (as of September 21, 2021).

[12] C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, Volume 27, Issue 3, pp. 379-423, 1948.

[13] F. Pedregosa, *et al*., "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, Volume 1, Issue 1, pp. 2825–2830, 2011.
[14] L. Breiman, "Random Forest," January 2001, https://www.stat.berkeley.edu/~breiman/randomforest2001.pdf, (as of September 21, 2021).
[15] S. Visalakshi and V. Radha, "A literature review of feature selection techniques and applications: Review of feature selection in data mining," *Proceeding of 2014 IEEE International Conference on Computational Intelligence and Computing Research, Coimbatore*, India, pp. 1-6, 2014.
[16] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *The Journal of Machine Learning Research*, Volume 3, pp. 1157–1182, 2003.
[17] M. A. Hall, "Correlation-based feature selection for machine learning," Thesis, Department of Computer Science, Waikato University, 1999.
[18] J. Brownlee, "Information Gain and Mutual Information for Machine Learning," https://machinelearningmastery.com/information-gain-and-mutual-information/ (as of October 16, 2019).
[19] Q. Gu, Z. Li, and J. Han, "Generalized fisher score for feature selection," *Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence*, Arlington, United States, pp. 266-273, 2011.
[20] R. J. Urbanowicz, M. Meeker, W.L. Cava, R.S. Olson, and J.H. Moore, "Relief-based feature selection: Introduction and review," *Journal of Biomedical Informatics*, Volume 85, pp. 189-203, 2018.
[21] N. K. Suchetha, A. Nikhil, and P. Hrudya, "Comparing the Wrapper Feature Selection Evaluators on Twitter Sentiment Classification," *Proceedings of 2019 International Conference on Computational Intelligence in Data Science* (ICCIDS), Chennai, India, pp. 1-6, 2019.
[22] M. Sarkar and T. Y. Leong, "Application of K-nearest neighbors' algorithm on breast cancer diagnosis problem," *Proceedings of AMIA Symposium, Journal of the American Medical Informatics Association*, pp. 759–763, 2000.
[23] M. P. Deisenroth, A. A. Faisal and C. S. Ong, *Mathematics for Machine Learning*, Cambridge University Press. ISBN: 9781108679930, 2020.
[24] M. N. Murty, and V. S. Devi, *Pattern Recognition: An Algorithmic Approach*, Springer Science & Business Media, ISBN 978-0857294944, 2011.
[25] W. Badr, "Why Feature Correlation Matters… A Lot!" https://towardsdatascience.com/why-feature-correlation-matters-a-lot-847e8ba439c4 (as of Jan 18, 2019).
[26] L. Tolosi and T. Lengauer, "Classification with correlated features: unreliability of feature ranking and solutions," *Bioinformatics*, Volume 27, Issue 14, 2011, pp. 1986–1994.
[27] B. Xue, M. Zhang, W. N. Browne, and X. Yao, "A Survey on Evolutionary Computation Approaches to Feature Selection," *IEEE Transactions on Evolutionary Computation*, volume 20, issue 4, pp. 606-626, 2016.
[28] H. Liu, J. Sun, L. Liu, and H. Zhang, "Feature selection with dynamic mutual information," *Pattern Recognition*, Volume 42, Issue 7, pp. 1330-1339, 2009.
[29] A. M. Anter and M. Ali, "Feature selection strategy based on hybrid crow search optimization algorithm integrated with chaos theory and fuzzy c-means algorithm for medical diagnosis problems," *Soft Computing*, Volume 24, pp. 1565 – 1584, 2019.
[30] A. Askarzadeh, "A novel metaheuristic method for solving constrained engineering optimization problems: Crow search algorithm," *Computers & Structures*, Volume 169, pp. 1 - 12, 2016.
[31] M. Xue, W. Zhang, N. Browne, and X. Yao, "A Survey on Evolutionary Computation Approaches to Feature Selection," *IEEE Transactions on Evolutionary Computation*, volume 20, issue 4, pp. 606-626, 2016.
[32] N. Shibuya, "KL Divergence Demystified", https://naokishibuya.medium.com/demystifying-kl-divergence-7ebe4317ee68 (as of November 5, 2021).
[33] A. Nagpal, "L1 and L2 Regularization Methods," https://towardsdatascience.com/l1-and-l2-regularization-methods-ce25e7fc831c, (as of September 21, 2021).
[34] P. Bannigidad and A. Deshpande, "The Fusion of Features for Detection of Clinical Symptoms of Diabetic Retinopathy and its Grading from Digital Fundus Images," *International Journal of Computer Information Systems and Industrial Management Applications*, ISSN 2150-7988, Volume 13, pp. 172-181, 2021.
[35] S. R Rathod and C. Y Patil, "Linear and non-linear HRV features for the prediction of heart disease among smokers: a predictive evaluation of machine learning model," *International Journal of Computer Information Systems and Industrial Management Applications*, ISSN 2150-7988, Volume 12, pp. 222-230, 2020.
[36] O. Setyawati, A. S. Arifianto and M. Sarosa, "Feature selection for the classification of clinical data of stroke patients," *Proceedings of 20th International Conference on Electrical Machines and Systems* (ICEMS), Sydney, Australia, pp. 1-4, 2017.
[37] S. Bonthu, P. Rodrigues-Armijo, T. Tanner, Q. Zhu, "Machine Learning to Improve Surgical Outcomes," *Proceedings of the 18th IEEE International Conference on Machine Learning and Applications* - ICMLA 2019, pp. 1426-1431, December 16-19, 2019.
[38] M. Danziger and F. B. de Lima Neto, "A Hybrid Approach for IEEE 802.11 Intrusion Detection Based on AIS, MAS and Naïve Bayes," *International Journal of Computer Information Systems and Industrial Management Applications*, ISSN 2150-7988, Volume 3, pp. 193-201, 2011.
[39] Q. Zhu, "On the Performance of Matthews Correlation Coefficient (MCC) for Imbalanced Dataset," *Pattern Recognition Letters*, Vol. 136, pp. 71-80, 0167-8655/©2020, Elsevier B.V., https://doi.org/10.1016/j.patrec.2020.03.030, 2020.

Author Biographies

Zhipeng Wang - Received the M.Sc. degree in computer science from the University of Nebraska at Omaha, Omaha, in 2021. He currently works as a software engineer at Mckinsey & Company. His current research interests include machine learning, pattern recognition, data visualization, big data, and software engineering..

Qiuming Zhu - Professor of computer science at the University of Nebraska at Omaha. Received his Ph.D. in computer and systems engineering from the Rensselaer Polytechnic Institute, Troy, New York, in 1986. His research areas are in digital image processing and computer vision, pattern recognition, data mining and knowledge discovery, and intelligent decision support systems.