Autocoding based Multi-Class Support Vector Machine by Fuzzy c-Means

Yukako Toko (ytoko@nstac.go.jp)
National Statistics Center, 19-1 Wakamatsu-cho, Shinjuku-ku, Tokyo 162-8668, Japan,

Mika Sato-Ilic (mika@risk.tsukuba.ac.jp)
University of Tsukuba, Tennodai 1-1-1, Tsukuba, Ibaraki 305-8573, Japan,

ABSTRACT

This paper proposes a new autocoding method for the coding task of the Family Income and Expenditure Survey. The data of the Family Income and Expenditure Survey included text descriptions extracted from digital receipts which have been getting large and complex in recent years. This paper proposes a new autocoding method to obtain stable results of discrimination as coding with high generalization performance dealing with cognitive uncertainty for text description data. This method is a combination of multi-class Support Vector Machine (SVM) by fuzzy c-means and the previously developed reliability score based classification method. The proposed method utilizes both SVM, a machine learning method known as high generalization performance, and fuzzy c-means that is a computational intelligence method known as high performance dealing with cognitive uncertainty. Also, the proposed method utilizes the previously developed classification method based on reliability score. A numerical example shows a better performance of the proposed method with the Family Income and Expenditure Survey compared with the previously proposed classification method. The proposed method is developed in python utilizing python libraries, and also it can be easily run in R, which is a popular language in the official statistics field.

Keywords: *Coding, Machine Learning, Word2Vec, Support Vector Machine, Fuzzy c-Means, Reliability Score*

JEL Classification: *C38*

1. Introduction

For the coding task on official statistics, though coding is originally performed manually, the studies of automated coding have made progress with the improvement of computer technology. For example, Hacking and Willenborg (2012) introduced coding methods, including autocoding. Gweon et al. (2017) illustrated methods for automated occupation coding based on statistical learning. For the coding task of *the Family Income and Expenditure Survey*, we have developed an autocoding method which is the reliability score

However, it is well known that the Bernoulli type simple Bayes model does not perform well for a large amount of complex data, whereas the data of the *Family Income and Expenditure Survey* data, included text descriptions that were extracted from receipts digitally, is getting large and complex. In order to obtain stable results of discrimination as coding with high generalization performance dealing with cognitive uncertainty for text description data, this paper proposes a new autocoding method which is a combined method of Support Vector Machine (SVM) (Cristianini and Shawe-Taylor, 2000) and fuzzy c-Means (Bezdek, 1981).

In our previous study (Toko and Sato-Ilic, 2021a), we developed a hybrid method of SVM utilizing Word2Vec (Mikolov et al., 2013), which is a method to produce word embeddings, and the previously developed classification method based on reliability score. However, as the previously proposed method simply applied SVM to a whole given data, there was room for more efficiently classifying of those data to improve the classification accuracy. Therefore, we have proposed a hybrid method of multi-class SVM and k-means based on reliability score (Toko and Sato-Ilic, 2021b). The method utilized k-means before applying SVM to capture significant features of data. Based on the captured features, SVM is applied individually to each data included in the corresponding cluster. The merit of obtaining several groups before applying SVM is that it allows applying SVM individually to each group considering each group's discriminable ability. In the previously method, we combined such a function with the previously proposed hybrid method of SVM utilizing Word2Vec and our previously developed classification method based on reliability score.

However, our recent data is obtained as receipts digitally, which include complex representations of text description. Since the k-means is one of the methods of hard clustering in which each data is classified to only one cluster, the k-means tends to need many clusters to obtain a better solution. In this case, the result has less solution robustness, and a large amount of computation is necessary for obtaining a better solution.

Therefore, the purpose of this paper is the inclusion of fuzzy clustering as fuzzy c-means in order to obtain stable results dealing with cognitive uncertainty for text description data. For this purpose, this paper proposes a new autocoding method which is a combined method of SVM and a reliability score based fuzzy c-means in computational intelligence, which is linguistically motivated computational paradigms, theory and design of fuzzy logic, neural networks, and evolutionary computation.

The rest of this paper is organized as follows: Fuzzy c-means method, Word2Vec, and SVM are explained in sections 2, 3, and 4. The method of autocoding based reliability score is described in section 5. The multi-class SVM by fuzzy c-means is proposed in section 6. The numerical examples are illustrated in section 7. Conclusions are described in section 8.

2. Fuzzy c-means

In fuzzy clustering, each object has a degree of belongingness to clusters which can range from any value from 0 to 1. In hard clustering, each object has only two values for the degree of belongingness which is 0 or 1. If an object belongs to a cluster, then the degree of belongingness of the object to the cluster is 1, otherwise it is 0. However, if the obtained data has complexity which causes boundary uncertainty situation of the belongingness to the certain number of clusters, then without increasing the number of clusters, it will be difficult to explain the complex classification situation of the data. Therefore, fuzzy clustering is useful to explain the complex data into the clustering.

In this study, we utilize fuzzy c-means (Bezdek, 1981), which is one fuzzy clustering method. The purpose of fuzzy c-means is to obtain U and V which minimize the following objective function:

$$J(U,V) = \sum_{k=1}^{K}\sum_{i=1}^{n}(u_{ik})^m \|\boldsymbol{x}_i - \boldsymbol{v}_k\|^2, (1)$$

where, $\boldsymbol{x}_i = (x_{i1}, \dots, x_{ip})$ is a vector of i-th object, $V = (v_{ka})$ shows a matrix of cluster centers consisted of a vector $\boldsymbol{v}_k = (v_{k1}, \dots, v_{kp})$ which is a center of cluster k, $U = (u_{ik})$ is a matrix of clustering results where u_{ik} is a degree of belongingness of i-th object to a cluster k which satisfy the conditions $u_{ik} \in [0,1], \sum_{k=1}^{K} u_{ik} = 1$. m is a parameter which controls fuzziness of the fuzzy clustering result and satisfies $m \in (1, \infty)$. K is a number of clusters, n is a number of objects, and p is a number of variables. Local optimum solutions which minimizes (1) are obtained as follows:

$$u_{ik} = \left[\sum_{j=1}^{K}\left(\frac{\|\boldsymbol{x}_i - \boldsymbol{v}_k\|}{\|\boldsymbol{x}_i - \boldsymbol{v}_j\|}\right)^{\frac{2}{m-1}}\right]^{-1}. \quad (2)$$

$$v_k = \frac{\sum_{i=1}^{n}(u_{ik})^m \boldsymbol{x}_i}{\sum_{i=1}^{n}(u_{ik})^m}. \quad (3)$$

Using (2) and (3), the algorithm of fuzzy c-means is shown as follows:

Step1. Initialize the degree of belongingness of objects to clusters

Step2. Calculate the cluster centers by using (3)

Step 3. Update the degree of belongingness of objects to clusters by using (2)

Step 4. Stop if the difference of the degree of belongingness of objects to clusters and the degree calculated in the previous iteration is smaller than ε; otherwise, iterate steps 2 and 3

3. Word2Vec

Word2Vec was developed based on an idea of a neural probabilistic language model in which words are embedded to a continuous space by using distributed representations of the words (Mikolov et al., 2013). The algorithm of Word2Vec learns word association from a given dataset utilizing a neural network model based on an idea of a neural probabilistic

given dataset utilizing a neural network model based on an idea of a neural probabilistic language model (Bengio et al., 2003). It produces a vector space and each word in the given dataset is assigned a corresponding numerical vector of a word in the produced vector space. The essence of the idea is to avoid the curse of dimensionality by distributed representations of words.

Word2Vec utilizes continuous bag-of-words (CBOW) model and continuous skip-gram model (Mikolov et al., 2013) to distributed representations of words. The CBOW model predicts the current word based on the context. The skip-gram model uses each current word to predict words within a certain range before and after the current word. It gives less weight to the distant context words. In this study, the skip-gram model is applied.

4. Support Vector Machine

SVM (Cristianini and Shawe-Taylor, 2000) is a supervised machine learning algorithm for classification.

If $\boldsymbol{w}$ is the weight vector realizing a functional margin of 1 on the positive point $\boldsymbol{x}^+$ and negative point $\boldsymbol{x}^-$, a functional margin of 1 implies

$$\langle \boldsymbol{w} \cdot \boldsymbol{x}^+ \rangle + b = +1,$$
$$\langle \boldsymbol{w} \cdot \boldsymbol{x}^- \rangle + b = -1,$$

while $\boldsymbol{w}$ is normalized. Then the margin γ is the functional margin of the resulting classifier

$$\gamma = \frac{1}{2}\left(\langle \frac{\boldsymbol{w}}{\|\boldsymbol{w}\|_2} \cdot \boldsymbol{x}^+ \rangle - \langle \frac{\boldsymbol{w}}{\|\boldsymbol{w}\|_2} \cdot \boldsymbol{x}^- \rangle\right)$$
$$= \frac{1}{2\|\boldsymbol{w}\|_2}(\langle \boldsymbol{w} \cdot \boldsymbol{x}^+ \rangle - \langle \boldsymbol{w} \cdot \boldsymbol{x}^- \rangle) = \frac{1}{\|\boldsymbol{w}\|_2}.$$

Therefore, the resulting margin will be equal to $1/\|\boldsymbol{w}\|_2$ and the following can be written. Given a linearly separable training sample

$$S = \left((\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \dots, (\boldsymbol{x}_l, y_l)\right)$$

the hyperplane $(\boldsymbol{w}, b)$ that solves the optimization problem

$$\min_{\boldsymbol{w},b} \langle \boldsymbol{w} \cdot \boldsymbol{w} \rangle, \tag{4}$$

$$\text{subject to } y_i(\langle \boldsymbol{w} \cdot \boldsymbol{x}_i \rangle + b) \geq 1, i = 1, \dots, l,$$

realizes the maximal margin hyperplane with geometric margin $\mathrm{M} = 1/\|\boldsymbol{w}\|_2$. Then, slack variables are introduced to allow the margin constraints to be violated

$$\text{subject to } y_i(\langle \boldsymbol{w} \cdot \boldsymbol{x}_i \rangle + b) \geq 1 - \xi_i, i = 1, \dots, l,$$
$$\xi_i \geq 0, i = 1, \dots, l.$$

From the above, the optimization problem shown in (4) can be written as

$$\min_{\xi,w,b}\langle \boldsymbol{w}\cdot\boldsymbol{w}\rangle + C\sum_{i=1}^{l}\xi_i, \tag{5}$$

$$\text{subject to } y_i(\langle \boldsymbol{w}\cdot\boldsymbol{x}_i\rangle + b) \geq 1-\xi_i, i=1,\dots,l,$$

$$\xi_i \geq 0, i=1,\dots,l.$$

where C is the cost parameter that will give the optimal bound as it corresponds to finding the minimum of $\|\xi\|_1$ in (5) with the given value for $\|\boldsymbol{w}\|_2$. This is soft-margin linear SVM.

Also, SVM performs a non-linear classification transforming input data into higher dimensional spaces and calculating the inner product between the data in higher dimensional space using kernel trick. SVM uses kernel functions to enable it to obtain the inner product of data in higher dimensional space (kernel trick), which is represented as follows:

$$k(\boldsymbol{x}_i,\boldsymbol{x}_j) = \varphi(\boldsymbol{x}_i)^T\varphi(\boldsymbol{x}_j),$$

where φ is a mapping function from an observational space to a higher-dimensional space. The conditions for $k(\boldsymbol{x},\boldsymbol{x}')$ to be a kernel function are as follows:

- Symmetry: $k(\boldsymbol{x},\boldsymbol{x}') = k(\boldsymbol{x}',\boldsymbol{x})$.
- Gram matrix is Positive semi-definite.

As there are many possible choices for the kernel function, the radial basis function is applied in this paper.

- Radial basis function kernel

$$k(\boldsymbol{x},\boldsymbol{x}') = \exp(-\gamma\|\boldsymbol{x}-\boldsymbol{x}'\|^2), \left(\gamma = \frac{1}{2\sigma^2}\right), \tag{6}$$

The mapped feature space of the radial basis kernel function has an infinite number of dimensions.

For multi-class SVM, there are two approaches: one-versus-the-rest and one-versus-one. In one-versus-the-rest, SVM builds binary classifiers that discriminate between one class and the rest, whereas it builds binary classifiers that discriminate between every pair of classes in one-versus-one.

5. Classification method based on reliability score

The classification method based on reliability score performs the extraction on objects and retrieval of candidate classes from the object frequency table provided by using the extracted objects. Then, it calculates the relative frequency of j -th object to a code k defined as

$$p_{jk} = \frac{n_{jk}}{n_j}, n_j = \sum_{k=1}^{K} n_{jk}, j=1,\dots,J, k=1,\dots,K,$$

where n_{jk} is the number of occurrence of statuses in which an object *j* assigned to a code k in the training dataset. J is the number of objects and K is the number of codes.

However, this classifier has difficulty correctly assigning codes to text descriptions for complex data included uncertainty. To address the problem, we developed the overlapping classifier that assigns codes to each text description based on the reliability score (Toko et al., 2018a, Toko et al., 2018b, Toko et al., 2019, Toko and Sato-Ilic, 2020). Then, the classifier arranges $\{p_{j1}, \cdots, p_{jK}\}$ in descending order and creates $\{\tilde{p}_{j1}, \cdots, \tilde{p}_{jK}\}$, such as $\tilde{p}_{j1} \geq \cdots \geq \tilde{p}_{jK}, j = 1, \cdots, J$. After that, $\left\{\tilde{\tilde{p}}_{j1}, \cdots, \tilde{\tilde{p}}_{j\tilde{K}_j}\right\}, \tilde{K}_j \leq K$ are created. That is, each object has a different number of codes. Then, the classifier calculates the reliability score for each code of each object. The reliability score of j-th object to a code k is defined as

$$\bar{p}_{jk} = T\left(\tilde{\tilde{p}}_{jk}, 1 + \sum_{m=1}^{\tilde{K}_j} \tilde{\tilde{p}}_{jm} \log_K \tilde{\tilde{p}}_{jm}\right), \qquad j = 1, \ldots, J, k = 1, \ldots, \tilde{K}_j.$$

$$\bar{p}_{jk} = T\left(\tilde{\tilde{p}}_{jk}, \sum_{m=1}^{\tilde{K}_j} \tilde{\tilde{p}}_{jm}^2\right), \quad j = 1, \ldots, J, k = 1, \ldots, \tilde{K}_j.$$

These reliability scores were defined considering both probability measure and fuzzy measure (Bezdek, 1981, Bezdek et al., 1999). That is, $\tilde{\tilde{p}}_{jk}$ shows the uncertainty from the training dataset (probability measure) and $1 + \sum_{m=1}^{\tilde{K}_j} \tilde{\tilde{p}}_{jm} \log_K \tilde{\tilde{p}}_{jm}$ or $\sum_{m=1}^{\tilde{K}_j} \tilde{\tilde{p}}_{jm}^2$ shows the uncertainty from the latent classification structure in data (fuzzy measure). These values of the uncertainty from the latent classification structure can show the classification status of each object; that is, how each object is classified to the candidate codes. *T* shows *T*-norm in statistical metric space (Menger, 1942, Mizumoto, 1989, Schweizer and Sklar, 2005). We generalize the reliability score by using the idea of *T*-norm, which is a binary operator in statistical metric space. Furthermore, to prevent an infrequent object having significant influence, sigmoid functions $g(n_j)$ were introduced to the reliability score. The reliability score considering the frequency of each object over the codes for each object in the training dataset as follows (Toko et al., 2019, Toko and Sato-Ilic, 2020):

$$\bar{\bar{p}}_{jk} = g(n_j) \times \bar{p}_{jk}.$$

In this study, algebraic product is taken as *T*-norm and $n_j/\sqrt{1 + n_j{}^2}$ is taken as a sigmoid function for the reliability score.

6. Autocoding method of SVM and fuzzy c-means method with the reliability score

The proposed autocoding method is a combined method of multi-class SVM by fuzzy c-means and classification method based on reliability score.

First, the proposed method tokenizes each text description by Sudachi (Takaoka et al., 2018). Then, it obtains numerical vectors corresponding words utilizing Word2Vec and normalizes each feature of the obtained set of numerical vectors. After that, it produces

sentence vectors for each text description based on the normalized vectors. Then, it applies fuzzy c-means to sentence vectors to classify them into several clusters. After applying fuzzy c-means, the proposed method assigns corresponding codes applying SVM for each dataset of each cluster. After that, it extracts dataset that assigned codes with low classification accuracy by SVM. Then, the proposed method applies fuzzy c-means and SVM to the extracted dataset. In fact, it iteratively applies fuzzy c-means and SVM to sub-clusters that has low classification accuracy at previous iteration until obtaining sufficiently better result. After code assignment by the combined method of fuzzy c-means and SVM, it extracts unmatched data and re-assign corresponding codes to the extracted data by the reliability score based classifier.

The detailed algorithm of the proposed method is the following:

Step 1. The proposed algorithm tokenizes each text description into words by Sudachi.

Step 2. It obtains numerical vectors corresponding to words utilizing Word2Vec: First, it produces a dataset concatenating all tokenized words consecutively. Then, it trains Word2Vec model using the produced dataset. Each unique word in the dataset will be assigned a corresponding numerical vector. The following are determined by trial and error:

- Type of model architecture: CBOW model or skip-gram model
- The number of vector dimensions
- The number of training iterations
- Window size of words considered by the algorithm

Step 3. It normalizes each feature of the obtained set of numerical vectors.

Step 4. It produces sentence vectors for each text description based on the normalized vectors at Step 3: First, it obtains a corresponding numerical vector for each word in each text description from the set of normalized numerical vectors. Then, it calculates the sum of obtained numerical vectors for each text description as sentence vectors.

Step 5. It applies the fuzzy c-means method: First fuzzy c-means method is applied to sentence vectors produced in step 4 to classify them into K clusters. For implementing the fuzzy c-means method, we determine the following by trial and error:

- Number of clusters K
- Error rate appeared in sect. 2 as ε
- m parameter appeared in sect. 2 as m
- Maximum number of iterations allowed

Step 6. It assigns corresponding codes applying SVM: It trains a Support Vector Machine and predicts a corresponding code for each target text description. For training a Support Vector Machine, we determine the following:

- Cost parameter appeared in (5) as C
- Kernel function to be applied
- Gamma parameter appeared in (6) as γ if radial basis function kernel is applied as a kernel function

- Type of methods: one-versus-the-rest or one-versus-one

In this study, a radial basis function as a kernel function is applied. We apply the one-versus-one method. Cost parameter C and gamma parameter γ are determined by grid search.

Step 7. It extracts datasets that assigned codes with low classification accuracy in step 6. The dataset that assigned codes with high classification accuracy in step 6 are accepted as classification results.

Step 8. It performs step 5 thorough step 7 iteratively to the extracted dataset in step 7 until obtaining sufficiently better result.

Step 9. It extracts unmatched data.

Step 10. It implements re-classification based on reliability score to the extracted unmatched data.

7. Numerical example

For the numerical example, the proposed method is applied to the *Family Income and Expenditure Survey* dataset. The *Family Income and Expenditure Survey* is a sampling survey related to a household's income and expenditure conducted by the Statistics Bureau Japan. This survey dataset includes short text descriptions related to a household's daily income and expenditure (receipt items name and purchase items name in Japanese) and their corresponding codes. In this numerical example, the target data is only data related to household expenditure in the dataset. The total number of codes related to household expenditure is around 520. Approximately 810 thousand text descriptions were used for this numerical example.

The proposed method was developed in python, applying the following python libraries: For training the Word2Vec model, we used "gensim" (Rehurek and Sojka, 2010). We selected the skip-gram model as a type of model architecture and set the number of vector dimensions as 100, the number of training iterations as 10,000, and the window size as 2. For implementing fuzzy c-means clustering, we used "skfuzzy" (Warner et al., 2019, scikit-fuzzy development team). We set the number of clusters as 2, m parameter appeared in sect. 2 as 1.1, error rate ε appeared in sect. 2 as 0.0001, and the maximum number of iterations as 10,000. For normalization of each feature of the set of numerical vectors and training support vector machines, we used "scikit-learn" (Pedregosa et al., 2011). We applied radial basis function kernel as the kernel function and selected the cost parameter C appeared in (5) and the gamma parameter γ appeared in (6) by grid search.

Table 1 shows the classification accuracy of SVM for each dataset obtained from the result of fuzzy c-means at the first iteration. The classification accuracy of dataset in C1 cluster is 0.858, whereas the classification accuracy of the dataset in C2 cluster is 0.917. Then, we accept the classification result of C2 cluster as classification result, and implement the second iteration to data in C1 cluster that has lower classification accuracy. Table 2 shows

classification accuracy of each cluster in the second iteration. The classification accuracy of C1_1 cluster is 0.788, whereas the classification accuracy of C1_2 cluster is 0.898, which is higher than the previously obtained score, 0.858. This means that we improve the accuracy of the data in C1 cluster partially. However, the remained data in C1, which is data in C1_1 is still a lower score at 0.788. Therefore, our next target to be improved accuracy is the data in C1_1. Then, we implement the third iteration to data in C1_1 cluster. Table 3 shows the classification accuracy of each cluster in the third iteration. The classification accuracy of C1_1_1 cluster is 0.837, whereas the classification accuracy of C1_1_2 cluster is 0.764. Again, we could obtain a better score which is 0.837 compared with 0.788. For the remained data, which has lower accuracy, has been implemented for further iteration to obtain a better accuracy. Table 4 shows successfully obtained the better score, which is 0.905 for the classification accuracy of the fifth iteration for the data in C1_1_2_1.

Table 1. First iteration: classification accuracy of each cluster

Cluster label	Number of text descriptions				Accuracy	SVM	
	Total	Train	Test	Correctly assigned		Cost	Gamma
C1	566,355	509,719	56,636	48,596	0.858	10	0.0012
C2	248,136	223,322	24,814	22,762	0.917	100	0.001
Total	814,491	733,041	81,450	71,358	0.876		

Table 2. Second iteration: classification accuracy of each cluster

Cluster label	Number of text descriptions				Accuracy	SVM	
	Total	Train	Test	Correctly assigned		Cost	Gamma
C1_1	249,261	224,334	24,927	19,635	0.788	10	0.0012
C1_2	317,094	285,384	31,710	28,488	0.898	10	0.0012
Total	566,355	509,718	56,637	48,123	0.850		

Table 3. Third iteration: classification accuracy of each cluster

Cluster label	Number of text descriptions				Accuracy	SVM	
	Total	Train	Test	Correctly assigned		Cost	Gamma
C1_1_1	88,614	79,752	8,862	7,420	0.837	10	0.0012
C1_1_2	160,647	144,582	16,065	12,275	0.764	10	0.0012
Total	249,261	224,334	24,927	19,695	0.790		

Table 4. Fourth iteration: classification accuracy of each cluster

Cluster label	Number of text descriptions				Accuracy	SVM	
	Total	Train	Test	Correctly assigned		Cost	Gamma
C1_1_2_1	61,398	55,258	6,140	5,556	0.905	10	0.0012
C1_1_2_2	99,249	89,324	9,925	6,700	0.675	10	0.0012
Total	160,647	144,582	16,065	12,256	0.763		

Table 5 shows the summary of classification accuracy of the proposed multi-class SVM by fuzzy c-means method. The total classification accuracy of the proposed method is 0.871 finally.

Table 5. Summary of classification accuracy of the proposed multi-class SVM by fuzzy c-means

Cluster label	Number of text descriptions				Accuracy
	Total	Train	Test	Correctly assigned	
C2	248,136	223,322	24,814	22,762	0.917
C1_2	317,094	285,384	31,710	28,488	0.898
C1_1_1	88,614	79,752	8,862	7,420	0.837
C1_1_2_1	61,398	55,258	6,140	5,556	0.905
C1_1_2_2	99,249	89,324	9,925	6,700	0.675
Total	814,491	733,040	81,451	70,926	0.871

In addition, table 6 compares the classification accuracy of the proposed method that is a combined method of multi-class SVM by fuzzy c-means method and the reliability score and the previously proposed method that is a combined method of multi-class SVM by k-means method and the reliability score (Toko and Sato-Ilic, 2021b). When combined multi-class SVM by fuzzy c-means and the reliability score, the classification accuracy is 0.922. This is better than the classification accuracy of the previously proposed method, as almost 250 data was increased for the correctly assigned. Although the number of increased text descriptions is not so large, the classification accuracy of the previously proposed method has already over 0.9. This means, therefore, text descriptions which easily classified to codes have already been assigned, and only difficult text descriptions to be classified have remained. Considering this matter, successfully classifying such difficult 250 text descriptions to the correct codes shows better results by the use of fuzzy clustering. Note that, for tokenizing text descriptions, the previously proposed method applies MeCab (Kudo et al., 2004), whereas this study applies Sudachi (Takaoka et al., 2018).

Table 6. Comparison of classification accuracy of the proposed method and the previously proposed method

Classification method	Accuracy
Combined method of multi-class SVM by **fuzzy c-means method** and the reliability score (Propsed method)	0.922
Combined method of multi-class SVM by **k-means method** and the reliability score (Previously propsed method)	0.919

Furthermore, when comparing the classification accuracy of multi-class SVM by fuzzy c-means and the classification accuracy of multi-class SVM by k-means, k-means in the previously proposed method requires many clusters to obtain a better result. Table 7 shows the classification accuracy of multi-class SVM by k-means. When comparing table 1 and table 7, the classification accuracy of the multi-class SVM by fuzzy c-means method is 0.876 with the number of clusters as 2, whereas the classification accuracy of the multi-class SVM

by k-means is 0.862 with the number of clusters as 10. From this comparison, it can be seen that SVM by fuzzy c-means allows us to reduce the number of clusters while retaining higher classification accuracy when compere with the multi-class SVM by k-means. In other words, when we utilize fuzzy c-means instead of k-means, data in each cluster tend to remain homogeneously, and the effectiveness of utilizing the cognitive uncertainty for text description data by fuzzy c-means can be seen in this result.

Table 7. Classification accuracy of SVM by k-means (previously proposed method)

Cluster	Number of text descriptions				Accuracy	SVM	
	Total	Training	Evaluation	Correctly assigned		cost	gamma
Cluster 1	4,568	4,111	457	375	0.821	30	0.0001
Cluster 2	37,454	33,708	3,746	3,719	0.993	100	0.0010
Cluster 3	137,157	123,441	13,716	12,068	0.880	30	0.0010
Cluster 4	148,585	133,726	14,859	12,909	0.869	10	0.0064
Cluster 5	38,003	34,202	3,801	3,082	0.811	10	0.0010
Cluster 6	31,288	28,159	3,129	3,116	0.996	100	0.0010
Cluster 7	275,852	248,266	27,586	22,929	0.831	90	0.0255
Cluster 8	47,421	42,678	4,743	4,243	0.895	90	0.0001
Cluster 9	48,332	43,498	4,834	4,093	0.847	100	0.0010
Cluster 10	45,831	41,247	4,584	3,672	0.801	10	0.0010
Total	814,491	733,036	81,455	70,206	0.862		

8. Conclusion

This paper proposes a new autocoding method which is a combined method of multi-class SVM by fuzzy c-means and the previously developed classification method based on reliability score to obtain stable result of discrimination as coding with high generalization performance dealing with cognitive uncertainty for text description data. SVM, a machine learning method known as high generalization performance, is utilized for classification based on the numerical vectors obtained by Word2Vec. Fuzzy c-means, a computational intelligence method known as high performance dealing with cognitive uncertainty with linguistically motivated computation, is utilized as a fuzzy clustering method to capture significant features of data before applying SVM. In addition, the previously developed classification method based on reliability score is applied to improve classification accuracy. The numerical example shows a better performance of the proposed method with *the Family Income and Expenditure Survey* data. From the result of the numerical example, it seems that data in each cluster tend to remain homogeneously, and the effectiveness of utilizing the cognitive uncertainty for text description data by fuzzy c-means. The proposed method is developed in python utilizing python libraries. Also, it can be run in R that is a popular language in the official statistics field utilizing the “reticulate” package (Ushey et al., 2021), which provides a comprehensive set of tools for interoperability between R and python. As

the “reticulate” package provides the “source_python( )” function that enable us to source a python script as we would source an R script, all functions in the python script become directly available to the R session after sourcing the script. Therefore, all functions and objects in our developed python script can be easily called from R utilizing the “reticulate” package. In future work, we would like to consider making the code open source in the framework in R.

References

1. Bengio, Y., Ducharme, R., Vincent, P., Jauvin, C. (2003) “A neural probabilistic language model”, Journal of Machine Learning Research, 3, pp. 1137-1155.
2. Bezdek, J.C. (1981), Pattern recognition with fuzzy objective function algorithms, Plenum Press.
3. Bezdek, J.C., Keller J., Krisnapuram, R., Pal, N.R. (1999), Fuzzy models and algorithms for pattern recognition and image processing, Kluwer Academic Publishers.
4. Cristianini, N., Shawe-Taylor, J. (2000), An Introduction to Support Vector Machines and other kernel-based learning methods, Cambridge University Press.
5. 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.
6. 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 December 2020).
7. 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, pp. 230-237.
8. Menger, K. (1942), “Statistical metrics”, in Proceedings of the National Academy of Sciences of the United States of America, Vol. 28, pp. 535-537.
9. Mikolov, T., Chen, K., Corrado, G., Dean, J. (2013), “Efficient estimation of word representations in vector space”, arXiv preprint arXiv:1301.3781.
10. Mizumoto, M. (1989), “Pictorical representation of fuzzy connectives, Part I: Cases of T-norms, t-Conorms and Averaging Operators”, Fuzzy Sets and Systems, Vol. 31, pp. 217-242.
11. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E. (2011), “Scikit-learn: Machine Learning in Python”, JMLR 12, pp. 2825-2830.

12. Rehurek, R., Sojka, P. (2010), “Software Framework for Topic Modelling with Large Corpora”, in Proceedings of LREC 2010 Workshop on New Challenges for NLP Frameworks. pp. 45-50.
13. Schweizer, S., Sklar, A. (2005), Probabilistic metric spaces, Dover Publications.
14. Takaoka, K., Hisamoto, S., Kawahara, N., Sakamoto, M., Uchida, Y., Matsumoto, Y. (2019), “Sudachi: a Japanese Tokenizer for Business”, in Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC), May 2018, Miyazaki, Japan, pp. 2246-2249, European Language Resources Association.
15. Toko, Y., Wada, K., Iijima, S., Sato-Ilic, M., (2018a), “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, Vol. 97, pp. 54-64.
16. Toko, Y., Iijima, S., Sato-Ilic, M. (2018b), “Overlapping classification for autocoding system”, Journal of Romanian Statistical Review, Vol. 4, pp. 58-73.
17. Toko, Y., Iijima, S., Sato-Ilic, M. (2019), “Generalization for improvement of the reliability score for autocoding”, Journal of Romanian Statistical Review, Vol. 3, pp. 47–59.
18. Toko, Y., Sato-Ilic, M., (2020), “Improvement of the training dataset for supervised multiclass classification”, Czarnowski, I., Howlett, R.J., Jain, L. C. (Eds.), Intelligent Decision Technologies, Smart Innovation, Systems and Technologies, Springer, Singapore, Vol. 193, pp. 291-302.
19. Toko, Y., Sato-Ilic, M., (2021a),”Efficient Autocoding Method in High Dimensional Space”, Romanian Statistical Review, Vol. 1, pp. 3-16.
20. Toko, Y., Sato-Ilic, M., (2021b), “A Hybrid Method of Multi-Class SVM and Classification Method Based on Reliability Score for Autocoding of the Family Income and Expenditure Survey”, Czarnowski, I., Howlett, R.J., Jain, L. C. (Eds.), Smart Innovation, Systems and Technologies, Vol. 238, pp. 403-414. Springer, Singapore.
21. Ushey, K., Allaire JJ, Tang Y. (2021), reticulate: Interface to ‘python’, R package version 1.22, https://CRAN.R-project.org/package=reticulate.
22. Warner, J., Sexauer, J., scikit-fuzzy, twmeggs, alexsavio, Unnikrishnan, A., et al. (2019) JDWarner/scikit-fuzzy: Scikit-Fuzzy version 0.4.2, https://doi.org/10.5281/zenodo.3541386.
23. scikit-fuzzy development team, “skfuzzy” Available at: https://pythonhosted.org/scikit-fuzzy (accessed November 2021)