Journal of Marketing Analytics
https://doi.org/10.1057/s41270-021-00145-4

ORIGINAL ARTICLE

Prediction and profitability in market segmentation typing tools

Marco Vriens[1] · Nathan Bosch[1] · Chad Vidden[2] · Jason Talwar[3]

Revised: 3 July 2021 / Accepted: 11 November 2021

Abstract
A vital component in strategic segmentation is the typing tool. Little is known about their prediction performance. Even less is known how well they perform at the segment-level, in imbalanced situations, and how well they predict the smallest (minority) segment. We investigate using simulated and real-life data, how well typing tools perform overall and at the specific segment-level and we show the following. One, even when overall prediction accuracy is good, specific segments may be predicted poorly. Two, for valuable (minority) segments with high targeting costs misclassification can have a substantial impact on the profitability of the segmentation strategy. Poor prediction of a minority segment can happen in high and mildly imbalanced segments. Three, prediction of minority segments can vary substantially across different base classifiers and across imbalance correction methods. We find that performance can vary substantially across base classifiers and that support vector machines, overall, perform best. Four, the prediction of a (minority) segment can always be improved by using imbalance correction methods, and overall random under-sampling performs best.

Keywords Strategic market segmentation · Typing tools · Imbalance correction methods · Classification

Introduction

Market segmentation is one of the most strategic research projects a company can undertake. Often commissioned with high expectations to impact product differentiation, brand architecture, marketing communication and sales efforts (e.g. Smith 1956; Wedel and Kamakura 2000; Yankelovich and Meer 2006). Yet, often ending in disappointment (e.g. Thoeni et al. 2016) because the segmentation is not compelling enough (e.g. Young et al. 1978), hard to interpret or to translate to segment-level marketing initiatives (e.g. Dibb and Simkin 2001; Dolcinar and Lazarevski 2009), or isn't aligned with other marketing initiatives (e.g. Gibson 2001; Dibb and Simkin 2010). According to Christensen et al. (2005), failed segmentation projects can also lead to new product introduction failures.

Strategic market segmentation has two analytical steps. The identification of the market segments and the development of a typing tool that predicts who falls in which segment. Most research on segmentation has been dedicated to developing and testing new segmentation identification methodologies (for example see Wedel and Steenkamp 1989; Hiziroglu 2013). Several academic and commercials reviews have been published summarizing and reviewing progress and identifying challenges in segmentation. Academic reviews can be found in Wind (1978), Beane and Ennis (1987), Wedel and Kamakura (2000), Allenby et al. (2002), Dolcinar (2002), Tuma et al. (2011), Liu et al. (2012). Vidden et al. (2016), Trevenen (2018), and Mora Cortez et al. (2021). Practitioners also provided frameworks and reviews (e.g. Thomas 1980; Vriens 2001; Andersen and Ho 2011), In practice, one does not know the segments in advance and there are no objective criteria by which we can judge the

✉ Marco Vriens

Nathan Bosch
Nathan.bosch@teamkwantum.com

Chad Vidden
cvidden@uwlax.edu

Jason Talwar
jason_talwar@brown.edu

[1] Kwantum Analytics, Sausalito, CA, USA

[2] Department of Mathematics, University of Wisconsin, La Crosse, 1725 State St., La Crosse, WI 54650, USA

[3] Brown University School of Engineering, Providence, RI, USA

success of the segmentation.[1] Typically, marketing management reviews a small set of possible segmentation solutions (e.g. Kim and Lee 2011), from which they choose the most actionable solution. The second step then involves profiling and developing a typing tool. A typing tool identifies variables that can predict which respondents belong to which segment.[2] Firms value a typing tool for three reasons. First, to enable accessibility of the segments. Accessibility is one of the original segmentation quality evaluation criteria (e.g. see Kotler 1967; Dibb 1999; Pires et al. 2011). For example, if demographic/firmographic and media usage variables predict segment memberships. This information can then be used to reach these segments with communication strategies. Second, to identify a small set of predictor variables (i.e. survey questions) which can be included in future survey studies. Respondents from these new surveys can now be assigned to the existing strategic segments. Thereby deepening the understanding of the strategic segments. Third, to score the customer database. Some firms have large customer databases (e.g. tens of thousands of customers). If the typing tool variables are in the database, then the firm can predict in which segment each individual customer belongs to. If the identified market segments are different in terms of customer needs, behaviors, or attitudes, then the firm can easily develop separate marketing programs for different customers in the database. This contributes substantially to the usefulness of the segmentation study because results can now become part of the firm's CRM and sales activities.

However, developing typing tools can be challenging. In most practical situations it is hard to develop a typing tool that has a high classification accuracy. From our experience, companies would like to see the overall accuracy in the range of 80% to 90% (or more) but may have to accept lower hit rates in some cases, say in the range of 50% to 70%. Low accuracy is likely when the active segmentation variables are different from the passive segmentation (profiling) variables (Liu et al. 2010). For example, if segments are based on product attribute importance and profiling is based on firmographics and media usage variables. To our knowledge very little has been published about how well typing tools perform overall (an exception is the study by Reibstein 1987). Another challenge is that not all segments are predicted with the same accuracy. There are two reasons why segment-level prediction is important in strategic segmentation. In situations where segments differ substantially in size (referred to as class imbalance), the smallest segment (referred to as minority segments) is often harder to predict. If this minority segment is valuable or strategic for the firm, accuracy below a certain level may simply not be acceptable. For example, in one of the commercial studies the authors were involved in, the smallest segment was predicted with 0% accuracy which was not acceptable to the client. Note, from an academic perspective we might consider merging the small segment with another segment and thereby eliminating the problem. However, in commercial research it is not the marketing research or analytics teams that gets to decide this, it is up to the marketing decision makers to decide this. In this commercial study, the minority segment was the most meaningful and most valuable segment. Hence, the firm needed to know accurately what this segment was about and merging this segment with another segment was not an option. Two, any given segment, regardless of its' size, may be more important to the firm than the other segments, and for that high-value segment management may wish to have a better prediction accuracy rate than for the other segments (e.g. Bult 1993).

In the market segmentation literature, there are several gaps regarding typing tools. One, surprisingly little has been published about typing tools. Some papers do mention a profiling step, but most fail to even mention that a typing tool is a vital step needed to ensure a successful commercial implementation. Hence, little is known about how well typing-tools work. Two, even less is known about how well these typing tools perform at the specific segment-level. Third, though studies have evaluated the effects of imbalance correction methods in the context of binary classification problems (e.g. credit scoring; direct mail, fraud detection, student success, cancer identification, etc.), we could not find many papers on how well base classifiers and imbalance correction work in multi-segment situations (an exception being Nanni et al. 2015). In binary classification the imbalance can be quite severe (i.e. 1:100, 1:1000). Predictive tools can achieve high accuracies by just predicting the majority class. Therefore, it is easy to see why such situations can lead to poor minority segment prediction. In strategic (multi-segment) segmentation we are more likely to encounter mild or modest imbalance. We have encountered poor prediction of a minority segment in a mild imbalance situation (see "Segment imbalance in strategic segmentation" section). Given these gaps in the literature, our paper aims to make several contributions by showing:

1. How typing tools perform, overall and at the segment-specific level, in a variety of (un) balanced situations. In multi-segment situations membership prediction of a specific (minority) segment can substantially be compromised, even when the overall accuracy is good and even in *mild* imbalance situations.

[1] Unless predictive segmentation approaches are used such as conjoint segmentation (e.g. DeSarbo et al. 1992) or tree-based approaches such as CHAID (e.g. Magidson 1994).

[2] The reason why this is often called a typing tool is because firms in addition to the predictive model need a simple software solution (e.g. an Excel macro) that will automatically map customers in the database into the identified market segments.

2. That if a minority segment is more valuable than others, and has a higher targeting and acquisition costs, different levels of misclassification can have substantial effects on profits. We show this analytically and empirically.
3. How segment-level prediction varies across different base classifiers and imbalance correction methods. Improving, especially minority segment prediction can be vital as small or minority segments can be important to the organization (e.g. Tonks 2009; Allaway et al. 2014). Specifically, we study logistic regression (LR), Decision Trees (DT), Random Forests (RF), Gaussian Naïve Bayes (NB), Gradient Boosting (GB) and Support Vector Machines (SVM). We find that overall support vector machines (SVMs) are better able to handle segment imbalance than other base classifiers. Various imbalance correction methods can improve segment-level prediction. Specifically, we study random under-sampling (RUS), random over-sampling (ROS), weighting, and synthetic minority over-sampling (SMOTE). Overall, we find that RUS performs best.

We use a simulation study and two empirical datasets. In the simulation study we systematically review how factors such as segment separability, number of segments, segment imbalance and sample size affect (minority) segment-level performance and how it can be best mitigated. We show that, not surprisingly, separability of the segments has the biggest effect on accuracy, followed by numbers of segments and lastly segment imbalance. Support vector machines and random under-sampling fared best overall. In empirical study one, we have a typing tool that is based on passive segmentation variables, and in empirical study two we have a typing tool that is based on active segmentation variables and one that is based on passive variables. We find that mild imbalance can be a significant problem and can be substantially mitigated by using imbalance correction methods and by using a variety of base classifiers. Here too, we find that using various base classifiers and imbalance correction methods yield substantial improvement. We also show that prediction improvements lead to higher expected profitability.

Segment imbalance in strategic segmentation

The ratio between the smallest segment (in the literature this is referred to as the minority segment) and the largest segment (called the majority segment) is referred to as class imbalance (or segment imbalance), if one class is substantially larger than the other. In business there are many scenarios where we deal with two-segment problems, e.g. customer churn (e.g. Lemmens and Croux 2006), Direct Marketing (e.g. Bult 1993), switchable consumer analysis (e.g. Gensch et al. 1990), fraud detection (Ngai et al. 2011), and student drop-out and course performance (e.g. Coussenment et al. 2020), etc. In these types of applications, we may encounter (extreme) imbalance. For example, in some industries the churn rate may only be around 2% (e.g. Lemmens and Croux 2006; Zhu et al. 2017). The percentage of respondents who respond to a direct mail offering can be very small, say 1% or 2% (e.g. see Bult 1993; Lemmens and Croux 2006), though can also be higher, say around 25% (Coussement et al. 2014). In such two-segment situations, the minority class memberships prediction is often worse than the majority class memberships prediction (e.g. Japkowicz and Stephen 2002; Estabrooks et al. 2004; Ahn et al. 2021). Several studies also find that extreme imbalanced datasets can lead to lower overall prediction accuracy (e.g. Estabrooks et al. 2004; Marqués et al. 2013; Marinakos and Daskalaki 2017).

Strategic marketing applications, though, are different as they typically result in more than two segments and it is less likely, though not impossible, that we encounter severe imbalance. We reviewed 55 segmentation studies published in marketing and business journals where we identified the number of segments and calculated the imbalance (See Appendix 1). Appendix 1 shows that in nineteen studies a three-segment solution was identified as the optimal solution, and thirty-two studies had four or more segments. Only four studies identified a two-segment solution as the optimal solution. The appendix also shows that only eight had roughly balanced segment sizes. In eleven studies the imbalance was 1:9 or higher. Hence, most solutions had mild to modest imbalance.

Severe imbalance does not occur often in strategic segmentation, but even modest imbalance can be a problem. Consider the following example (selected from our empirical study 1). The company had identified four market segments. The number of respondents in segment two was small relative to other segments (49 vs 224 for the largest segment): an imbalance of roughly 1:4. The initial overall classification accuracy of the typing tool for the four-segment solution was 58% (using LR) but segment two was predicted with zero percent accuracy.

It is important to note that smaller segments are not per se less important than larger segments. Segments can be attractive for a variety of reasons such as expected segment growth, relative market share, competitive intensity (e.g. a small segment could be a white space where there is no competition yet), entry barriers and purchasing power (Tonks 2009) or expected profitability (Liu et al. 2010). The reason why differences in segment-level prediction matter is because different levels of misclassification rates for different segments can affect the profitability of a segmentation strategy:

Let:

S_i be the estimated (true) number of consumers in segment i.

T_i be the correctly predicted consumers (buyers) in segment i.

GM_i be the gross margin (revenue minus cost) of the buyers in segment i.

C_i be the individual campaign cost for a non-buyer in segment i.

GP be the gross profit of running a segment-specific campaign.

Then, the gross profit of running a segment-specific campaign is:

$$GP = \sum_i \left(\left(T_i * GM_i \right) - \left(S_i - T_i \right) * C_i \right) \tag{1}$$

From (Eq. 1) we can see that if the campaign costs for a given segment are high (e.g. highly valuable consumers may be hard to reach and difficult to persuade), then the second component in (1) will grow while the first component will shrink. For example, say we have a segment one. with 1000 prospective consumers who we want to target. There is a segment two that is less likely to respond to our acquisition campaign. if the GM of a buyer in segment 1 is $ 500, and the cost of the acquisition campaign is $ 50 per customer, then a classification rate of 10% results in (100*$ 500) – (900*$ 50) = 5 K profit. If we were able to improve the classification to 70%, then our profit will be: (300* $ 500) – (700* $ 50) = $ 115 K. In section seven, we analyze this topic more extensively.

Imbalance correction methods

To mitigate the negative effects of imbalance re-sampling and algorithmic approaches have been proposed (e.g. He and Garcia 2009). Within the re-sampling methods we can distinguish between random under-sampling (RUS), random over-sampling (ROS), and non-random over-sampling. Within the non-random over-sampling approaches we count weighting and synthetic over-sampling.

In random over-sampling we over-sample the minority segments whereas in random under-sampling we randomly sub-sample from the majority segment. Over-sampling the minority class means we randomly sample units from the minority class (with replacement) until all segments have the same number of observations. When using random under-sampling, we randomly select respondents from the majority classes until both majority class and minority class have the same number of observations. Both approaches have pros and cons, and each may outperform the other in certain situations: i.e. over-sampling performed better in Marqués et al. (2013) and under-sampling was found to be better in some cases in Estabrooks et al. (2004). However, when applying under-sampling the sample size gets smaller. When using DT, this can quickly become a problem because each branch down results in increasingly smaller samples. In our study, we use both RUS and ROS.

Applying weighting to some degree is very similar to random over-sampling, though not identical as the process is not fully random. Also, the exact implementation of weighting varies by method. First, we define weights as:

$$w_i = \frac{n}{c \cdot n_i} \tag{2}$$

where w_i is the ith class weight, n is the total number of respondents, c is the number of classes, n_i is the number of respondents in class i

The weighting formula (Eq. 2) aims to give higher weight to minority classes and lower weight to majority classes. This is done uniformly across classes (respondents in the same class receive the same weight) and relative to class sizes as well as the total number of respondents.

The weights in LR are added into the cost function (King and Zeng 2001):

$$\sum_{i=1}^{n} w_i \left(y_i \log\left(h\left(y_i \right) \right) - \left(1 - y_i \right) \log\left(1 - h\left(y_i \right) \right) \right) \tag{3}$$

where y_i is the correct output (class membership) and $h(y_i)$ is the softmax function (e.g. see Zeng 2009, p. 11199), also outlined in formula 3:

$$h\left(y_i \right) = \frac{e^{y_i}}{\sum_{j=1}^{n} e^{y_j}} \tag{4}$$

which returns the predicted probability that the true class is y_i. Minimization of the cost function (Eq. 3) is equivalent to maximizing the weighted Maximum Likelihood function for LR which equates to maximizing the log likelihood.

When using the weights in DT, the weights are incorporated in the calculation of the Gini index. In each node within a given branch, the class imbalance will change, and the conditional probabilities used to calculate the impurities and the overall Gini index (Gini 1936) will be weighted according to the relative class sizes in that node. Hence, there is not one universal set of weights but rather weights are updated in each step of the Greedy search algorithm as it searches for the best splits.

RF is an ensemble approach, generating a number of decision trees with the trees within the ensemble of shallow depth. Within each DT in the RF weighting is applied the same as in the DT approach. The RF prediction is based on the majority DT voting rule.

NB is a simple Bayesian model where we assume features are normally distributed and that there is independence

between the features. This makes the approach simple and efficient. There is no need to apply class weighting, as the class priors are considered in its MAP (maximum a posteriori) decision rule, as can be observed in (Eq. 5)

$$\hat{y} = \underset{k\in\{1,\ldots,K\}}{\arg\max}\, p(C_k) \prod_{i=1}^{n} p(x_i|C_k) \tag{5}$$

where $\hat{y}$ is equal to the class C_k that maximizes the previous equation. Note the prior $p(C_k)$ is already internal to the model. The simplicity of NB can be drawback in more complex classification problems.

The SVM method is a commonly used classifier that attempts to find hyperplanes (e.g. lines when the data is in two dimensions), which maximally separates two or more classes (Cortes and Vapnik 1995; Bennett and Campbell 2000). In the default case, SVM tries to find a linear separation but with Kernel functions non-linear solutions can be found. In this study, we use the Radial Basis Function (RBF) Kernel. To apply weighting to SVMs, specifically C-support Vector Classification, we employ the approach outlined in the scikit-learn software package used to run the analyses, which is based on the LIBSVM library (Chang and Lin 2011). In this approach, the C parameter, which acts as a regularizer for the SVM, is adjusted per class based on the provided class weight. In our analyses, this class weight is equal to the class priors. We refer to LIBSVM for further discussion on this.

GB is a boosting approach and generally uses DT as the base classifier similar to RF. However, in GB the trees are trained iteratively in a boosting training regime. Intuitively, this means that a new tree estimator may be trained to accommodate for the weaknesses of the previous tree. This approach generally outperforms RF (Hastie et al. 2001). For GB we did not apply weighting as there was no easily accessible option in the software packages used to run the analyses.

Non-random re-sampling can be achieved by using weights or by using synthetic samples, e.g. Synthetic Minority Over-sampling Technique (SMOTE), see Chawla et al. (2002). Several advanced non-random over-sampling techniques have been proposed. Marqués et al. (2013), studying binary credit scoring, found that all these alternatives yielded similar performance metrics, although the different SMOTE varieties did yield better classification accuracy. Applying SMOTE to correct for imbalance is conceptually very similar to random over-sampling, except the over-sampling is not done in a random way. When using SMOTE, the minority class is over-sampled by introducing *synthetic* examples along the line segments joining any, or all, of the k minority class nearest neighbors. Synthetic samples are generated by taking the difference between the feature vector (sample) under consideration and its nearest neighbor:

Table 1 Simulation study design factors

Factors	Levels
Number of segments	3, 4, 5 and 6 segments
Segment imbalance	Balanced, 1:2, 1:5 and 1:10
Separability	0.5, 1, 2
Sample size	1000, 2000, and 10,000

i.e. market segments are defined by differences in the mean values of the active segmentation variables. If the feature vector for one of the neighboring segments is say (7, 1, 2), the feature vector of the minority segment is (1, 4, 5), then we look at those individual respondents who feature values are closest (in distance) to the neighboring segment. Then differences are calculated between the feature values of these borderline respondents and the neighboring segment feature values. These differences are then multiplied by a random number between 0 and 1. The subsequent resulting data points are then added it to the feature vector (i.e. the vector with respondent level values for the active segmentation variables) under consideration for the minority segment. This causes the selection of a random point along the line segment between two specific features. This data generation process is further explained in Appendix 2.

Simulation study

Simulation studies have been used to evaluate segmentation methods (e.g. Vriens et al. 1996). In our simulation study we varied four factors (See Table 1): Number of segments, segment imbalance, separability of segments and sample size. We employed a full factorial design ($4^2 \times 3^2$) resulting in 144 datasets.

Three levels were used for separability of the segments: 0.5, 1.0, 2.0, where a lower value indicates that segments are closer together and may be harder to separate, and the class imbalance (ratio between majority and minority class: 1:1, 1:2, 1:5, 1:10). This represents the imbalance encountered in published segmentation studies.

Data generation procedure

We use a data generation approach outlined in the Scikit-Learn python package (sklearn.datasets.make_classification; see Pedregosa et al. 2011), which is inspired by Guyon (2003). This approach generates multiple classes, the class imbalance of which can be set manually. The approach generates an *N*-dimensional hypercube, where *N* refers to the number of (informative) features present in the dataset. Classes are generated centered around vertices of the hypercube using multidimensional normal distributions with

standard deviation 1.0. The distance between classes, i.e. the length of side of the hypercube, is determined by the "separability" parameter and can be expressed as (turn into equation: length = 2*separability). We use three features (all informative) and only 1 cluster per class.

Analysis

We use several base classifiers: Logistic Regression (LR), Decision Trees (DT), Random Forest (RF) naïve Bayes (NB), Gradient Boosting (GB) and Support Vector Machines (SVM). Our goal is not a comprehensive comparison of base classifiers. Our set constitutes a range from basic (LR, DT, NB) to more advanced (RF, GB, and SVM) so that we can get insight into whether base classifiers have an impact on minority segment prediction.

Many studies have studied the relative empirical performance of base classifiers (e.g. Bult 1993; Bequé and Lessman 2017; Coussement and van der Poel 2008; Gordini and Veglio 2017; Hong et al. 2005; McCarty and Hastak 2007; Miguéis et al. 2018; West et al. 1997). In most of these studies the difference between various methods is often quite small. According to Marqués et al. (2013) it is impossible to identify one classifier that is generally superior to others. Instead, data characteristics may have a bigger impact on the classification performance (such as segment imbalance).

For LR, DT, RF, GB, SVM, and NB we used the scikit-learn Python package. DT was trained using Gini Impurity as the splitting method (see Breiman et al. 1984). For all algorithms and analyses, we ran a grid search over several possible hyperparameter settings. For RF and GB, we search over several numbers of decision tree estimators, namely 5, 10, 50, and 100 trees. For the SVM, we search over several values for the C and gamma hyperparameters, namely C: [0.1, 1.0, 10] (where a higher value indicates less regularization) and the gamma hyperparameter: $[\frac{1}{\text{number of features}}, \frac{1}{\text{number of features} \times \text{variance}(X)}]$, where X is the input data.

To see if any negative effects of imbalance on segment-membership predictions can be mitigated, we applied several imbalance correction approaches: RUS, ROS (e.g. He and Garcia 2009), weighting (e.g. He and Garcia), and SMOTE (Chawla et al. 2002) as described in "Imbalance correction methods" section. Again, our goal here is not a comprehensive comparison of imbalance correction methods—it is to demonstrate that imbalance correction methods can improve minority-segment prediction.

Hyperparameters need to be selected manually. To emulate a real-world scenario, we run a grid search over a set of possible hyperparameters for each classifier for every simulated dataset. This grid search is performed using cross-validation on the classifiers on the training set, after which we evaluate the performance on the test set. We use a 3:1 train to test data ratio. In the case that a re-sampling imbalance correction method is applied, this re-sampling is only done on the training set. Due to the stochastic nature of dataset generation and randomized imbalance correction methods, we average all achieved results over five simulation runs.

Evaluation

For imbalanced situations, the overall classification accuracy (referred to as hit rate) does not suffice as the sole measure of classification success (e.g. Barandela et al. 2003). Hence, to evaluate performance, we look at overall performance via weighted and unweighted hit rate), the weighted and unweighted F1 score, and the classification accuracy at the segment level. Let TP (True Positives) be the true positives (the number of respondents the model correctly predicts in the correct segment). True Negatives (TN) is the number of respondents correctly predicted not to belong to segments to which they don't belong too. False Positives (FP) is the number of respondents that the model incorrectly predicts to belong to a certain segment to which they don't belong. False Negatives (FN) is the number of respondents that the model predicts not be in a certain segment whereas they do belong to that certain segment. We calculate an overall hit rate as (TP + TN)/(TP + FP + TN + FN). We calculate segment-specific classification accuracy as TP/(TP + FP). Weighted accuracy refers to the standard definition of classification accuracy (defined above), whereas unweighted accuracy is equal to the mean segment-specific classification accuracy. Weighted F1-Score and unweighted F1-Score are similar. F1-Score is equal to the harmonic mean between a classifier's precision ($\frac{\text{TP}}{\text{TP+FP}}$) and recall ($\frac{\text{TP}}{\text{TP+FN}}$) (e.g. Van Rijsbergen 1979):

$$\text{F1 - Score} = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} = \frac{\text{TP}}{\text{TP} + \frac{1}{2}(\text{FP} + \text{FN})} \tag{6}$$

In a multi-segment classification scenario, this would produce an F1-Score for each segment. To aggregate these based on class priors is referred to as the weighted F1-Score. The mean over all segment-specific F1-Scores we refer to as the unweighted F1-Score. The metrics are all calculated for out-of-sample.

Results

The results of the 144 simulated data sets via analysis of variance (ANOVA) is shown in Table 2.

As Table 2 shows, we have six main effects. For overall accuracy, all main effects are statistically significant except the segment imbalance factor which is not significant for the weighted accuracy measure. All main effects are also

Table 2 *F*-tests of main and interaction effects (Simulation study)

Factors	Accuracy		F1		Accuracy		F1	
	Weighted	Unweighted	Weighted	Unweighted	Majority class	Minority class	Majority class	Minority class
Imbalance correction method (df = 4)	29.26	11.84	20.12	24.36	49.84	16.69	244.43	18.45
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Number of classes (df = 1)	2068.40	1961.15	2116.59	1810.23	1912.14	2315.22	1355.08	1722.16
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Sample size (df = 1)	37.67	61.03	42.10	59.18	19.83	8.08	94.81	83.78
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Separability (df = 1)	16,583.54	19,274.84	16,362.02	20,228.68	12,038.31	13,840.90	11,525.79	16,318.03
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Weight ratio (df = 3)	0.00	75.49	4.15	188.56	9.45	32.92	672.70	2082.01
	0.72	0.00	0.01	0.00	0.00	0.00	0.00	0.00
Classifier (df = 5)	343.39	319.17	328.40	381.89	302.87	273.66	86.23	240.93
	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Imbalance correction method*Number of classes (df = 4)	0.00	0.00	0.00	0.00	0.00	0.00	4.54	0.01
	0.75	1.17	0.72	0.47	0.46	0.56	0.00	1.48
Imbalance correction method*Number of samples (df = 4)	2.04	0.00	0.01	0.00	0.01	0.00	0.01	0.01
	0.09	0.94	1.64	1.56	1.64	0.93	1.27	1.70
Imbalance correction method*Separability (df = 4)	8.41	3.88	5.10	2.76	17.59	5.43	93.25	8.77
	0.00	0.00	0.00	0.03	0.00	0.00	2e-16)	0.00
Imbalance correction method*Weight ratio (df = 12)	6.95	2.28	4.41	6.10	7.84	3.39	48.37	6.29
	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00
Imbalance correction method*Classifier (df = 18)	1.79	1.27	1.22	1.88	2.98	0.93	18.92	3.40
	0.02	0.20	0.24	0.01	0.00	0.54	0.00	0.00

statistically significant for the minority class predictive accuracy. Of the interaction effects, only the interaction effects between imbalance correction method and separability and imbalance correction method and segment imbalance are significant.

In Table 3 we show the mean performance under the different classifiers.

Table 3 shows that on average all classifiers perform well on overall accuracy though SVM is slightly better overall, and the more advanced approached (SVM, RF, and GB) outperform the others (LR, DT, NB). In Table 4 we show the mean performance under the various levels of the design factors.

We see that overall performance gradually decreases as the number of segments increases, slightly increases with increasing sample size. Separability has a more substantial impact on overall performance while segment imbalance does not have a big effect on overall performance and gets smaller under imbalance correction methods.

In Tables 5 and 6 we show specifically the accuracy for minority class accuracy (hit rate).

Table 5 shows that in most cases, applying an imbalance correction method results in an improvement of the prediction of the minority segment. Overall, performance is best when random under-sampling is combined with the SVM classifier. Table 6 shows that applying imbalance correction methods lead to the biggest gains when the segments are not well separated or when we have a relative high degree of segment imbalance. In both cases, performance increases by almost 50%.

Empirical study 1

The empirical data of this study comes from a commercial project we were involved in. To protect the confidentiality of our client, we cannot the name the firm nor the category other than this was a durable consumer good. Their brands were sold by small independent services firms, managed by professionals. The study pertains to a survey in three countries (the U.S., Germany and France). The basis for the segments was a series of MaxDiff questions (e.g. Louviere et al. 2015), analyzed using a latent class anchored MaxDiff

Table 3 Mean overall performance of imbalance correction methods and classifiers (Simulation study)

Classifiers		Imbalance correction methods				
		Unbalanced	Weighted approach	Random under-sampling	Random over-sampling	SMOTE
DT	Unweighted accuracy	0.81	0.81	0.80	0.81	0.82
	Weighted accuracy	0.84	0.84	0.80	0.84	0.83
	F1 score	0.84	0.84	0.81	0.84	0.83
RF	Unweighted	0.85	0.85	0.84	0.85	0.86
	Weighted	0.88	0.88	0.85	0.88	0.87
	F1 score	0.87	0.87	0.85	0.87	0.87
GB	Unweighted	0.84	NA	0.83	0.85	0.85
	Weighted	0.87		0.84	0.87	0.86
	F1 score	0.87		0.84	0.86	0.86
SVM	Unweighted	0.86	0.87	0.86	0.87	0.87
	Weighted	0.89	0.88	0.87	0.88	0.88
	F1 score	0.89	0.88	0.87	0.88	0.88
NB	Unweighted	0.77	NA	0.78	0.78	0.79
	Weighted	0.80		0.79	0.79	0.79
	F1 score	0.80		0.79	0.79	0.79
LR	Unweighted	0.76	0.78	0.78	0.78	0.78
	Weighted	0.80	0.79	0.78	0.79	0.79
	F1 score	0.80	0.79	0.79	0.79	0.79

model. An optimal four-segment solution was deemed useful and actionable by management, and a typing tool model was requested. In this application, the firm wanted to predict segment membership based on variables that the client also had in their database with the purpose of "scoring" the entire database on the four segments: i.e. assign a segment membership to each customer in the database. Thirteen database variables were found common between the survey and the database: i.e. a combination of demographics (e.g. gender, age) and firmographics (revenue of the firm, number of outlets, years in business, experience of management, etc.). The relative segment sizes identified in this study: 24%, 9%, 42% and 23%; an imbalance of approximately 1:4. As is common, a standard typing (classification) tool had been developed using LR. The initial overall out-of-sample classification accuracy of the typing tool was 58%. However, when classification accuracy was evaluated at the segment-level, substantial differences were found. For example, segment two was predicted with zero percent success (out-of-sample). The client commented that the segment that was predicted worst, segment two, was the one they were most interested in.

Analysis and evaluation

We use the same base classifiers, imbalance correction methods, and evaluation metrics as in the simulation study.

Results

In Table 7 we show the overall and segment-level classification performance (hit rate and F1 score, both weighted and unweighted) for the six base classifiers, and for the four imbalance correction methods.

First, Table 7 shows that overall unweighted classification accuracy is modest (roughly between 51 and 61%). Two, by applying various base classifiers without using any imbalance correction methods, we can increase the overall unweighted hit rate to 61%, and we can get the prediction of the valuable minority segment up to 14% without sacrificing too much prediction of the other segments. Note, we can get the prediction of the minority segment up to 86% but this would lead to a 0% prediction for segment three. Three, when applying imbalance correction methods, we can get the minority segment prediction further up; depending how much we want to keep the prediction of the other segments as much as possible intact. If one is willing to sacrifice the prediction of the other segments, we can achieve 91% for the minority segment. Combining RUS with SVM accomplishes a 32% prediction success while still maintaining decent predictions for the other segments. In terms of overall accuracy, there is no single base classifier that performs best. In terms, of segment-level performance, SVM seems to fare best as the accuracy is spread, fairly, evenly across segments whereas other classifiers tend to result in larger across-segment differences.

Table 4 Mean performance under levels of the factors (Simulation study)

Factors			Unbalanced	Weighted approach	Random under-sampling	Random over-sampling	SMOTE
Number of segments	3	Unweighted	0.87	0.88	0.88	0.88	0.88
		Weighted	0.90	0.90	0.88	0.89	0.89
	4	Unweighted	0.83	0.84	0.83	0.84	0.84
		Weighted	0.86	0.86	0.84	0.85	0.85
	5	Unweighted	0.80	0.81	0.80	0.81	0.81
		Weighted	0.83	0.83	0.80	0.82	0.82
	6	Unweighted	0.77	0.78	0.76	0.78	0.78
		Weighted	0.80	0.80	0.77	0.79	0.79
Sample size	1000	Unweighted	0.81	0.82	0.80	0.82	0.82
		Weighted	0.84	0.84	0.81	0.83	0.83
	3000	Unweighted	0.82	0.83	0.82	0.83	0.83
		Weighted	0.85	0.85	0.82	0.84	0.84
	10,000	Unweighted	0.82	0.84	0.83	0.83	0.83
		Weighted	0.85	0.85	0.83	0.84	0.84
Separability	0.5	Unweighted	0.65	0.67	0.65	0.67	0.67
		Weighted	0.70	0.70	0.66	0.69	0.68
	1	Unweighted	0.83	0.85	0.84	0.84	0.85
		Weighted	0.86	0.87	0.84	0.86	0.86
	2	Unweighted	0.96	0.97	0.96	0.97	0.97
		Weighted	0.97	0.97	0.96	0.97	0.97
Class imbalance ratio	1	Unweighted	0.83	0.84	0.83	0.83	0.84
		Weighted	0.83	0.84	0.83	0.83	0.84
	2	Unweighted	0.83	0.84	0.83	0.83	0.83
		Weighted	0.83	0.84	0.83	0.83	0.84
	5	Unweighted	0.81	0.83	0.81	0.83	0.83
		Weighted	0.83	0.84	0.83	0.83	0.84
	10	Unweighted	0.79	0.81	0.79	0.81	0.82
		Weighted	0.83	0.84	0.83	0.83	0.84

Table 5 Mean minority segment performance by classifiers (Simulation study)

Classifiers	Imbalance correction methods					Max improvement
	Unbalanced	Weighted approach	Random under-sampling	Random over-sampling	SMOTE	
DT	0.71	0.71	**0.79**	0.71	0.75	0.08
RF	0.72	0.71	**0.82**	0.75	0.79	0.10
GB	0.70	NA	**0.82**	0.78	0.81	0.12
SVM	0.73	0.83	**0.84**	0.82	0.83	0.11
NB	0.61	NA	**0.77**	0.77	0.77	0.16
LR	0.60	0.77	**0.77**	0.77	0.77	0.17

In terms of the minority segment, NB performs very well in combination with random over-sampling and SMOTE, but this comes at the cost of other segments being predicted very poorly.

Empirical study 2

Study 2 was a strategic segmentation study, also pertaining to a consumer durable product, and was commissioned to