TOWARDS A GENERAL TECHNIQUE FOR TRANSFORMATION OF NOMINAL FEATURES INTO NUMERIC FEATURES IN SUPERVISED LEARNING

[1]Eftim Zdravevski [2]Petre Lameski [3]Andrea Kulakov
Department of Information Systems, Faculty of Computer Sciences and Engineering
University Ss. Cyril and Methodius
Skopje, Macedonia
[1]eftim@finki.ukim.mk, [2]lameski@finki.ukim.mk, [3]andrea.kulakov@finki.ukim.mk

ABSTRACT

Almost all of the machine learning problems require data preprocessing. This stage is especially important for problems where the datasets contain features of mixed types (i.e. nominal and numeric). An often practice in such cases is to transform each nominal features into many dummy (i.e. binary) features. Also many classification algorithms have preference of numeric attributes over nominal attributes, and sometimes the distance between different data points cannot be estimated if the values of the attributes are not numeric and normalized. One way to transform nominal into numeric features is to use the Weight of Evidence (WoE) technique. WoE has some properties that make it very useful tool for transformation of attributes, but unfortunately there are some preconditions that need to be met in order to calculate it. Additionally WoE originally works only on supervised learning problems where data is labelled with two classes. In this paper we propose modified calculation of the Weight of Evidence that overcomes these preconditions, and additionally makes it usable for test examples that were not present in the training set. The proposed transformation can be used for all supervised learning problems and arbitrary number of classes. This paper establishes the theoretical background for such modifications, and does not present any comparative results with other similar techniques.

Keywords: Weight of evidence, WoE, data transformation, nominal features, numeric features, smoothing, multiclass supervised learning

I. INTRODUCTION

In almost every classification task, the data preprocessing phase is the most time consuming, because it is closely related to the data itself, and as a consequence it can be applied in different ways. When first presented with a data set, many statisticians and analysts think that they are able to use the data directly, but this is unfortunately not the case. The first step should be to analyze the data, and then to transform them into something usable. Data transformations are used to normalize the distribution of the values of an attribute.

Although there are general guidelines about how to process and transform specific kinds of data, the same transformations are not applicable for all attributes, even if they are of the same data type. In [1] and [2] are given some of the methodologies for data transformations.

Probably the most widely used technique for transformation of nominal features is introducing dummy (indicator) variables (features). As described in [3], a dummy variable is a simple and useful method of introducing into a regression analysis information contained in variables that are not conventionally measured on a numerical scale (e.g. race, sex, region, occupation, etc.). In other words, it is an artificial variable that has the value of one whenever the category it represents occurs, and zero otherwise. At first this method was used mostly in regression analysis, but when other more sophisticated machine learning methods were developed it was also used in them. This is due to the fact that transformation of nominal attributes into numeric with dummy variables is very easy to implement, and the interpretation of the dummy values is straightforward. However, there are limitations to this method. Namely the number of dummy variables that are generated from each nominal variable grows in linear fashion with the number of different values of the variable. In some cases, if an appropriate feature selection technique is applied, some of the dummy variables can be discarded. Despite this fact, the potential number of retained variables can be very large and therefore introduce computational challenges to many machine learning algorithms. The increased number of features should be addressed carefully especially if it has a significant effect on the algorithm's running time or memory requirements.

In text mining problems usually the term frequency-–inverse document frequency (TF--IDF) [4] weight is used as a numerical statistic which reflects how important a word is to a document in a collection or corpus. In a similar manner this weight can be used to transform arbitrary nominal values into numerical values just as it assigns weight to words in text mining and information retrieval. However, the applicability of this approach in data transformation and its performance in relation to other techniques has not been well investigated and documented yet.

The Weight of Evidence (WoE) is one of the tools used for the transformation of nominal attributes into continual attributes in labelled data sets i.e. in supervised learning problems. WoE has some properties that make it a very useful tool for transformation of attributes, but unfortunately there are some preconditions that need to be met in order to calculate the WoE. Additionally, in its original form it is only applicable to binary classification problems. In this paper we will present some enhancements to WoE, which can extend its applicability and overcome most of its obstacles.

II. Weight of evidence

Every day we make decisions based on the probability of some event to occur. Some situations are more trivial, as well as the decisions associated with them. For example, one can decide whether to take an umbrella based on how the weather looks like, or based on the weather forecast. Other decisions require information from multiple sources and are more complex. Regardless of the complexity of the situation, usually the probability of an outcome is far from empirical as it depends on more facts, which could have complex inter dependencies [5]. For each decision we determine the circumstances that are associated with it and the weight of the facts. Basically, this maps the risk associated with a particular choice or a fact on a linear scale, which aids the human brain in assessing the risk.

The idea of numerically weighting evidence was first published in [6] and is a result of the work by performed by Alan Turing and I.J. Good during World War II. In statistics weight of evidence is defined as a quantitative method for evaluating the evidence in support of a hypothesis. This method has been extensively applied in medical diagnosis, where the evidence consisted of a set of symptoms and the hypothesis was in the form of ``the patient has disease X". Then this evidence is weighted depending on the pattern of associations between the symptoms and the disease. This weighting is usually performed with the parameter named Weight of Evidence (WoE) [1]. It is a fairly simple parameter, but yet has a good mathematical background, which makes it a great tool for assessing the relative risk based on the available information. The remainder of the current section describes the mathematical background for calculation of WoE for binary classification problems as it is described in [7]:

$$WoE_i^A = \ln\left(\frac{\frac{N_i^A}{SN}}{\frac{P_i^A}{SP}}\right) = \ln\left(\frac{N_i^A}{P_i^A}\right) - \ln\left(\frac{SN}{SP}\right) \qquad (1)$$

Where SN and SP are defined with (2) and (3), respectively:

$$SN = \sum_{i=1}^{n^A} N_i^A \qquad (2)$$

$$SP = \sum_{i=1}^{n^A} P_i^A \qquad (3)$$

Equation (1) defines the weight of evidence (WoE) of the *i*-th value of the variable *A*, where N_i^A is the number of data points (i.e. instances) that were labelled as negative, and P_i^A is the number of data points that were labelled as positive for the *i*-th value of the variable *A*. *SN* is the total number of negatively labelled data points, *PN* is the total number of positively labelled data points in the training set, and n^A is the number of different values for the variable *A*.

The second part in (1) illustrates that WoE is consisted of two components: a variable component which relates to the group of data points that have a particular value of variable *A*, and a constant component which relates to the whole sample i.e. to the training data set. These numbers are calculated during the preprocessing phase of the data and do not depend on the machine learning algorithm that is going to be applied later on.

Equation (1) implies that the values for N_i^A and P_i^A have to be different than zero, and given that they represent counts, these constraints transform to $N_i^A > 0$ and $P_i^A > 0$. Later in this section an implementation that overcomes these constraints will be described.

Example 1: Let a binary training set contain 10000 negatively labelled and 40000 positively labelled data points, making a total of 50000 data points. Let one of the variables be ``Age" (in years) and let there be 70 different values for this variable. We want to determine the WoE for the ``Age" variable when its value is 20 (years), given that there are 1000 instances that have the value 20 for the attribute ``Age", from which 700 are negatively labelled, and 300 are positively labelled concerning the classification outcome. All needed parameters are stated in Table 1, and by using (1), and (2) and (3) we can calculate WoE_i^{Age}. This is illustrated by (4):

$$WoE_i^{Age} = \ln\frac{700}{300} - \ln\frac{40000}{10000} = -0.539 \qquad (4)$$

The obtained value shows that the particular group, to which the WoE is applied, is with a higher risk than the average, i.e. it has a negative weight of evidence. The WoE for any group with average odds is zero, because the constant and the variable portion of (1) would be approximately the same.

The WoE has a linear relationship with the logistic function, which makes it a well-suited tool for feature transformation when logistic regression is used.

There are few main reasons why WoE is a useful measure:

- It provides an easy and intuitive estimate of the relative risk of the different values of a particular variable.
- Indicates the more and less risky groups of values.
- It can be used as a very practical tool for easy transformation of the variables from one type to another one. This is particularly useful for transforming multi valued nominal variables into numerical variables.
- After the transformation of the variables is performed, the groups of values with similar relative risk could be easily distinguished. This property could be used for binning multiple values into fewer coarse groups. Using this property, the attribute ``Age" in Example Example 1 could be reduced so it would have significantly less different groups. Using this binning, one group would correspond to multiple values that have

similar relative risk, and the group would be represented by the average WoE of the values in that group.

- The information value (i.e. the predictive power) of an feature could be estimated using the WoE of its values, as discussed in [1].

In opposition to the useful characteristics of WoE, there are few that are, in fact, drawbacks. This is something that needs to be addressed properly before applying the WoE transformation to a variable:

- WoE does not consider the proportion of data points with a particular value of an variable, only the relative risk. In Example 1, $P_i^{Age} = 300$, $N_i^{Age} = 700$ and $WoE_i^{Age} = -0.539$. The same value for WoE would be obtained if $P_i^{Age} = 3$ and $N_i^{Age} = 7$, even though these cases are very different in terms of the proportion of data points from the whole data set. This issue has to be addressed with other statistical techniques.
- WoE measures discriminability of a single variable, but would not capture the discriminability of a variable in combination with another. For instance, the variable *X* may generally have a low WoE for its value *A*, but when a second attribute *Y* has a value *B*, the combination *X=A* and *Y=B* may have high WoE and become very useful for classification. This means, that giving *X=A* a low WoE, should be done carefully, because sometimes this is not appropriate. A possible solution to this is to first detect the interacting attributes with an appropriate method, and then to properly model the known interactions. Non-parametric methods, such as classification trees and neural networks, are well suited for identifying interactions and dealing with them. One can use classification trees to identify the interactions, and afterwards to use another classification algorithm to build a predictive model [8]. Modeling interactions can be done either by segmenting the data into groups and using different classification model for each group; or by generating interaction attributes and using one model for the whole data set.

III. Modified calculation of Weight of evidence for binary problems

As we have emphasized previously in Section II, WoE has some properties that are well-suited for analyzing the features and each of their values, for features' transformation, or for estimating the features' information values. Unfortunately, there are some restrictions that need to be overcome in order WoE to be computable. The following two subsections describe these restrictions, and review the adjustments proposed in [7] that should lead to successful and accurate calculation of WoE for binary problems. Section IV presents our proposal of how the WoE method can be enhanced so it can be applied to problems with an arbitrary number of classes.

A. Unsatisfied preconditions

As it was mentioned previously when we defined WoE, the constraints $P_i^A > 0$ and $N_i^A > 0$ need to be satisfied in order to calculate WoE with (1). If these conditions are not met for some values of an variable, then the WoE of that variable would not be computable. But if we want to use the WoE method as a tool for transforming or binning the attributes, or to estimate attributes' information values, we need the WoE of all values of the variable. This implies that some adjustments have to be introduced so that the WoE can be calculated even when the preconditions are not met. The different types of unsatisfied preconditions are listed below.

Case 1: The number of positively labelled data points is zero $(P_i^A = 0)$ and the number of negatively labelled data points is zero $(N_i^A = 0)$. There are no data points with the i-th value of the variable A, so we assume that WoE_i^A is zero, meaning that this value will have no impact on any transformations nor will change the calculation of some other parameters that are dependent on WoE. In fact, this value could be even deleted from the possible set of values for the current attribute. But, because it does not have effect on anything, it could be retained in the set of possible values in case some data points from the data sets have the i-th value of variable A, as well as for future analysis.

Case 2: The number of positively labelled data points is zero $(P_i^A = 0)$ and the number of negatively labelled data points is greater than zero ($N_i^A > 0$). There are no positively labelled data points, and only negatively labelled data points with the i-th value of the variable A. We propose to add one data point that is labelled as positive, so $P_i^A = 1$, and to add the appropriate number of negatively labelled data points, so the overall ratio of the added data points will be equal to the ratio of the whole data set (SN/SP). Equations (5) and (6) define how the number of added data points will be calculated, as well as, the proposed estimate of WoE with (10).

$$\frac{\delta p_i^A}{\delta n_i^A} = \frac{SP}{SN} \tag{5}$$

Where $\delta p_i^A = 1$ and denotes the artificially added positive data points (in this case only one), where δn_i^A denotes the added negative data points. Applying $\delta p_i^A = 1$ in (5), we can calculate δn_i^A with (6):

$$\delta n_i^A = \frac{SN}{SP} \tag{6}$$

Now P_i^A and P_i^A that were defined in Section II can be modified to include the artificially added data points:

$$\Delta P_i^A = P_i^A + \delta p_i^A = 1 \tag{7}$$

$$\Delta N_i^A = N_i^A + \delta n_i^A = N_i^A + \frac{SN}{SP} \tag{8}$$

Now instead of calculating WoE with (1) using N_i^A and P_i^A, we can calculate it with their modified ΔN_i^A and ΔP_i^A, defined in the two previous equations:

$$WoE_i^A = \ln(\frac{\Delta N_i^A}{\Delta P_i^A}) - \ln(\frac{SN}{SP}) \tag{9}$$

And if we apply (7) and (8) in (9), finally we get the proposed estimate of WoE for this case of unsatisfied preconditions:

$$WoE_i^A = \ln(\frac{N_i^A \times SP + SN}{SN}) \tag{10}$$

Case 3: The number of negatively labelled data points is zero $(N_i^A = 0)$ and the number of positively labelled data points is greater than zero $(P_i^A > 0)$. There are no negatively labelled data points, and only positively labelled data points with the i-th value of the variable A. We propose to add one data point that is labelled as negative, so $N_i^A = 1$, and to add the appropriate number of positively labelled data points, so the overall ratio of the added data points will be equal to the ratio of the whole data set (SN/SP). In this case the number of artificially added negative data points is one $(\delta n_i^A = 1)$, so equation (5) can be transformed as:

$$\delta n_i^A = \frac{SP}{SN} \tag{11}$$

Now P_i^A and N_i^A that were defined in Section II can be modified to include the artificially added data points:

$$\Delta P_i^A = P_i^A + \delta p_i^A = P_i^A + \frac{SP}{SN} \tag{12}$$

$$\Delta N_i^A = N_i^A + \delta n_i^A = 1 \tag{13}$$

Finally for this case of unsatisfied preconditions, instead of calculating WoE with (1) using N_i^A and P_i^A, we can calculate it with their modified values ΔN_i^A and ΔP_i^A, defined in the two previous equations:

$$WoE_i^A = \ln(\frac{SP}{P_i^A \times SN + SP}) \tag{14}$$

This subsection proposed estimations of WoE for the specific cases when it cannot be calculated using the regular formula (1), because of unsatisfied preconditions. The estimation algorithm first virtually adds a particular number of data points so that the number of positively and negatively labelled data points for all variables and all their values is greater than zero. However, the very small number of data points is added with care in regards to their distribution, so the overall distribution of the data set is not changed at all. The benefits from the proposed estimation of WoE are:

- It would be computable for all variables and all values in the data set, meaning that WoE could be used to transform the nominal variables into numeric.
- The computed WoE could be used for binning of some values of the variables.
- Information value of all variables could be computed, and later it could be used in the variable (i.e. feature) selection phase.
- Many classification algorithms have preference of numeric attributes over nominal variables, and sometimes the distance between different data points cannot be estimated if the values of the variables are nominal. The transformed variables can be compared in terms of WoE (i.e. relative risk in respect to the class variable and the current dataset).

The estimations could be used for nominal and continual variables, because the unsatisfied preconditions in general pose a problem for all kinds of variables.
However, the proposed transformation could potentially lead to incorrect results in the presence of noise. Namely, if the noise is significant, then the estimated risk of a particular value of an variable could differ from the real risk. However, noisy data pose a serious problem for data mining, in general, and should be properly handled before applying any kind of data transformation.

B. Unknown values in the training set

Using equation (1) when all preconditions are met or otherwise the appropriate estimations of WoE that were proposed in Subsection A, it can be precisely calculated or estimated accurately enough for all values of all variables in the training data set. Regardless of the transformations of the variables, and regardless of the selected variables (features), a classification model can be constructed that will be dependent on the training data set. As a standard procedure, the classification model is validated and tested using some different data sets than the training data set, and it is not uncommon for these data sets to contain new and unknown values. This is even more likely to happen if the classification model is deployed in a production system, or if the validation and test data sets come from different time periods than the training data sets.

If WoE is used to transform nominal variables into numeric variables, then the new value will not pose any problem, because we can assume that its WoE is zero. When we can acquire labelled instances that contain the new values of nominal variables their WoE weight can be updated with (1). Doing anything more is only possible if we manually map the new value to some exciting value of the same variable. To illustrate this, let us consider the following example:

Example 2: We have a credit scoring problem, and one of the variables is the profession of the applicant. At the time when the prediction model was built 100 different values (professions) were possible for the variable ``Profession''. As time goes by, some of the applicants may have have a profession that is not one of the 100 possible values (e.g. Data Mining Consultant). We can assume that for the value ``Data Mining Consultant" of the variable ``Profession" the appropriate WoE is zero because we do not have any labelled instances with that value to claim otherwise. On the other hand, we can tweak the variable manually by saying ``All Data Mining Consultants are also Machine Learning Consultants". In this way instead of assuming that $WoE_{\Pr ofession}^{DataMiningConsul\tan t} = 0$ we can use in our advantage that the two professions are semantically connected and assume that $WoE_{\Pr ofession}^{DataMiningConsul\tan t} = WoE_{\Pr ofession}^{MachineLearningConsul\tan t}$. Be that as it may, such conversion and mapping of values requires a lot of manual work and is rarely used because there is no guarantee that it will improve the prediction performance.

On the other hand, if a numeric variable is transformed using WoE, than it is not so accurate, nor practical to assume that the WoE of some new value of a variable is zero. The new value may be very similar to some other value of the same variable that is already present in the data set, and in that case we may approximate that WoE of the new value is the same as WoE of the existing value. In order to use this approximation, we need to define a measure of similarity between values of a variable. With Algorithm 1 we propose a static, but easily computable method for finding similar values.

Let the training data set is denoted by *X* and represented in a matrix-like form where the number of rows is *m* and the number of columns is *n*. Each row of *X* represents an training instance (data point) and each column contains represents a feature (variable). Note that each column can contain arbitrary types of data (binary, numeric, nominal, etc.), so *X* in general case is not a matrix on which matrix operations can be applied directly. Using this notation we can say that x_i^j, where $0 \le i < m$ and $0 \le j < n$ represents the value of the *i*-th instance of the *j*-th feature, X^j is a row vector with size $1 \times n$ and X_i is a column vector with size $m \times 1$. Additionally the class values (labels) of each instance are represented with a column vector *Y* with size $m \times 1$.

Algorithm 1 Estimation of similar values for numeric features during the training phase

```
for $j = 1 \to n$ do
    $WoE^j = CalculateWoE(X^j, Y);$
    if j-th feature is numeric then
        $TV^j = GetUniqueValues(X^j);$
        $V^j = OrderAscending(TV^j);$
        $\Delta V^j = GetNeighbourDifferences(V^j);$
        $\mu^j = Average(\Delta V^j);$
        $\sigma^j = StDev(\Delta V^j);$
    end if
end if
```

First, from the *j* column that represents all values of the *j*-th feature the unique values are calculated and stored in TV^j, and then are ordered ascending and stored in V^j. Afterwards, we compute the differences between the neighboring values in V^j and store them in ΔV^j. Then we compute the average, denoted by μ^j, and the standard deviation, denoted by σ^j, of ΔV^j. Finally, if the the new value of the *j*-th feature is denoted by V_{new}^j, then the interval of similar values can be defined as:

$$V_{new}^j - \frac{\sigma^j + \sigma^j}{2} \le V_{similar}^j \le V_{new}^j + \frac{\sigma^j + \sigma^j}{2} \qquad (15)$$

Algorithm 2 shows how WoE is calculated during the training phase for all features and all values.

Algorithm 2 Calculation of WoE

```
function $CalculateWoE(V, Y)$
    $DV = GetUniqueValue(V);$
    $DC = GetUniqueValue(Y);$
    $C = 0$
    for $k = 1 \to Count(V)$ do
        $i = GetIndexOf(V_k, DV)$
        $j = GetIndexOf(Y_k, DC)$
        $C_i^j = C_i^j + 1$
    end for
    if $Count(V) = 2$ then
        $t = 1$
    else
        $t = Count(V)$
    end if
end function
```

Algorithm 3 shows how features are transformed during the test phase.

Algorithm 3 Transformation of features during the test phase

```
for $i = 1 \to p$ do
  for $j = 1 \to n$ do
    if $V^j$ $contains T_i^j$ then
      $k = GetIndex(V^j, T_i^j);$
      $Z_i^j = WoE_k^j;$
    else
      if j-th feature is numeric then
        $S = FindSimilarValues(T_i^j, j)$
        $W = GetWoE(S, j)$
        $Z_i^j = Avg(W)$
      else
        $Z_i^j = 0$
      end if
    end if
  end for
end for
```

REFERENCES

[1] R. Anderson, The Credit Scoring Toolkit - Theory and Practice for Retail Credit Risk Management and Decision Automation, Oxford University Press Inc., New York, USA, 2007.

[2] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I. H. Witten, The WEKA Data Mining Software: An Update, SIGKDD Explorations, Volume 11, Issue 1, 2009.

[3] D. B. Suits, Use of Dummy Variables in Regression Equations, Journal of the American Statistical Association, Volume 52, No. 280, pp. 548-551, December 1957

[4] H. C. Wu, R. W. P. Luk, K. K. F. Wong, K. L. Kwok, Interpreting TF-IDF term weights as making relevance decisions, ACM Transactions Journal on Information Systems, Volume 26, Issue 3, June 2008.

[5] N. Chater, M. Oaksford, Eds., The Probabilistic Mind: Prospects for Bayesian Cognitive Science, Oxford University Press, 2008.

[6] I. J. Good, Probability and the Weighing of Evidence, C.Griffn & Co., London, UK, 1950.

[7] E. Zdravevski, P. Lameski, A. Kulakov, Weight of Evidence as a Tool for Attribute Transformation in the Preprocessing Stage of Supervised Learning Algorithms, The 2011 International Joint Conference on Neural Networks (IJCNN), pp.181-188, San Jose, USA, July 2011

[8] L.C. Thomas, D.B. Edelman, J.N. Crook, Credit Scoring and its Applications, Society for Industrial and Applied Mathematics SIAM Publishing, Philadelphia, USA, 2002.