Overlapping classification for autocoding system

Yukako Toko
(ytoko@nstac.go.jp)
National Statistics Center, Japan

Shinya Iijima
(siijima@nstac.go.jp)
National Statistics Center, Japan

Mika Sato-Ilic
(msato@nstac.go.jp)
National Statistics Center, Japan / University of Tsukuba

ABSTRACT

Coding is the classification of objects (or features) based on given classification codes, and it is frequently required in the field of official statistics. This paper proposes a supervised overlapping multiclass classifier for autocoding. The classifier is implemented in R.

The purpose of this study is to efficiently apply this classifier to the coding task of the Family Income and Expenditure Survey in Japan. We previously developed a non-overlapping multiclass classifier that obtains "exclusive" classes. Even though the developed classifier provides high accuracy for the autocoding task, some objects with ambiguous input information are still incorrectly assigned codes. This shows that exclusive classification has a limitation when dealing with uncertainty. To solve this problem, we propose a new classifier that lists multiple candidates in descending order of the degree of reliability as output and assists experts in selecting a correct code from the listed candidate codes. We refer to this proposed classifier as the overlapping multiclass classifier. A new reliability score based on the weights of entropy is employed in the proposed classifier. With this new reliability score, the proposed classifier improves cumulative accuracy and practicability while the advantages of the structural simplicity of the algorithm and practical calculation time remain unchanged. The proposed algorithm is implemented in R to improve its versatility.

Keywords: *Coding, Machine learning, Overlapping classification*
JEL Classification*: C38*

1. INTRODUCTION

Coding is an important activity in the field of official statistics as there are certain text response fields in governmental survey questionnaires that are required to be translated into a given classification for data processing.

In recent years, the importance of autocoding, that is, automated coding, is increasing with the improvement of computer technology. Therefore, the methods of autocoding have been discussed in the field of official statistics. For example, Hacking and Willenborg (2012) presented methods for coding including autocoding techniques in the field of official statistics. They reported the problem of handling ambiguous written text. In addition, they used a weight for each word to indicate how specific the word is in the training dataset for code assignment through automated coding. Gweon et al. (2017) introduced methods for automated occupation coding based on statistical learning.

Originally, we developed a multiclass classifier for autocoding based on a simple machine learning technique (Toko et al., 2017; Tsubaki et al., 2017; Shimono et al., 2018) to contribute to the improvement of the efficient coding of *the Family Income and Expenditure Survey* in Japan. The classifier assigns classification codes (or classes) with certain level of high accuracy, and it provides the advantages of simplicity and practical calculation time when compared with typical machine learning methods such as deep learning and random forests. However, this classifier yielded a certain volume of unmatched output. While analyzing incorrectly classified data, we found that a classification code could not be uniquely determined for some objects (or features) because of problems such as semantics, interpretation, and insufficiently detailed input information. To address these issues, we introduced partition coefficient or partition entropy (Bezdek, 1981; Bezdek et al., 1999), which shows the classification status of each object (or feature) to the previously proposed classifier. Based on this, we proposed a new multiclass classifier considering the classification status of each object (or feature) for representing the uncertainty situation of the classification of each object (or feature) (Toko et al., 2018). Even though we found that the idea of partition coefficient and partition entropy could be effective for representing the difference in the uncertainty of the classification of objects (or features), we still experienced problems when classifying objects (or features) to exclusive classes. These problems are the semantic similarity of classes, the similarity of the interpretation of classes, and the insufficiently detailed input information of several objects (or features).

The main reason for these problems is the unrealistic restriction that one object (or feature) is classified to a single class (or code). Therefore, we propose a new classifier that allows for the assignment of one object (or feature) to multiple classes (or codes) while utilizing the idea of our previously proposed multiclass classifier considering the classification status of each object (or feature). In addition, to assist a user in the assignment of an object (or feature) to codes (or classes), we define the reliability score of each code (or class) among the codes (or classes) assigned from the object (or feature).

The reliability score utilizes the idea of partition entropy, that is, the degree of uncertainty of classification of each object (or feature) to multiple classes is based on the idea that a class that is clearly classified is a more reliable candidate to be assigned from the object (or feature).

Numerical examples show that the accuracy of the proposed classifier is improved while the advantages of structural simplicity of the algorithm remain unchanged.

The proposed classifier is developed in R for the following two reasons: First, the abundant packages available in R allow for the efficient development of the classifier. We used packages such as "dplyr" and "data. table" for the development. Second, when implemented in R, our classifier could be a good contribution to coding tasks in the field of official statistics as the number of R users is increasing in this field.

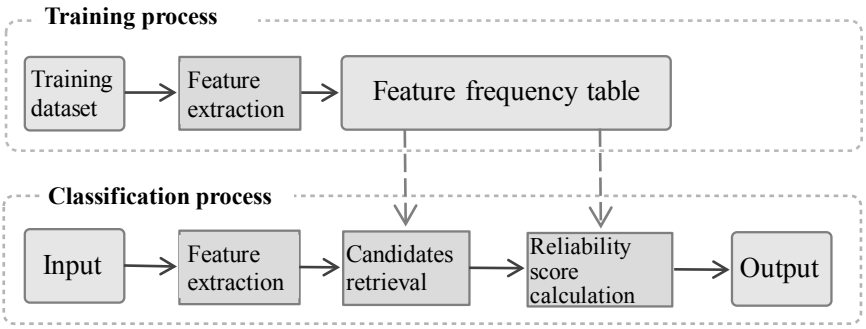
The rest of this paper is organized as follows: The previously developed autocoding systems (Toko et al., 2017; Tsubaki et al., 2017; Shimono et al., 2018; Toko et al., 2018) are described in section 2, and a novel algorithm that lists multiple candidates is proposed in section 3. The numerical examples with *the Family Income and Expenditure Survey* data are described in section 4. Conclusions and suggestions for future work are presented in section 5.

2. PREVIOUSLY DEVELOPED AUTOCODING SYSTEM

The previously developed autocoding system (Toko et al., 2017; Tsubaki et al., 2017; Shimono et al., 2018) assigns the most promising classification codes (or classes), and it comprises training and classification processes, as shown in Fig. 1.

Basic structure of our classifier

Figure 1



In the training process, features are extracted and a feature frequency table is created. We took word-level N-grams from the word sequences of text descriptions (or instances) after tokenizing each description using MeCab (Kudo et al., 2004), which is a well-known dictionary-attached morphological Japanese text analyzer. Here, unigrams (any word), bigrams (any sequence of two consecutive words), and entire sentences are considered as features (or objects). After feature extraction, the system tabulates all extracted features (or objects) based on their given classification codes into a feature frequency table.

In the classification process, the system implements the following procedures: feature extraction, the retrieval of candidates, and classification code assignment. The system performs feature extraction in the same manner as the training process. Then, each extracted feature (or object) refers to the feature frequency table provided by the training process for retrieving the corresponding classification codes (or classes), which are candidates and frequencies. Finally, it determines the most promising class to assign a classification code (or class) based on the following idea: Let multinomial classes C take values in $\{1, ..., K\}$, and let $F = (F_1, ..., F_J)$ be a J-dimensional random variable whose elements take a value of 0 or 1, which indicate the absence or presence of a particular feature (or object), respectively. Then, as each feature (or object) is assumed to be conditionally independent of any other feature (or objects) given C, the conditional probability of the features (or objects) of **F** for a given class C can be written as

$$P(F_j = f_j, j = 1, \dots, J | C = k) = \prod_{j=1}^{J} P(F_j = f_j | C = k) = \prod_{j=1}^{J} p_{jk}^{f_j} (1 - p_{jk})^{1-f_j}, \quad k = 1, \dots, K, \tag{1}$$

where $p_{jk} \circ P(F_j = 1 | C = k)$. Let n_k be the number of features (or objects) in a class k in the training dataset, and n_{jk} be the number of features (or objects) in class k in the training dataset with $f_j = 1$. The maximum likelihood estimate of p_{jk} can be written as

$$\hat{p}_{jk} = \frac{n_{jk}}{n_k}. \tag{2}$$

The posterior probability $P(C = k | F_j = f_j, j = 1, \dots, J)$ is proportional to

$$p_k \prod_{j=1}^{J} p_{jk}{}^{f_j} (1 - p_{jk})^{1-f_j}, \tag{3}$$

where p_k ° $P(C = k)$. The posterior probabilities are described as

$$P\left(C = k \middle| F_j = f_j, j = 1, \dots, J\right) = \frac{exp(\sum_{j=1}^{J} f_j \beta_{jk})}{\sum_{l=1}^{K} exp(\sum_{j=1}^{J} f_j \beta_{jl})}, \tag{4}$$

when $p_k = a\{\prod_{j=1}^{J}(1 - p_{jk})\}^{-1}, a \neq 0, \forall k$ are assumed. Where, $\beta_{jk} \equiv \log \frac{p_{jk}}{1-p_{jk}}$. Equation (4) can be rewritten as

$$P\left(C = k \middle| F_j = f_j, j = 1, \dots, J\right) = \frac{p_k \prod_{j=1}^{J} p_{jk}}{\sum_{l=1}^{K} (p_l \prod_{j=1}^{J} p_{jl})}, \tag{5}$$

when $f_j = 1$. From (5), it can be seen that $P(C = k \mid F_j = f_j, j = 1, \dots, J)$ when $f_j = 1$ is influenced only by n_{jk}. $\tilde{p}_{jk}$ is defined as

$$\tilde{p}_{jk} = \frac{n_{jk}}{n_j}, \qquad n_j = \sum_{k=1}^{K} n_{jk}. \tag{6}$$

We applied the idea of partition coefficient and partition entropy to the above classification algorithm (Toko et al., 2018). From the definition of $\tilde{p}_{jk}$ shown in (6), $\tilde{p}_{jk}$ satisfies the following conditions:

$$\tilde{p}_{jk} \in [0,1], \qquad \sum_{k=1}^{K} \tilde{p}_{jk} = 1, \qquad j = 1, \dots, J. \tag{7}$$

Then $\tilde{p}_{jk}$ shown in (7) is the probability of feature (or object) j to code (or class) k. We define the following PC_j as a value that shows the status of the classification of the j-th feature (or object) over K classification codes (or classes) based on partition coefficient.

$$PC_j = \sum_{k=1}^{K} \tilde{p}_{jk}^2, \qquad j = 1, \dots, J. \tag{8}$$

If the j-th feature (or object) is classified clearly, the value of PC_j would be close to 1. In addition, the following PE_j is defined as a value that shows the status of the classification of the j-th feature (or object) over K classification codes (or classes) based on partition entropy.

$$PE_j = -\sum_{k=1}^{K} \tilde{p}_{jk} \log_2 \tilde{p}_{jk}, \qquad j = 1, \ldots, J. \tag{9}$$

If the j-th feature (or object) is classified clearly, the value of PE_j would be close to 0. From (7), (8), and (9), PC_j and PE_j satisfy the following conditions:

$$\frac{1}{K} \leq PC_j \leq 1, \quad j = 1, \ldots, J, \tag{10}$$

$$0 \leq PE_j \leq \log_2 K, \quad j = 1, \ldots, J. \tag{11}$$

We used the value of PC_j or PE_j for assigning classification codes (or classes) as follows:

(Step 1) Assignment of classification codes (or classes), which is performed in the same manner as in our original system.

(Step 2) Creation of a partition coefficient table and a partition entropy table for each feature (or object) in the feature frequency table along with the most promising classification code (or class). First, the system determines the most promising classification code (or class) for each feature (or object) in the feature frequency table based on the frequencies in the table. Second, from (8), the system calculates the partition coefficient for each feature (or object). In the same manner, from (9), the system calculates the partition entropy for each feature (or object). Finally, the calculated values are tabulated in a partition coefficient table or a partition entropy table.

(Step 3) Re-assignment of classification codes (or classes) to each unmatched instance based on the values of PC_j or PE_j. A classification code that has a maximum value of PC_j in the partition coefficient table is assigned. Additionally, a classification code that has a minimum value of PE_j is assigned.

(Step 4) Heuristic determination of the threshold for PC_j or PE_j for each iteration.

(Step 5) Re-assignment of classification codes (or classes) to each unmatched instance based on the values of PC_j in the partition coefficient table. First, the system removes the data for which the values of PC_j are larger than the threshold determined in Step 4 from the partition coefficient table. Then, it re-assigns a classification code (or class) using the remaining data in the partition coefficient table. Further, the system re-assigns a classification code (or class) based on the values of PE_j in the partition entropy table. The system removes the data for which the values of PE_j are smaller than the threshold determined in Step 4 from the partition entropy table. Then, it re-assigns a classification code using the remaining data in the partition entropy table.

(Step 6) Iteration of steps 4 and 5 until the re-assigned classification accuracy is below a heuristically determined threshold.

3. OVERLAPPING MULTICLASS CLASSIFIER METHOD

As the $\tilde{p}_{jk}$ shown in (7) is the probability of feature (or object) j to code (or class) k, we may consider the maximum value of $\tilde{p}_{jk}$ over K classes to select the assigned class for feature (or object) j. However, sometimes we have the following situation for the maximum value of $\tilde{p}_{jk}$:

$$\tilde{p}_{jk} \approx \tilde{p}_{jk'} \text{ for } \exists \text{ j}, \forall\, k, k' \in \{1, \ldots, K\}.$$

In this case, we cannot assign feature (or object) j to a "single" code (or class). Thus, we realize the limitation of exclusive classification in which a feature (or object) is assigned to a "single" code (or class). Therefore, it is required to select multiple codes (or classes) as the candidates for a feature (or object).

First, we define $S_j \equiv \{\tilde{p}_{j1}, \ldots, \tilde{p}_{jK}\}$ and arrange $\tilde{p}_{j1}, \ldots, \tilde{p}_{jK}$ in descending order so that $\tilde{\tilde{p}}_{j1} \geq \tilde{\tilde{p}}_{j2} \geq \cdots \geq \tilde{\tilde{p}}_{jK}$ holds. Additionally, we redefine $\tilde{S}_j \equiv \{\tilde{\tilde{p}}_{j1}, \ldots, \tilde{\tilde{p}}_{jK}\}$.

Then, we select the $\tilde{K}$ largest values from $\tilde{S}_j$, as follows:

$$\{\tilde{\tilde{p}}_{j1}, \ldots, \tilde{\tilde{p}}_{j\tilde{K}}\}, \qquad \tilde{K} \in \{2, \ldots K\},$$

where $\tilde{K}$ codes (or classes) are the selected codes (or classes) for the j-th feature.

Based on the idea of partition entropy in (9), we define a new reliability score $\bar{p}_{jk}$ as follows:

$$\bar{p}_{jk} = \tilde{\tilde{p}}_{jk}\left(1 + \sum_{m=1}^{\tilde{K}} \tilde{\tilde{p}}_{jm} \log_K \tilde{\tilde{p}}_{jm}\right). \tag{12}$$

From (7) and (12), $\bar{p}_{jk}$ satisfies the following condition when $\alpha = \beta = 0$:

$$0 \leq \bar{p}_{jk} \leq 1. \tag{13}$$

The reliability score in (12) considers not only the probability of feature (or object) j to code (or class) k, $\tilde{\tilde{p}}_{jk}$, but also the classification status of feature (or object) j over the largest $\tilde{K}$ codes (or classes), $1 + \sum_{m=1}^{\tilde{K}} \tilde{\tilde{p}}_{jm} \log_K \tilde{\tilde{p}}_{jm}$. If the probability of feature (or object) j to the code (or class) k is larger while the classification status of the feature (or object) j over the selected $\tilde{K}$ codes (or classes) is clearer, then both values of $\tilde{\tilde{p}}_{jk}$ and $1 + \sum_{m=1}^{\tilde{K}} \tilde{\tilde{p}}_{jm} \log_K \tilde{\tilde{p}}_{j}$ become larger. Then from the definition of (12), the value of reliability score, $\bar{p}_{jk}$, will also be larger. Otherwise, the score will be smaller.

The proposed system assigns a classification code (or class) based on the value of $\bar{p}_{jk}$ through the following procedures in the classification process shown in Fig. 1:

(Step 1) The system performs feature extraction and the retrieval of candidate codes (or classes) in the same manner as that of the previous study described in section 2.

(Step 2) The system calculates $\tilde{p}_{jk}$ for every retrieved candidate code (or class). Then, it determines the top $\tilde{K}$ promising candidate codes (or classes) for each feature based on $\tilde{p}_{jk}$. In this study, we heuristically set $\tilde{K} = 5$.

(Step 3) The system calculates the new reliability score $\bar{p}_{jk}$ whose features belong to an instance and lists codes (or classes) in descending order corresponding with the values of $\bar{p}_{jk}$.

(Step 4) Finally, for each feature, the system determines the top L (L = 1, 2, 3 …) codes (or classes) for output data.

4. EXPERIMENTS AND RESULTS

For the numerical example, we applied the proposed classifier to *the Family Income and Expenditure Survey* dataset. We have approximately 4.55 million instances (word sequences) for training and approximately 0.65 million instances for evaluation. Each instance contains an item name (a

transacted item name that is related to income or expenditure) in Japanese. For efficient classification, we separately performed the training and classification processes for income data and expenditure data.

Table 1 shows the classification accuracy of the proposed classifier. The classifier lists 5 candidate codes (or classes) with their reliability scores as output results. Let Q be the number of input instances and M_i be the number of matched instances at the i-th candidate code (or class). Then, we define "cumulative accuracy" as follows:

$$\text{Cumulative accuracy} = \frac{\sum_{i=1}^{T} M_i}{Q}.$$

We find that more than 90% of the target instances are assigned correct codes (or classes) as the 1st candidate code (or class). In addition, our classifier correctly assigns codes (or classes) for more than 97% of the target instances among the top 5 candidate codes (or classes).

Classification accuracy of the proposed classifier

Table 1

	Number of total instances	Number of matched instances	Number of cumulative matched instances	Cumulative accuracy
1st candidate	655,572	592,342	592,342	0.904
2nd candidate		30,275	622,617	0.950
3rd candidate		9,240	631,857	0.964
4th candidate		4,274	636,131	0.970
5th candidate		2,519	638,650	0.974

We prepared a mini dataset for the comparison of classification accuracy. We randomly extracted 11,000 instances of foodstuffs and dining-out data from *the Family Income and Expenditure Survey* dataset. Then, we assigned 11 new classification codes to the dataset for performance evaluation (see Table 2). Following this, we randomly divided the dataset into 10,000 instances for training and 1,000 instances for evaluation. We repeatedly performed this division task and prepared there different datasets. Each instance of these datasets contained an item name (a foodstuff name, a food product name, or an item on a restaurant menu) in Japanese.

Overview of the prepared mini dataset

Table 2

No.	Contents	Classification code	Number of instances in dataset 1	Number of instances in dataset 2	Number of instances in dataset 3
1	Cereals	A	1,018	1,007	1,049
2	Fish and shellfish	B	927	950	926
3	Meat	C	775	746	765
4	Dairy products and eggs	D	717	727	729
5	Vegetables and seaweed	E	2,966	2,954	2,913
6	Fruits	F	485	505	498
7	Oils, fats, and seasonings	G	661	713	686
8	Cakes and candies	H	1,026	1,025	1,048
9	Cooked food	I	1,221	1,211	1,270
10	Beverages, including alcoholic beverages	J	868	845	814
11	Meals outside the home	K	336	317	302

Table 3 shows the classification accuracy of the proposed classifier. Table 4 shows the classification accuracy of our original classifier (Toko et al., 2017) and a random forest model (Breiman, 2001). From Tables 3 and 4, we find that even though the classification accuracy of the first candidate of the proposed classifier is almost similar to our original classifier and the random forest model, the proposed classifier performs better when considering cumulative accuracy based on overlapping classification.

Classification accuracy of the proposed classifier (mini dataset)

Table 3

		Number of total instances	Number of matched instances	Number of cumulative matched instances	Cumulative accuracy
dataset 1	1st candidate	1,000	842	842	0.842
	2nd candidate		68	910	0.910
	3rd candidate		14	924	**0.924**
dataset 2	1st candidate		832	832	0.832
	2nd candidate		69	901	0.901
	3rd candidate		26	927	**0.927**
dataset 3	1st candidate		837	837	0.837
	2nd candidate		59	896	0.896
	3rd candidate		32	928	**0.928**

Classification accuracy of the original classifier and random forest (mini dataset)

Table 4

		Number of total instances	Number of matched instances	Accuracy
dataset 1	Original classifier	1,000	842	0.842
	Random forest		822	0.822
dataset 2	Original classifier		819	0.819
	Random forest		822	0.822
dataset 3	Original classifier		839	0.839
	Random forest		802	0.802

Fig. 2 shows the reliability score and $V_j^{(i)} \equiv 1 + \sum_{m=1}^{K} \tilde{\tilde{p}}_{jm} \log_K \tilde{\tilde{p}}_{jm}$, which show the classification status of each matched instance in dataset 1 when the *i*-th candidate code (or class) was assigned correctly. Note that each instance has a single *i*-th candidate code (or class) for features that belong to the instance. We find that more than half of the evaluated instances in dataset 1 are assigned correct codes (or classes) with a high reliability score (over 0.9). In addition, this figure shows the difference between the proposed reliability score and $V_j^{(i)}$. There is discrepancy of monotony between the reliability score and $V_j^{(i)}$. This implies that the proposed reliability score considering $\tilde{\tilde{p}}_{jk}$ addresses the issue that $V_j^{(i)}$ does not compare the superiority of each code (or class) for feature (or object) *j*.

Reliability score and $V_j^{(i)}$ of each matched instance in dataset 1

Fig. 2

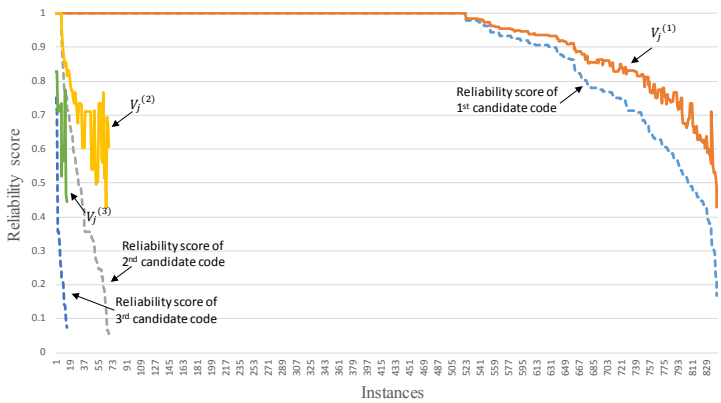
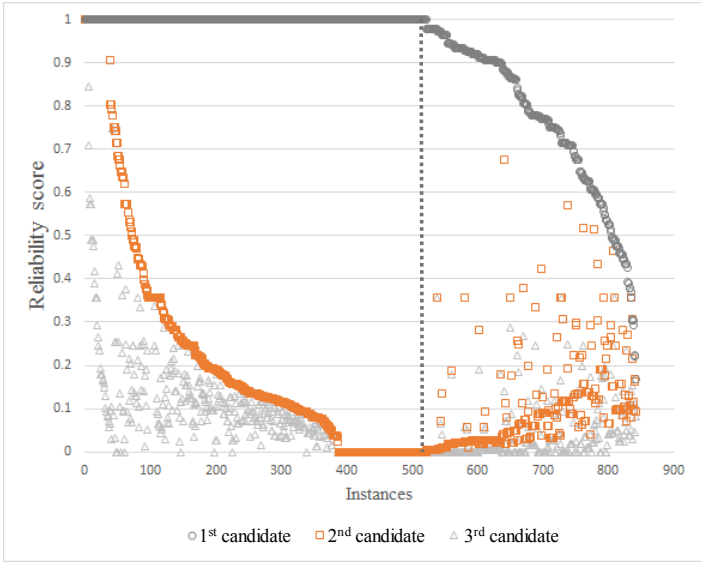


Fig. 3 shows the reliability scores of each candidate code (or class) plotted using the instances that match with the 1st candidate code (or class) in dataset 1. From the figure, we find that a certain volume of instances have large differences between the reliability score of the 1st candidate code (or class) and those of other candidates codes (or classes), whereas a few objects do not have large differences. This figure suggests the status of the clarity of the classification of each instance. In addition, when the reliability score for the 1st candidate code (or class) is large, the classification status is clear compared with the case when the reliability score for the 1st candidate code (or class) is small. Furthermore, Fig. 4 shows the reliability scores of each candidate code (or class) plotted using the instances that match with the 2nd candidate code (or class) in dataset 1. According to Fig. 3 and Fig. 4, it appears that the target instances that can be clearly classified are correctly assigned codes (or classes) at the 1st candidate. In such a case, the difference between the reliability scores of the 1st candidate code (or class) and those of other candidate codes (or classes) would be large. On the contrary, it appears that the target instances that cannot be clearly classified are assigned correct codes (or classes) at the 2nd candidate code (or class).

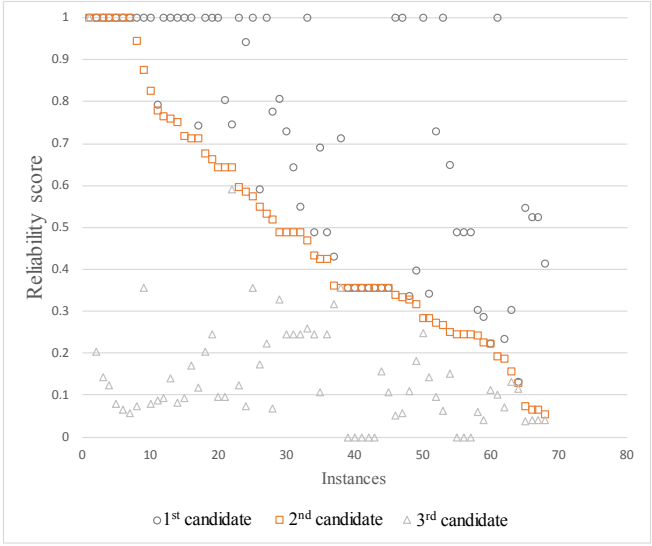
Reliability scores of instances that match with the 1st candidate code in dataset 1

Fig. 3



Reliability score of instances that match with the 2nd candidate code in dataset 1

Fig. 4



We also applied our classifier to the Stack Overflow dataset, which is a publicly available short description data set that was published by Jiaming Xu et al. (2015) on Kaggle. We used this dataset to evaluate the classifier with another language dataset. Each instance of this dataset consists of question titles in English and 20 different classification codes (see Table 5). This dataset contains 20,000 instances, and we randomly divided the dataset into 18,000 instances for training and 2,000 instances for evaluation. In addition, we performed the following pre-procedures before this experiment: convert all letters to lowercase and remove symbols that are clearly unnecessary for code assignment such as sentence ending symbols "?" and ".".

Overview of the Stack Overflow dataset

Table 5

No.	Contents	Classification code	Number of instances in the dataset	No.	Contents	Classification code	Number of instances in the dataset
1	wordpress	1	1,000	11	spring	11	1,000
2	oracle	2	1,000	12	hibernate	12	1,000
3	svn	3	1,000	13	scala	13	1,000
4	apache	4	1,000	14	sharepoint	14	1,000
5	excel	5	1,000	15	ajax	15	1,000
6	matlab	6	1,000	16	qt	16	1,000
7	visual-studio	7	1,000	17	drupal	17	1,000
8	cocoa	8	1,000	18	linq	18	1,000
9	osx	9	1,000	19	haskell	19	1,000
10	bash	10	1,000	20	magento	20	1,000

Table 6 shows the classification accuracy of our classifier. It was found that more than 85% of target instances were correctly assigned codes for each dataset. In addition, the cumulative accuracy for the 1st to 3rd candidates was approximately 94%. Even though our classifier has been developed for the classification of Japanese text descriptions, it could be applied to the classification of short English text descriptions.

Classification accuracy for the Stack Overview dataset

Table 6

		Number of total instances	Number of matched instances	Number of cumulative matched instances	Cumulative accuracy
dataset 1	1st candidate	2,000	1,739	1,739	0.870
	2nd candidate		104	1,843	0.922
	3rd candidate		42	1,885	**0.943**
dataset 2	1st candidate		1,736	1,736	0.868
	2nd candidate		123	1,859	0.930
	3rd candidate		25	1,884	**0.942**
dataset 3	1st candidate		1,707	1,707	0.854
	2nd candidate		130	1,837	0.919
	3rd candidate		41	1,878	**0.939**

5. CONCLUSION

This paper presents a new overlapping multiclass classifier for autocoding. To handle ambiguous text descriptions, we propose a new reliability score based on the weights of entropy. In addition, to address the

limitation of the exclusive classification of a feature (or object) to a code (or class), our proposed classifier lists multiple candidates of codes (or classes) with their reliability scores. This significantly assists manual coding.

The numerical examples show that our overlapping multiclass classifier assigns a classification code with high accuracy. The accuracy of our classifier is better than that of our previously developed classifier (non-overlapping classifier) and the random forest model when considering cumulative accuracy based on the idea of overlapping classification.

In future studies, determining a certain threshold to separate the instances that uniquely assigned classification codes (or classes) from other instances should be discussed. In addition, obtaining the optimal number of selected codes (or classes) considering the balance between accuracy and processing cost is an important problem. Furthermore, even though the proposed classifier has been developed in R, it has not been published as an R package. To contribute the efficient coding tasks, we are considering releasing our classifier as an R package after improving its usability.

Acknowledgements

We would like to thank Kaggle for making the Stack Overflow dataset available.

References

1. **Bezdek, J.C.**, 1981, *Pattern Recognition with Fuzzy Objective Function Algorithms*, Plenum Press, New York.
2. **Bezdek, J.C., Keller J., Krisnapuram, R., Pal, N.R.**, 1999, *Fuzzy Models and Algorithms for Pattern Recognition and Image Processing*, Kluwer Academic Publishers, New York.
3. **Breiman, L.**, 2001, *"Random forests"*, Machine Learning, Vol. 45, pp. 5-32.
4. **Gweon, H., Schonlau, M., Kaczmirek, L., Blohm, M., Steiner, S.**, 2017, *"Three methods for occupation coding based on statistical learning"*, Journal of Official Statistics, Vol. 33, No. 1, pp. 101-122. DOI: ***https://doi.org/10.1515/jos-2017-0006***.
5. **Hacking, W., Willenborg, L.**, 2012, *"Coding; interpreting short descriptions using a classification"*, Statistics Methods, Statistics Netherlands, The Hague, Netherlands, Available at: ***https://www.cbs.nl/en-gb/our-services/methods/statistical-methods/throughput/throughput/coding*** (accessed August 2018).
6. **Kudo, T., Yamamoto, K., Matsumoto, Y.**, 2004, *"Applying conditional random fields to Japanese morphological analysis"*, in the 2004 Conference on Empirical Methods in Natural Language Processing, Barcelona, Spain, 25-26, Jul. 2004, pp. 230-237.
7. **Shimono, T., Wada, K., Toko, Y.**, 2018, *"A supervised multiclass classifier using machine learning algorithm for autocoding"*, Research Memoir of Official Statistics, Vol. 75, pp. 41-60 (in Japanese).
8. **Toko, Y., Wada, K., Kawano, M.**, 2017, *"A supervised multiclass classifier for an autocoding system"*, Romanian Statistical Review, Vol. 4, pp. 29-39.

9. **Toko, Y., Wada, K., Iijima, S., Sato-Ilic, M.**, 2018, "Supervised multiclass classifier for autocoding based on partition coefficient", Czarnowski, I., Howlett, R.J., Jain, L. C., and Vlacic, L. (Eds.), Intelligent Decision Technologies 2018, Smart Innovation, Systems and Technologies, Springer, Switzerland, Vol. 97, pp. 54-64.
10. **Tsubaki, H., Wada, K., Toko, T**., 2017, *"An extension of Taguchi's T method and standardized misclassification rate for supervised classification with only binary inputs"*, presented in poster session in the ANQ Congress, 20-21, Sep. 2017, Kathmandu, Nepal.
11. **Xu, J., Wang, P., Tian, G., Xu, B., Zhao, J., Wang, F. and Hao, H**., 2015, "Short text clustering via convolutional neural networks" In Proceedings of NAACL-HLT. May 31-June 5, 2015. pp. 62-69. Denver, Colorado, USA.