*South African Statistical Journal*
2018, Vol. 52, No. 2, 139–155


# AUTOBIN: A PREDICTIVE APPROACH TOWARDS AUTOMATIC BINNING USING DATA SPLITTING

***Tanja Verster***
Centre for Business Mathematics and Informatics,
North-West University, Potchefstroom, South Africa,
e-mail: *tanja.verster@nwu.ac.za*

The concept of binning is known by many names: discretisation, classing, grouping and quantisation. It entails the mapping of continuous or categorical data into discrete bins. Binning is an important pre-processing step in most predictive models and considered a basic data preparation step in building a credit scorecard. Credit scorecards are mathematical models which attempt to provide a quantitative estimate of the probability that a customer will display a defined behaviour (e.g. default) with respect to their current credit position with a lender. Among the practical advantages of binning are the removal of the effects of outliers and a way to handle missing values. Many binning methods exist but they are often time consuming to actually carry out. We propose a new method, Autobin, that is based on data splitting and maximising a cross-validation form of the predicted log-likelihood. Autobin has the advantage of being nearly automatic and requires very little by way of tuning parameters. In a limited simulation study done, it was found that Autobin outperforms its competitors.

*Key words:* Binning, Credit scoring, Data splitting, Predictive models.

## 1. Introduction

In predictive modelling, many statistical techniques are sensitive to the way data are analysed. Binning or discretisation is one of the simplest and most popular methods used in this regard. This paper is motivated by credit scoring applications and considers the case of regressing a zero-one response variable on a discrete explanatory variable which may need to be binned in order to improve its predictive quality. An example from a real life credit scoring data set is shown in Table 1. This is a home loans data set, for details of which see Wielenga, Lucas and Georges (1999). The response variable $Y$ indicates whether or not an applicant eventually defaulted (went "bad"). Here the regressor $X$ is "delinq" which is one of the independent variables in the data set and indicates the number of delinquent trade lines of the customer (as observed by the credit bureau 12 months earlier). It takes 14 different values ($v$) indicated in the second column of Table 1. The corresponding frequencies ($f$) are shown in the third column and the number of bads ($b$) among them are shown in the fourth column. The maximum likelihood estimate (MLE) of the probability of default (PD) at each value of $X$ is given by $p = b/f$ and is shown in the last column.

It is clear from Table 1 that most of the customers have zero delinquent trade lines, and the frequencies drop rapidly as the number of delinquent trade lines increases, with very few customers having more than six. Where the frequencies ($f$) are low, the MLE estimates of $p$ are likely to be

*MSC2010 subject classifications.* 91G40 62M20.

**Table 1**. Number of delinquent trade lines (delinq).

| $m$ | $v$ | $f$ | $b$ | $p = b/f$ |
|---|---|---|---|---|
| 1 | 0 | 4179 | 583 | 0.140 |
| 2 | 1 | 654 | 222 | 0.339 |
| 3 | 2 | 250 | 112 | 0.448 |
| 4 | 3 | 129 | 71 | 0.550 |
| 5 | 4 | 78 | 46 | 0.590 |
| 6 | 5 | 38 | 31 | 0.816 |
| 7 | 6 | 27 | 27 | 1.000 |
| 8 | 7 | 13 | 13 | 1.000 |
| 9 | 8 | 5 | 5 | 1.000 |
| 10 | 10 | 2 | 2 | 1.000 |
| 11 | 11 | 2 | 2 | 1.000 |
| 12 | 12 | 1 | 1 | 1.000 |
| 13 | 13 | 1 | 1 | 1.000 |
| 14 | 15 | 1 | 1 | 1.000 |

inaccurate and our predictions of the corresponding $Y$ may be poor. Binning the value of $X$ may offer a solution to this conundrum. By choosing appropriate bins to which the individual values belong, we may be able to estimate the PDs more accurately and thus improve the predictions of the corresponding $Y$.

*We first give a brief review of the literature on the notion of binning*. The concept of binning is known by many names such as discretisation, classing, categorisation, grouping and quantisation. For simplicity we use the term binning throughout this paper. Binning is the mapping of continuous or categorical data into discrete bins (Nguyen, Müller, Vreeken and Böhn, 2014). It is an important pre-processing step in most predictive models, and considered a basic data preparation step in building a credit scorecard (Thomas, 2009). Credit scorecards are mathematical models which attempt to provide a quantitative estimate of the probability that a customer will display a defined behaviour (e.g. default) with respect to their current credit position with a lender; see e.g. Thomas (2009), Siddiqi (2006) and Siddiqi (2017). Among the practical advantages of binning are removal of the effects of outliers and a way to handle missing values (Anderson, 2007).

Some references refer to the concept of binning in two stages, the first stage called initial enumeration or fine classing and the second stage called coarse classing (Anderson, 2007). However, not all references regard two stages as necessary, e.g. Liu, Hussain, Tan and Dash (2002) and Lee (2007). Another example is the credit scoring book (Siddiqi, 2006) that refers to one stage only. Baesens, Rosch and Scheule (2016) also refer to one stage only, namely coarse classing. In our paper we will think of the concept of binning as having one stage. According to Baesens et al. (2016) binning can be done for various reasons in the context of credit scorecards. For categorical variables, it is needed to reduce the number of categories. This could lead to fewer parameters and a more robust model may be obtained. For continuous variables, binning may also be beneficial to capture nonlinear effects in linear models.

Many binning methods exist of which we mention a few. One of the simplest methods to bin a continuous variable is to partition it into *equal-width intervals* (Chmielewski and Grzymala-Busse,

1996). Another method is to partition the variable such that the sample frequency in each interval is approximately the same; this is called the *equal-frequency-per-interval* method. Proc HPBIN (SAS Institute, 2014) from SAS incorporates "bucket" and "quantile" binning that are equivalent to these equal-width and equal-frequency methods.

A popular binning method is one with class entropy as a criterion to evaluate a list of "best" breakpoints which, together with the domain boundary points, induce the desired intervals (minimal-class-entropy method) (Fayyad and Irani, 1992). A similar method of binning is used in C4.5 (Quinlan, 1993). These methods are sometimes referred to as "*decision tree methods*" (Breiman, Fredman, Olsen and Stone, 1984). A variation of this decision type method of binning has been implemented in the SAS Enterprise Miner Interactive grouping node (Oliveira, Chari and Haller, 2008) and more practical instructions on this specific binning method are described in the SAS Course Notes (SAS Institute, 2015). A common name used for these decision tree methods is CART (Classification and Regression Trees). The implementation of CART can be found in most statistical software packages, for example RPART in R (Therneau and Atkinson, 2017).

Another method of binning is called the *cluster analysis* method. For an example, Chmielewski and Grzymala-Busse (1996) propose a method to bin variables by using hierarchical cluster analyses. There are also methods that optimise the binning based on the univariate distribution; e.g. one approach is based on the information-theoretic minimum description length principle (Kontkanen and Myllymäki, 2007). A traditional method used in scorecards is weights of evidence (Lund and Raimi, 2012), although this method is more often used to evaluate a specific binning and not to propose a binning method.

Other methods, using rough sets, are found in the literature, e.g. Beynon and Peel (2001), Roy and Pal (2003) and Beynon (2004). More methods for continuous attributes can be found, e.g. Cantú-Paz (2001), Liu et al. (2002), Lee (2007), Chen, Tang, Liu and Li (2011) and Okumura (2011). Many other excellent studies on binning can be found in the literature. We thank the referee for bringing to our attention the algorithm and associated SAS code of Lund (2017) which is guaranteed to find the best k-bin solution for each k with respect to either information value or log-likelihood (where k indicates the number of bins) as well as the best monotonic solutions if they exist.

*To summarise the literature on binning*, many methods exist and often binning is a time-consuming process (Anderson, 2007) and impractical especially in big data sets containing many variables (Siddiqi, 2006).

We propose a new method that automatically bins explanatory variables, based on data splitting. Before describing this method, we also give *a brief review of the literature on the notion of data splitting*.

The concept of data splitting is often motivated by a quote from the popular introductory data analysis textbook "*Data Analysis and Regression*" (Mosteller and Tukey, 1977). The quote states the following: "*Testing the procedure on the data that gave its birth is almost certain to overestimate performance, for the optimising process that chose it from among many possible procedures will have made the greatest use of any and all idiosyncrasies of those particular data. . .* " In predictive modelling, the typical strategy for honest assessment of model performance is data splitting. Data splitting is the method of dividing a sample into two parts and then developing a hypothesis or estimation method on the basis of one part and testing it on the other part (Barnard, 1974). Picard and Berk (1990) review data splitting in the context of regression and provides specific guidelines for

validation in regression models, i.e. 25% to 50% of the data is recommended for validation. Faraway (2016) illustrates that a split data analysis is preferred to a full data analysis for predictions, with some exceptions.

For an early history of data splitting, see Stone (1974). At present, data splitting is used by an internet business, Kaggle (http://www.kaggle.com), which challenges analysts to develop the best predictive method using a training set supplied to them. Part of the data held back by Kaggle is used to judge the quality of the submitted methods and to select the "best" predictive method.

*To summarise the literature on data splitting*, it is often used in predictive models to enhance performance. The literature also gives guidelines on the sizes of these splits and lists many benefits of data splitting.

Since our eventual aim is prediction, we formulate the binning problem in terms of data splitting and cross prediction and compare this newly proposed method, referred to as Autobin, with two other existing methods. The largest advantages of Autobin are that the process and the choice of its tuning parameters are nearly automatic. The simulation study also shows that our technique outperforms some of the existing binning techniques.

The layout of the rest of the paper is as follows: In Section 2 we formulate the problem from the point of view of maximising a suitable mutual cross-validation likelihood based measure. In Section 3 we discuss an optimisation algorithm based on dynamic programming that underlies Autobin. In Section 4 we return to the illustration used above. Section 5 reports the results of a simulation study which compares Autobin to some of the existing binning techniques. The conclusions and some ideas for future research are given in Section 6.

## 2. Optimal binning via sample splitting and predictive likelihood cross-validation

The following mathematical notation will be used throughout the paper. Let $Y$ be a zero-one response variable and $X$ a discrete regressor that can take values in the set $V = \{v_1, v_2, \ldots, v_M\}$. Henceforth we refer to the $v_m$ as the "$X$-values" for clarity reasons since the term "value" occurs in many other contexts as well. Suppose that we have a sample of $N$ observations on $Y$ and $X$ and these data are given in the form of a frequency table with the value $v_m$ occurring $f_m$ times while the corresponding numbers of observed 1s and 0s of $Y$ are $b_m$ and $g_m$ respectively, for $m = 1, 2, \ldots, M$. In credit scoring the 1s are regularly referred to as the "bads" and the 0s as the "goods". Our ultimate aim is to predict the value of $Y$ given a newly observed value of $X$ and for this purpose we will define $\pi_m = P(Y = 1|X = v_m)$. If $\pi_m$ is large (small) we may predict that $Y = 1$ (0). For this purpose we need to estimate $\pi_m$. The MLE of $\pi_m$ is given by $p_m = b_m/f_m$ .

As our example in Table 1 above shows, data sets often have many of the $v_m$ occurring infrequently, so that estimating the probability of $Y$ taking the value 1 at such $v_m$ will not be accurate. This raises the question of whether we would do better by binning the original $X$-value set into a smaller set of values that occur with higher frequencies and thus ultimately enable us to get better predictions.

Our solution proceeds via data splitting and we now bring this into our notation. To start, we draw two separate random samples from the data set, the first one having $N'$ observations and the second one having $N''$ observations. Arrange both samples in frequency tables.

Suppose the value $v_m$ of $X$ occurs $f'_m$ ($f''_m$) times in the first (second) sample while the corresponding numbers of observed 1s (bads) of $Y$ are $b'_m$ ($b''_m$) and the numbers of 0s (goods) are $g'_m$ ($g''_m$) in the

two samples, respectively, for $m = 1, 2, \ldots, M$. Let $p'_m$ and $p''_m$ denote estimates of $\pi_m$ based on the first and second samples, respectively. To be explicit, we take the $p'_m = b'_m/f'_m$ and $p''_m = b''_m/f''_m$, assuming that both $f'_m$ and $f''_m$ are positive, but other estimators are also possible.

In general, if $Y$ is a random variable with support $\{0, 1\}$ and $y$ is an observed value of $Y$, while $p = P(Y = 1)$, then the log-likelihood is given by

$$l(y, p) = [y \log(p) + (1 - y) \log(1 - p)] \text{ for } y = 0, 1 \text{ and } 0 \leq p \leq 1. \tag{1}$$

In the context above, suppose that the value of $Y$ for the $n$-th observation in the second sample is $y''_n$ and the value of $X$ is $v_m$. The first sample gives us the estimate $p'_m$ of the probability of getting $Y = 1$ when $X = v_m$. Substituting $y''_n$ and $p'_m$ into (1) we get a predicted log-likelihood (PL) of $l(y''_n, p'_m)$ when comparing the first sample's probability estimate with the second sample's observed $Y$. By summing $l(y''_n, p'_m)$ over all observations $n$ in the second sample and rearranging the sum according to the different possible values of $X$, the total PL when comparing the probability predictions based on the first sample with the observed $Y$s in the second sample is

$$PL'' = \sum_{m=1}^{M} \left\{ b''_m \log\left(p'_m\right) + g''_m \log\left(1 - p'_m\right) \right\}. \tag{2}$$

Now we interchange the roles of the two samples by comparing the predictions based on the second sample with observations in the first sample. This total PL is

$$PL' = \sum_{m=1}^{M} \left\{ b'_m \log\left(p''_m\right) + g'_m \log\left(1 - p''_m\right) \right\}. \tag{3}$$

By adding the two PLs and dividing by the total number of observations in the two samples, we get the **average mutual cross-validation PL** (CV for short)

$$CV = \frac{1}{N' + N''} \sum_{m=1}^{M} \left\{ b''_m \log\left(p'_m\right) + g''_m \log\left(1 - p'_m\right) + b'_m \log\left(p''_m\right) + g'_m \log\left(1 - p''_m\right) \right\}. \tag{4}$$

The issue of **optimal binning** from a predictive point of view can now be formulated as follows. Let $\boldsymbol{C} = \{C_1, C_2, ..., C_K\}$ denote a partition of the $X$-value set $V$ into $K$ mutually exclusive and exhaustive subsets (bins) and denote the corresponding observed data of the first sample in the bin $C_k$ by $f'_k(\boldsymbol{C}) = \sum_{m=1}^{M} f'_m I(v_m \in C_k)$, $b'_k(\boldsymbol{C}) = \sum_{m=1}^{M} b'_m I(v_m \in C_k)$ and $g'_k(\boldsymbol{C}) = \sum_{m=1}^{M} g'_m I(v_m \in C_k)$ and similarly for the second sample when partitioned. Also introduce similar notation for the estimated probabilities $p'_k(\boldsymbol{C})$ and $p''_k(\boldsymbol{C})$ based on the first and second samples for bin $C_k$. Then we ask for that choice of partition $\boldsymbol{C}$ that maximises

$$\begin{aligned} CV(\boldsymbol{C}) = \frac{1}{N' + N''} \sum_{k=1}^{K} \Big\{ & b''_k(\boldsymbol{C}) \log\left(p'_k(\boldsymbol{C})\right) + g''_k(\boldsymbol{C}) \log\left(1 - p'_k(\boldsymbol{C})\right) \\ & + b'_k(\boldsymbol{C}) \log\left(p''_k(\boldsymbol{C})\right) + g'_k(\boldsymbol{C}) \log\left(1 - p''_k(\boldsymbol{C})\right) \Big\} \end{aligned} \tag{5}$$

among all possible partitions. In general the solution of this problem is very difficult. However, if the $X$-value set $V = \{v_1, v_2, \ldots, v_m\}$ is ordered, an efficient dynamic programming solution is possible, as discussed in the next section.

## 3. Optimal binning for ordered regressors: Autobin

Suppose that the $X$-value set $V = \{v_1, v_2, \ldots, v_M\}$ is ordered, i.e. we are dealing with an ordinal regressor. The $v_m$ need not be numbers, but we denote their ordering by $v_1 < v_2 < \cdots < v_M$. Further we suppose that the bins in the partition must consist of contiguous intervals of the $v_m$, i.e. we must have

$$C_k = \left\{v_{t_{k-1}+1}, v_{t_{k-1}+2}, \ldots, v_{t_k}\right\} \text{ for } k = 1, 2, \ldots K. \tag{6}$$

Here $1 \leq t_1 < t_2 < \cdots < t_K = M$ denote the **endpoints** of the $K$ bins within the set $\{v_1, v_2, \ldots, v_M\}$ and we take $t_0 = 0$ for notational convenience. Let $F'_m = \sum_{i=1}^m f'_i$, $B'_m = \sum_{i=1}^m b'_i$ and $G'_m = \sum_{i=1}^m g'_i$ denote the cumulative frequencies of the first sample (and define $F'_0 = B'_0 = G'_0 = 0$). Then $f'_k(\boldsymbol{C}) = F'_{t_k} - F'_{t_{k-1}}$, $b'_k(\boldsymbol{C}) = B'_{t_k} - B'_{t_{k-1}}$ and $g'_k(\boldsymbol{C}) = G'_{t_k} - G'_{t_{k-1}}$. Also introduce similar notation for the frequencies of the second sample. Consider the interval $[v_{r+1}, \ldots, v_s]$ for $1 \leq r < s \leq M$ and write

$$\begin{aligned}\Delta(r,s) = \left(B''_s - B''_r\right) \log\left(p'_{rs}\right) + \left(G''_s - G''_r\right) \log\left(1 - p'_{rs}\right) \\ + \left(B'_s - B'_r\right) \log\left(p''_{rs}\right) + \left(G'_s - G'_r\right) \log\left(1 - p''_{rs}\right). \end{aligned} \tag{7}$$

Here $p'_{rs} = (B'_s - B'_r)/(F'_s - F'_r)$ is the estimate of $P(Y = 1 | v_r < X \leq v_s)$ from the first sample and similarly for $p''_{rs}$. The sum in (5) can then be written as $\sum_{k=1}^K \Delta(t_{k-1}, t_k)$ and the optimal partition problem is to calculate

$$CV^*(M) = \max \frac{1}{N + N'} \left\{ \sum_{k=1}^K \Delta(t_{k-1}, t_k) \, : 1 \leq t_1 < t_2 < \cdots < t_K = M \, , \, 1 \leq K \leq M \right\}. \tag{8}$$

A dynamic programming algorithm to solve this problem can be set up as follows. We look at the initial section of the first $J$ values of $V$ and wish to partition it optimally into $L$ bins with $1 \leq J \leq M$ and $1 \leq L \leq J$ . The optimal total prediction loss for this problem is

$$PL^*(L, J) = \max \left\{ \sum_{l=1}^L \Delta(t_{l-1}, t_l) : 1 \leq t_1 < t_2 < \cdots < t_L = J \right\}. \tag{9}$$

Considering the different possible placements of $t_{L-1}$ we can write

$$PL^*(L, J) = \max \left\{ PL^*(L-1, i) : \, L - 1 \leq i \leq J \right\} \tag{10}$$

for $2 \leq L \leq J$. Also if $t_1^*(L, J), \ldots, t_L^*(L, J)$ denote the choices of $t_1, \ldots, t_L$ at which the maximum $PL^*(L, J)$ is achieved and $i^*$ is the choice of $i$ which maximises (10), then

$$t_1^*(L, J) = t_1^*(L-1, i^*), \ldots, t_{L-2}^*(L, J) = t_{L-2}^*(L-1, i^*), t_{L-1}^*(L, J) = i^*, t_L^*(L, J) = J. \tag{11}$$

Since $PL^*(1, J)$ is easily calculated, (10) and (11) constitute the dynamic programming iterations that can be used to calculate $PL^*(K, M)$ and then solve (9) exactly as

$$CV^*(M) = \max \frac{1}{N + N'} \left\{ PL^*(K, M) \, , \, 1 \leq K \leq M \right\}. \tag{12}$$

Note that $p'_{rs} = (B'_s - B'_r)/(F'_s - F'_r)$ is only defined if $F'_s - F'_r > 0$ and similarly for $p''_{rs}$ . Moreover, if $B'_s - B'_r = 0$ or $G'_s - G'_r = 0$ then $p'_{rs} = 0$ or $p'_{rs} = 1$ in which cases $\log(p'_{rs})$ or $\log(1 - p'_{rs})$

**Table 2**. DELINQ binned.

| $v$ | $f$ | $b$ | **MLE PD estimates** | **Autobin PD estimates** |
|---|---|---|---|---|
| 0 | 4179 | 583 | 0.140 | 0.13951 |
| 1 | 654 | 222 | 0.339 | 0.34144 |
| 2 | 250 | 112 | 0.448 | 0.45068 |
| 3 | 129 | 71 | 0.550 | 0.54793 |
| 4 | 78 | 46 | 0.590 | 0.57139 |
| 5 | 38 | 31 | 0.816 | 0.91974 |
| 6 | 27 | 27 | 1.000 | 0.91974 |
| 7 | 13 | 13 | 1.000 | 0.91974 |
| 8 | 5 | 5 | 1.000 | 0.91974 |
| 10 | 2 | 2 | 1.000 | 0.91974 |
| 11 | 2 | 2 | 1.000 | 0.91974 |
| 12 | 1 | 1 | 1.000 | 0.91974 |
| 13 | 1 | 1 | 1.000 | 0.91974 |
| 15 | 1 | 1 | 1.000 | 0.91974 |

is $-\infty$. Again, similar remarks hold for the second sample. In order to avoid these difficulties when ratio estimators are used for the probabilities in question, we have to do the optimisation in (8) under the additional restrictions that each bin contains at least one 1 and one 0. More generally, we may want to do the optimisation under prescribed lower bounds larger than zero for the bin frequencies and numbers of 1s and 0s. This is easily handled by restricting the choice of $i$ which maximises (10), such that

$$\begin{array}{ll} F'_{t_k} - F'_{t_{k-1}} \geq f_{min}\,, & F''_{t_k} - F''_{t_{k-1}} \geq f_{min}\,, \\ B'_{t_k} - B'_{t_{k-1}} \geq b_{min}\,, & B''_{t_k} - B''_{t_{k-1}} \geq b_{min}\,, \\ G'_{t_k} - G'_{t_{k-1}} \geq g_{min}\,, & G''_{t_k} - G''_{t_{k-1}} \geq g_{min}\,. \end{array} \tag{13}$$

Up to this point we used the sample splitting process and CV criterion to choose corresponding optimal bins. Once this is done, we return to the original full sample to find PD estimates. The PD estimate for each $v$-value in a specific bin is the same and this PD estimate is the ratio of the total number of the $b$ (number of bads) in that bin to the total of the $f$ (frequency) of observations in that bin. The description so far was in terms of a single split of the data, but in practice we can do a large number of splits and average the corresponding PD estimates over the splits.

We refer to this method as **Autobin**. It involves the tuning parameters $f_{min}$ , $b_{min}$ and $g_{min}$ as well as the number of random splits to use. Requiring at least two observations per bin (i.e. $f_{min} = 2$) with at least one bad (i.e. $b_{min} = 1$) and at least one good (i.e. $g_{min} = 1$) enforce very little restrictions on Autobin and may be thought of as reasonable default choices. We have observed them to work very well in practice and all the results below are based on this choice. The number of splits to use is really a matter of the more the better, but computational time may dictate what is practically possible. In all the results below we used 1000 splits. The Autobin methodology was implemented in PROC IML in SAS, and the code is available from the author.

## 4. Illustration

We use the data described in Section 1 to illustrate Autobin with default parameter choices as indicated above. Table 2 lists and Figure 1 illustrates the results graphically.

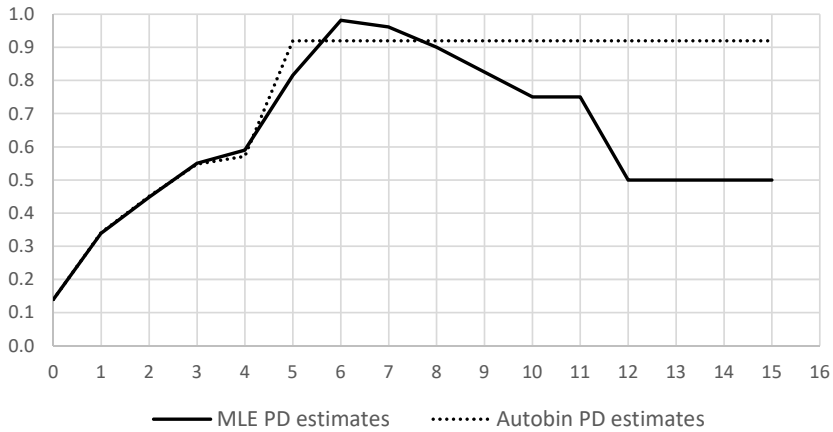


**Figure 1**. MLE and Autobin PD estimates for home loan data.

It is evident that Autobin keeps the first five $X$-values separate and amalgamates the remainder into a single bin. Over the first five $X$-values the MLE PD estimates and the Autobin estimates agree, but they differ substantially over the amalgamated bin.

Will Autobin deliver better PD estimates than its competitors? Below, we discuss the results of simulation studies designed to answer this question.

## 5. Binning methods compared by a simulation study

In this section, we report results on simulation comparisons to investigate how Autobin performs compared to two popular binning methods, namely RPART from R and HPBIN from SAS.

The motivation for our choice of using RPART is as follows. RPART (Therneau and Atkinson, 2018) is the R implementation of the CART methodology (Breiman et al., 1984). Some credit scoring textbooks refer to the successful use of decision trees to bin variables, e.g. Siddiqi (2006) and the study of Greif (2013), using RPART for binning variables in a credit scoring environment. Section9.4 of the SAS Course Notes (SAS Institute, 2015) illustrates the use of a decision tree node to bin variables. Thus decision trees appear popular as a binning technique and therefore we chose the RPART implementation.

Regarding HPBIN, the SAS online documentation motivates that "Binning is a common step in the data preparation stage of the model-building process. You can use binning to classify missing variables, reduce the impact of outliers, and generate multiple effects. The generated effects are useful and contain certain nonlinear information about the original interval variables. The HPBIN procedure conducts high-performance binning" (SAS Institute, 2014). Also, Baesens et al. (2016) state that the PROC HPBIN in SAS is useful for binning. Thus we chose to include PROC HPBIN also.

The RPART programs (Therneau and Atkinson, 2018) build classification or regression models of a very general structure using a two stage procedure; the resulting models can be represented as binary trees. There are two tuning parameters in the RPART, namely **minbucket**, which refers to the minimum number of observations in a terminal node, and **cp**, which refers to the threshold

complexity parameter. Choosing the tuning parameters for RPART and HPBIN was difficult. Many combinations are available and we followed the literature where RPART was used in a credit scoring environment for binning (Greif, 2013). The default value for **cp** is 0.01, but this resulted in too few bins in our results below and we used 0.001 instead. For **minbucket** we used choices between 2 and 500, motivated by the rule of thumb given by Siddiqi (2006).

The HPBIN procedure in SAS conducts binning by using four different methods (SAS Institute, 2014), namely bucket, quantile, pseudo-quantile and Winsorized binning. **Bucket** binning creates equal-length bins. The default number of bins (the binning level) is 16 but we can make other choices. **Quantile** binning aims to assign the same number of observations to each bin, if the number of observations is evenly divisible by the number of bins. As a result, each bin should have the same number of observations, provided that there are no tied values at the boundaries of the bins. Because PROC HPBIN always assigns observations that have the same value to the same bin, quantile binning might create unbalanced bins if any variable has tied values. **Pseudo–quantile** binning is an approximation of quantile binning. The pseudo–quantile binning method is a much more efficient way to bin (less processing time required) and mimics the results of the quantile binning method. **Winsorized** binning is similar to bucket binning except that both tails are cut off to obtain a smooth binning result. This technique is often used to remove outliers during the data preparation stage. If Winsorized binning is chosen, the user must specify the WINSORRATE option with a value from 0.0 to 0.5 exclusive. We used 5% (i.e. WINSORRATE=0.05) as in the examples illustrated in the online help documentation (SAS Institute, 2014).

The binning methods were compared using four special model cases. To describe these cases we use the following notation in addition to that of Section 2. Let $\phi_m$ denote the probability of $X$ taking the value $v_m$, i.e. $\phi_m = P(X = v_m)$. Then, specifying the $X$-value set $V = \{v_1, v_2, ..., v_M\}$ together with the parameters $\{(\phi_m, \pi_m), m = 1, ..., M\}$ constitute choosing an underlying model and we can take the parameters such that various cases of interest are obtained. Once a model is chosen, we can generate a data set of $N$ independent and identically distributed (i.i.d.) observations of the pair $(X, Y)$ and then apply the binning methods to the generated data to get estimates of the corresponding PDs (i.e. the $\pi_m$). We do repeated simulation runs, each producing PD estimates and calculate the averages and mean squared errors (MSEs) of these PD estimates over repeated runs to compare the different binning methods.

### Case 1

Recall that our original motivation for binning was the need to improve PD estimation at low frequency $X$-values. Our first simulation case takes this situation to an extreme using a large $X$-values set all with low frequencies but varying PD values. Specifically, we take $V = \{1, 2, ..., 100\}$, $\phi_m = 0.01$ and $\pi_m = (1 + \sin(m/8)/2)$ for $m = 1, 2, \ldots, 100$ . The sample size was set at $N = 200$ and the default tuning parameters for Autobin were used. 1000 repeated simulation runs were done.

Four variations of RPART and five variations of HPBIN were used to compare with Autobin. Choosing the tuning parameters for RPART and HPBIN were not trivial matters, being somewhat like a "hit-and-run" exercise. The combinations of available choices are huge and due to time and space constraints, we only report on a few combinations. We experimented with various tuning parameters in RPART and fixed **cp** at 0.001 but varied **minbucket** over the values 2, 5, 10 and 15. We used all four options available in HPBIN, namely **bucket**, **quantile**, **pseudo-quantile** and

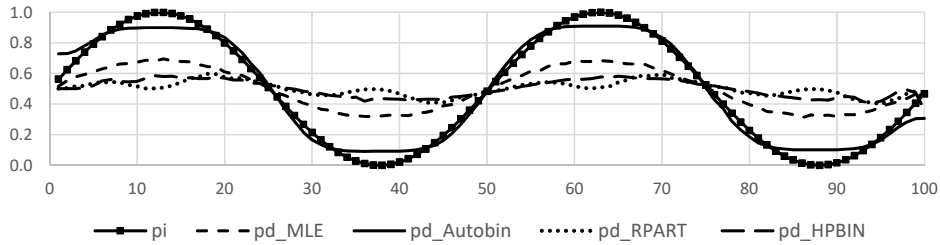


**Figure 2**. True and average estimated PDs for Case 1 .

**Winsorized** binning. The default number of bins for these four settings is 16 bins. We also used a variation on the default setting in the bucket option where we specified 4 bins. For the Winsorized binning we used a 5% trimming rate.

Figure 2 shows the first results obtained. The horizontal axis is the $X$-value labels $m$ , the solid line with the square marker is the true $\pi_m$ as function of $m$ and the short-dashed line shows the averages over the 1000 repeated simulation runs of the MLEs of the $\pi_m$. The solid line shows the averages of the Autobin estimates. For visibility reasons, average estimates of only one of the five variations of RPART are shown by the dotted line, namely where **cp** had the value of 0.001 and **minbucket** had the value 10. Similarly, the average estimates of only one of the four variations of HPBIN are shown by the long-dashed line, namely where the quantile binning option was used with the default setting of 16 bins. Note that the differences between the average curves and the true PD-curve give the estimated biases of the methods.

The results can be explained as follows. With just 200 observations in total, there are on average just a few observations per value $m$. Where $\pi_m$ is high, there will be more "bads" than when $\pi_m$ is low. So the MLE PDs tend to produce higher (or lower) curves where $\pi_m$ is higher (or lower), but we cannot expect them to be accurate with such low frequencies. We see that the MLE curve does follow the true $\pi_m$ curve to some extent, but not closely. The Autobin curve does significantly better, since it bins the $X$-values locally and then obtains larger frequencies and larger (or lower) numbers of "bads" where the true $\pi_m$ curve is higher (or lower). This helps it to gain accuracy of estimation, in line with our initial intuitive motivation for the use of binning. Of course, the binning process needs to be sophisticated enough not to overdo this since otherwise it may produce a step type curve. In our view Autobin meets this challenge effectively. On average Autobin used approximately 8 bins but their locations varied over repeated simulation runs, enabling it to produce fairly smooth average estimates.

Comparing Autobin with RPART and HPBIN, it is clear that both RPART and HPBIN did not really notice the variation in the true PDs, their average estimates hovering mostly near to 0.5 and differing slightly at the high and low PD regions.

Figure 3 shows the mean square error (MSE) of the estimates over the 1000 simulation repetitions for Case 1. The solid line with the square marker is again the true $\pi_m$-s and all MSE values were scaled up by a factor of 10 to make their curves more visible. The short-dashed line is the MSEs for the MLEs and the solid line shows the MSEs for Autobin which tend to be lower than that of the MLEs, especially at the higher values of PD.

The MSEs for RPART and HPBIN are shown by the dotted line and by the long-dashed line

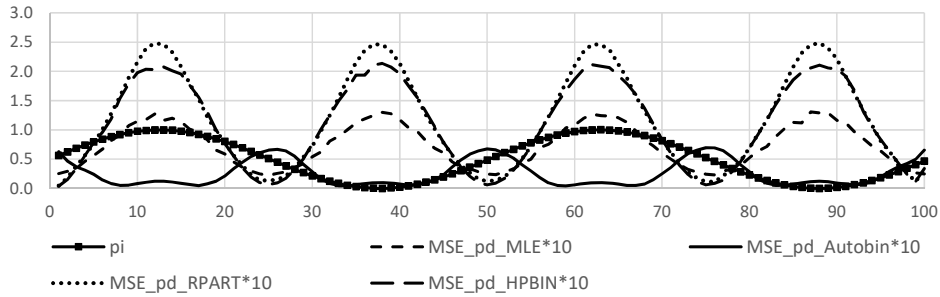


**Figure 3**. True PDs and MSEs of estimated PDs for Case 1.

respectively. In both the RPART and HPBIN cases, the MSEs are much worse when the true PDs are high, but it is interesting to note that when the true PDs are close to 0.5, we observe better MSEs than for the MLE and the Autobin procedure. HPBIN seems to have slightly lower MSE values than RPART. Overall, our judgement is that Autobin performed more reliably than MLE, RPART and HPBIN.

*A note on the results of different tuning parameters chosen for RPART and HPBIN*: among the selected combinations of tuning parameters we chose, we could not find a set that causes the estimated PDs to follow the true PD curve closely on average. We did notice that certain specific choices of tuning parameters improve the MSE for a few selected $X$-values (sometimes up to a 200% improvement for the MSE), but these were very rare, and each time for different $X$-values and a different set of tuning parameters. For most of the individual $X$-values the MSE did not improve over the different sets of tuning parameters. It might be possible that for each of the 100 $X$-values, there exists a specific combination of tuning parameters that could potentially improve the MSE, but this would result in a very time-consuming search. In our selected nine combinations, we rarely found a combination that had a lower MSE than that of Autobin. In most cases the MSEs were much higher for all nine combinations compared to that of Autobin. The detailed results of these variations are available from the author.

### Case 2

If there are higher frequencies for each $X$-value, the MLE should improve and there should be less need to use a binning method. The second simulation case is designed to consider this situation. We increased the sample size to $N = 1000$ but kept all the other parameter choices as in Case 1. Figure 4 shows the results for the average PD estimates.

Both the MLE and Autobin curves follow the true $\pi_m$ curve more closely and there is also a smaller difference between the MLE and the Autobin curves. On average, the binning estimates were based on some 21 bins. With higher frequencies for each $X$-value, Autobin can reach the same accuracy with fewer $X$-values and therefore use more bins. This is to be expected, but the benefit is that this is done automatically.

Figure 4 also shows one variation of RPART and one variation of HPBIN for Case 2. Again, Autobin outperformed both RPART and HPBIN by more closely following the true PD curve. Many variations of tuning parameter choices for RPART and HPBIN were investigated, but none of them

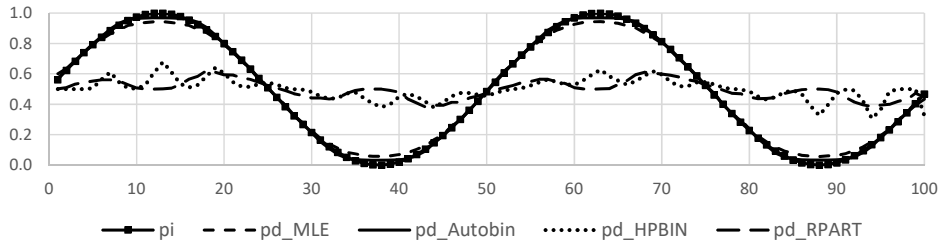


**Figure 4**. True PDs and average estimated PDs for Case 2.

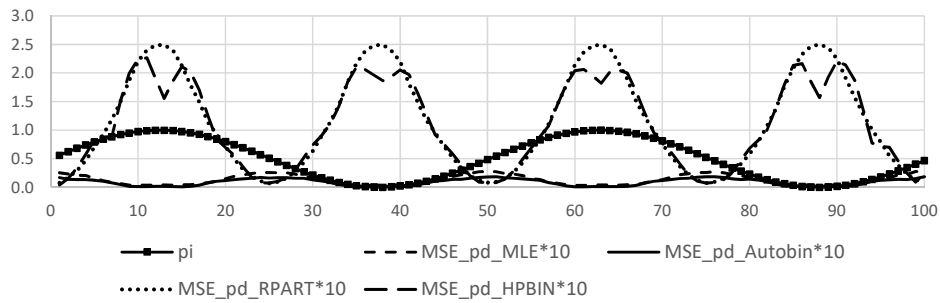


**Figure 5**. True PDs and MSEs of the estimated PDs for Case 2.

resulted in average estimation PD curves that closely followed the true PD curve.

Figure 5 shows that Autobin and MLE had similar results with regards to MSE in Case 2, both doing substantially better than their RPART and HPBIN competitors. It seems that both RPART and HPBIN tend to produce PD estimates close to 0.5 and in that sense are way off when the true PDs are close to 1 and 0. This is also clear from the extremely high MSEs for PDs close to 1 and 0 as shown in Figure 5.

Again, similar results for other tuning parameters choices for RPART and HPBIN are available from the author.

### Case 3

In Cases 1 and 2, each $X$-value had the same probability. If some $X$-values have smaller and others larger probabilities of occurring, the MLE should do worse at the lower probability $X$-values, but Autobin should be less affected. The third simulation case is designed to look into this matter. We changed the $\phi_m$ as follows: $\phi_m = 0.015$ for $m = 1, \ldots, 50$ and $\phi_m = 0.005$ for $m = 51, \ldots, 100$ but took the other parameters as for Case 2 . Figure 6 and Figure 7 show the results.

Over the first 50 higher frequency $X$-values both average MLE estimated PD curves and those of Autobin are close to the true PD curve, but over the 50 lower frequency $X$-values at the end, the MLE curve is not so close to the true PD curve while the Autobin curve still does well. This confirms our anticipation, showing that Autobin automatically adapts to the situation at hand, better so than the MLE estimates.

Comparing Autobin with the other two binning techniques, it is again clear that Autobin outperforms both RPART and HPBIN. The different tuning parameters used in the five variations for RPART

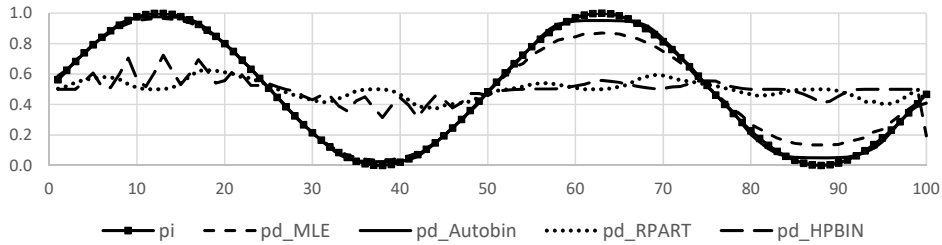


**Figure 6**. True PDs and average estimated PDs for Case 3.

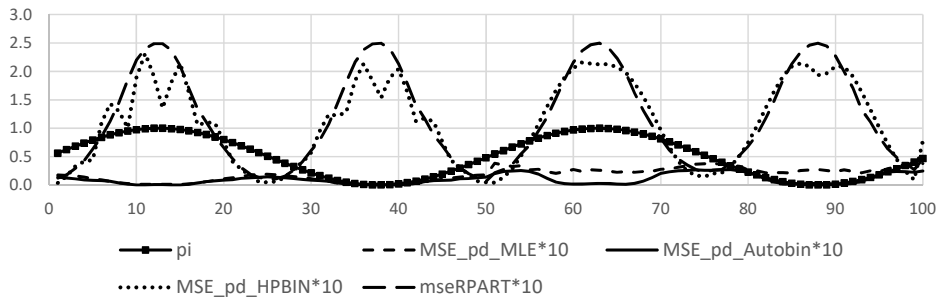


**Figure 7**. True PDs and MSEs of the estimated PDs for Case 3.

and the four variations in HPBIN did not generate a PD estimation curve that closely followed the true PD curve. Note that some of the combinations resulted in more "spikey" curves that were not flat around 0.5 but also not close to the true PD curve even where higher frequencies were available.

Moving to the MSE results in Figure 7, we observe that the Autobin and MLE had similar results for the first 50 $X$-values, but Autobin did better for the last 50 $X$-values, again as anticipated. Both RPART and HPBIN performed relatively poorly again.

In summary, this case again shows the advantage of Autobin: if some of the $X$-values have low frequencies, the binning process automatically compensates by joining enough $X$-values to get the joint frequency large enough to make the PD estimates more reliable and this comes at no cost in terms of poorer behaviour at high frequency $X$-values.

### Case 4

To further confirm the summary above, we look at a much larger data set, namely with sample $N = 10\,000$ but coupled with greater variability in frequencies of the $X$-values. To this effect we divide $\phi_m$ into even more sections, namely $\phi_m = 0.025$ for $m = 1,\ldots,25$ and $\phi_m = 0.01$ for $m = 26,\ldots,50$ , $\phi_m = 0.004$ for $m = 51,\ldots,75$ and $\phi_m = 0.001$ for $m = 76,\ldots,100$. So the later $X$-values have progressively smaller probabilities and thus relatively lower frequencies in the data. The PDs $\pi_m$ were the same as in the previous case. Figure 8 and Figure 9 show the results.

From Figure 8 it is clear that on average the MLE and Autobin estimates are very close to the true PD values, with Autobin being somewhat better towards the end where the lower frequencies occur. It is also clear that Autobin still outperforms RPART and HPBIN which tend to behave quite erratically at different PD frequency levels. We also investigated if we could improve the results of

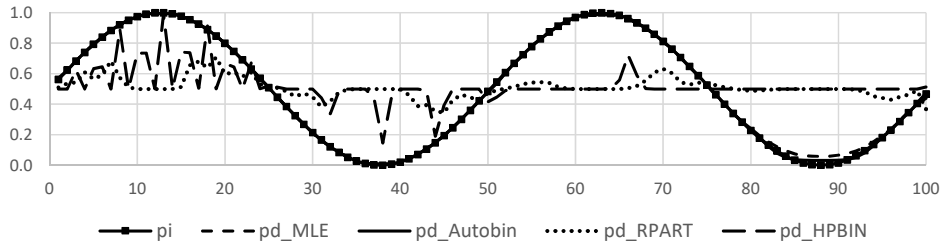


**Figure 8**. True PDs and average estimated PDs for Case 4.

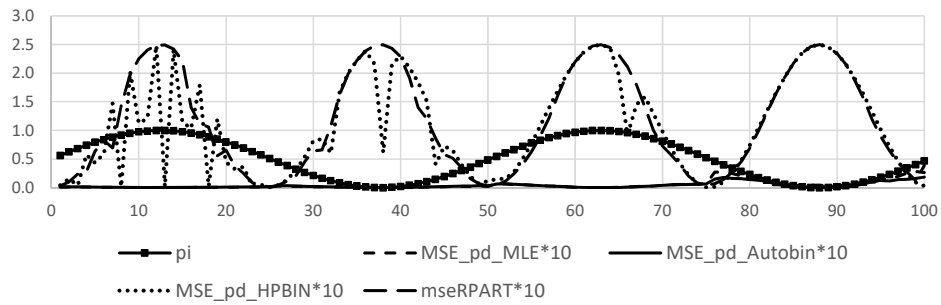


**Figure 9**. True PDs and MSEs of the estimated PDs for Case 4.

RPART and HPBIN by specifying a larger number of minimum observations in the buckets and other choices of tuning parameters but found none that resulted in notable improvements. Details of these results are also available from the author.

This simulation study further confirms that in large data sets and with high frequencies at all $X$-values, we may as well work with the MLEs, but if there happen to be $X$-values with low frequencies, then Autobin is preferable and has the added advantage that its performance will match that of MLE closely at high frequency $X$-values.

## 6. Conclusion and future research

In this article, a brief literature study on the concepts of binning and data splitting was provided, focussing on the credit scoring environment. Often in credit scoring, binning is required to more accurately predict the default probabilities. The problem of binning was formulated from the point of view of maximising a suitable mutual cross-validation based prediction log-likelihood measure. We proposed an optimisation algorithm based on dynamic programming that can be used to find optimal bins for the case of ordered regressor values. This culminated in our main research contribution, namely a new binning technique, called Autobin.

We illustrated Autobin in terms of a real data example and provided the results of simulation studies to compare Autobin with two popular binning techniques, namely RPART and HPBIN. Our conclusion is that Autobin outperformed its competitors while also having the added benefit that it is virtually automatic, requiring only specification of minimal frequencies per bin which can be set at the innocuous default levels of at least two observations and at least one bad and at least one good

observation per bin. Compared to other binning procedures this is a major advantage. The choices of the number of bins to use in RPART and HPBIN are not a trivial at all. Also, the **cp** tuning parameter of RPART and the choice between the four different methods (bucket, quantile, pseudo-quantile, Winsorized) in HPBIN are a difficult matter. Using these methods may require much manual input, making binning a time consuming process with large datasets, all of which are avoided with Autobin.

Future research ideas include the comparison of Autobin with even more binning methods, the extension of Autobin to be able to handle also the case of continuously distributed predictors and other requirements such as estimators that are monotonic in the predictor if needed. In this paper, random splitting was used, but a future research idea will be to consider whether balanced splitting will improve the results.

**Acknowledgements.** The author acknowledges that this research idea benefited from input by the referees and by Prof. Hennie Venter and the author would like to extend her appreciation for these valuable contributions to this paper. This work is based on research supported in part by the Department of Science and Technology (DST) of South Africa. The grant holder acknowledges that opinions, findings and conclusions or recommendations expressed in any publication generated by DST-supported research are those of the author(s) and that the DST accepts no liability whatsoever in this regard.

## References

Anderson, R. (2007). *The Credit Scoring Toolkit: Theory and Practice for Retail Credit Risk Management and Decision Automation*. Oxford University Press, Oxford.

Baesens, B., Rosch, D., and Scheule, H. (2016). *Credit Risk Analytics: Measurement Techniques, Applications, and Examples in SAS*. Wiley & Sons, Hoboken, New Jersey.

Barnard, G. (1974). Discussion of "Cross-validatory choice and assessment of statistical predictions" by M. Stone. *Journal of the Royal Statistical Society. Series B (Methodological)*, **36**, 133–135.

Beynon, M. (2004). Stability of continuous value discretisation: an application within rough set theory. *International Journal of Approximate Reasoning*, **35**, 29–53.

Beynon, M. J. and Peel, M. J. (2001). Variable precision rough set theory and data discretisation: An application to corporate failure prediction. *Omega*, **29**, 561–576.

Breiman, L., Fredman, J. H., Olsen, R. A., and Stone, C. J. (1984). *Classification and Regression Trees*. Wadsworth, Pacific Grove, California.

Cantú-Paz, E. (2001). Supervised and unsupervised discretization methods for evolutionary algorithms. *In Workshop Proceedings of the Genetic and Evolutionary Computation Conference 2001, San Francisco, CA*. 213–216. San Francisco.

Chen, S., Tang, L., Liu, W., and Li, Y. (2011). A improved method of discretization of continuous attributes. *Procedia Environmental Sciences*, **11**, 213–217.

Chmielewski, M. R. and Grzymala-Busse, J. W. (1996). Global discretization of continuous attributes as preprocessing for machine learning. *International Journal of Approximate Reasoning*, **15**, 319–331.

Faraway, J. J. (2016). Does data splitting improve prediction? *Statistics and Computing*, **26**, 49–60.

Fayyad, U. M. and Irani, K. B. (1992). On the handling of continuous-valued attributes in decision tree generation. *Machine Learning*, **8**, 87–102.

Greif, T. (2013). R Credit Scoring – WoE & Information Value in woe Package.
URL: *https://www.r-bloggers.com/r-credit-scoring-woe-information-value-in-woe-package*

Kontkanen, P. and Myllymäki, P. (2007). MDL histogram density estimation. *In Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics*. Puerto Rico, 219–226.

Lee, C. H. (2007). A Hellinger-based discretization method for numeric attributes in classification learning. *Knowledge-Based Systems*, **20**, 419–425.

Liu, H., Hussain, F., Tan, C., and Dash, M. (2002). Discretization: An enabling technique. *Data Mining and Knowledge Discovery*, **6**, 393–423.

Lund, B. (2017). SAS® macros for binning predictors with a binary target. *In Proceedings of the SAS Global Forum 2017 Conference*.
URL: *http://support.sas.com/resources/papers/proceedings17/0969-2017.pdf*

Lund, B. and Raimi, S. (2012). Collapsing levels of predictor variables for logistic regression and weight of evidence coding. *In Proceedings of the Midwest SAS Users Group 2012 Conference*.

Mosteller, F. and Tukey, J. W. (1977). Data analysis and regression: A second course in statistics. *In Series in Behavioral Science: Quantitative Methods*. Addison-Wesley, Reading.

Nguyen, H. V., Müller, E., Vreeken, J., and Böhn, K. (2014). Unsupervised interaction-preserving discretization of multivariate data. *Data Mining Knowledge Discovery*, **28**, 1366–1397.

Okumura, H. (2011). Kernel regression for binary response data. *Memoirs of the Faculty of Science and Engineering Shimane University. Series B. Mathematical Science*, **4**, 33–53.

Oliveira, I., Chari, M., and Haller, S. (2008). SAS/OR®: Rigorous constrained optimized binning for credit scoring. *In Proceedings of the SAS Global Forum 2008, Cary, North Carolina*.

Picard, R. R. and Berk, K. N. (1990). Data splitting. *The American Statistician*, **44**, 140–147.

Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, California.

Roy, A. and Pal, S. K. (2003). Fuzzy discretization of feature space for a rough set classifier. *Pattern Recognition Letters*, **24**, 895–902.

SAS Institute (2014). Base SAS® 9.4 Procedures Guide: High-Performance Procedures PROC HPBIN, Third Edition.
URL: *http://support.sas.com/documentation/cdl/en/prochp/67530/HTML/default/viewer.htm#prochp_hpbin_overview.htm*

SAS Institute (2015). Applied Analytics Using SAS Enterprise Miner (SAS Institute Course Notes).

Siddiqi, N. (2006). *Credit Risk Scorecards: Developing and Implementing Intelligent Credit Scoring*. Wiley & Sons, Hoboken, New Jersey.

Siddiqi, N. (2017). *Intelligent Credit Scoring: Building and Implementing Better Credit Risk Scorecards.* Wiley & Sons, Hoboken, New Jersey.

Stone, M. (1974). Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society. Series B (Methodological)*, **36**, 111–133.

Therneau, T. M. and Atkinson, E. J. (2017). An introduction to recursive partitioning using the RPART routines.
URL: *https://cran.r-project.org/web/packages/rpart/vignettes/longintro.pdf*

Therneau, T. M. and Atkinson, E. J. (2018). *rpart: Recursive Partitioning and Regression Trees*.
URL: *https://CRAN.R-project.org/package=rpart*

Thomas, L. C. (2009). *Consumer Credit Models: Pricing, Profit and Portfolios*. Oxford University Press, Oxford.

Wielenga, D., Lucas, B., and Georges, J. (1999). Enterprise Miner: Applying data mining. Technical report, SAS Institute, Cary, North Carolina.

Manuscript received 2017-10-31, revised 2018-02-27, accepted 2018-04-14.