Black-Box Classification Techniques for Demographic Sequences : from Customised SVM to RNN

Muratova, Anna and Sushko, Pavel and Espy, Thomas H.

National Research University Higher School of Economics, Moscow , Russia, University of Pittsburgh, USA

17 September 2017

Black-Box Classification Techniques for Demographic Sequences: from Customised SVM to RNN

Anna Muratova[1], Pavel Sushko[2], Thomas H. Espy[3]

[1] National Research University Higher School of Economics, Moscow, Russia
amuratova@hse.ru
[2] Institute of Sociology of the Russian Academy of Sciences, Moscow, Russia
sushkope@mail.ru
[3] University of Pittsburgh, USA
the7@pitt.edu

Abstract. Nowadays there is a large amount of demographic data which should be analysed and interpreted. From accumulated demographic data, more useful information can be extracted by applying modern methods of data mining. The aim of this study is to compare the methods of classification of demographic data by customising the SVM kernels using various similarity measures. Since demographers are interested in sequences without discontinuity, formulas for such sequences similarity measures were derived. Then they were used as kernels in the SVM method, which is the novelty of this study. Recurrent neural network algorithms, such as SimpleRNN, GRU and LSTM, are also compared. The best classification result with SVM method is obtained using a special kernel function in SVM by transforming sequences into features, but recurrent neural network outperforms SVM.

Keywords: data mining, demographics, support vector machines, neural networks, classification, sequences similarity.

1 Introduction

Nowadays researchers from different countries have access to a large amount of demographic data about important demographic events and their sequences. More useful information from accumulated demographic data can be extracted by applying modern methods of data mining.

The main task of this study is to find the most accurate classification method for analysing demographic sequences. For classification, various methods such as: decision trees, support vector machines (SVM), k nearest-neighbours (kNN), neural networks and others are used. This paper is a continuation of [9], in which decision trees, kNN and SVM were compared. The purpose of this paper is to compare methods for classifying demographic data by customising the SVM kernel using various similarity measures for sequences of events. Neural network algorithms are also compared. Alternative treatment of the problem by means of Pattern Mining [15], Formal Concept Analysis [12] and Pattern Structures [10,11], in particular, is given in [13,14].

Data were obtained from the scientific laboratory of socio-demographic policy at HSE and contain results of a survey of 6,626 people, including 3,314 men and 3,312 women. In the database, the dates of significant events in respondents' lives are indicated, such as partner, marriage, break up, divorce, education, work, separation from parents and birth of a child. Also, there are features of people: type of education (general, higher, professional), location (city, town, country), religion, frequency of church attendance, generation (Soviet, 1930-1969; modern, 1970-1986) and gender.

Chapter 2 presents a brief theoretical framework on sequence similarity measures, namely "the longest common subsequence" LCS and "all common subsequences" ACS. Chapter 3 presents the results of the work on the classification of demographic data by sequences of events without discontinuities. In Section 3.1, the special core variants are used in the SVM method (Support Vector Machines), and in Section 3.2 the results of recurrent neural networks (SimpleRNN, LSTM, GRU) are presented. Section 4 is devoted to comparing different classification methods and Section 5 presents the conclusions of the work.

The novelty of this work lies in the use of special kernel variants in the SVM method. In addition, the results are improved with the help of recurrent neural network algorithms.

2 Sequence similarity measures

Sequence analysis is an important task in data analysis and machine learning [1-4]. Pairwise relations between sequences are often used for them. For example, methods such as clustering and kernels depend on the calculation of distances and similarity measures between sequences. When calculating measures of similarity, it is necessary to take into account complex combinatorial aspects, since the sequences look like ordered sets of objects. Below we will consider measures of sequence similarity from objects on the basis of common subsequences contained in them.

The measure of similarity between two sequences S and T "all common subsequences" (ACS) [3] is defined as

$$sim_{ACS}(S,T) = \frac{\phi(S,T)}{max\{\phi(S),\phi(T)\}} \quad (1)$$

The measure of similarity "longest common subsequence" (LCS) is calculated by the formula

$$sim_{LCS_{size}}(S,T) = \frac{|LCS(S,T)|}{max\{|S|,|T|\}} \quad (2)$$

Demographers are interested in sequences without discontinuity (gaps). We are interested in two options: common prefixes and common subsequences without discontinuity. Let us transform the original formulas.

First consider the prefixes. In this case, the number of common prefixes of two sequences is equal to the length of the largest prefix of these sequences. Prefixes of

length zero are not considered. The number of prefixes of the sequence S is equal to the length of the sequence $|S|$.

Thus, the formulas for all common prefixes and for the longest common prefix of two sequences are the same and equal

$$sim_{ACSP}(S,T) = sim_{LCSP}(S,T) = sim_{CP}(S,T) = \frac{|LCSP(S,T)|}{max\{|S|,|T|\}} \tag{3}$$

where LCSP is the longest common sequence prefix, ACSP is the set of all common prefixes of sequences S and T and CP is the set of common prefixes.

Now consider subsequences without discontinuities. First consider the case of the longest common subsequence. Like the previous case, we get:

$$sim_{LCS}(S,T) = \frac{|LCS(S,T)|}{max\{|S|,|T|\}} \tag{4}$$

where LCS is the longest common subsequence of S and T without discontinuities.

Now let us look at all common subsequences without discontinuities. For this we consider all common subsequences S and T of different lengths without discontinuities. The number of subsequences of the sequence S without discontinuities is

$$\frac{|S|(|S|+1)}{2} \tag{5}$$

Since a longer sequence has more subsequences, we obtain the formula:

$$sim_{ACS}(S,T) = \frac{2 \cdot \sum_{k \le l} \phi(S,T,k)}{l(l+1)}$$
$$l = max\{|S|,|T|\} \tag{6}$$

where ACS is the set of all common subsequences of S and T without discontinuities, k is the length of the common subsequence and $\phi(S, T, k)$ is the number of common subsequences S and T without discontinuities of length k.

3 Classification results

3.1 Using a special kernel in the SVM method

The SVM method is implemented in the scikit-learn library [6]. The functions of the method support the data classification by features. There are no functions for sequence analysis. The implementation of the method provides the ability to connect custom kernel functions. We use this feature to code and connect functions that analyse sequences.

Kernels of the SVM method based on sequence similarity measures without discontinuity. In this paper, the previously calculated Gram matrix was transmitted to

the SVM method. Each element of the matrix in row *i* and column *j* corresponds to the calculated measure of the similarity of the two sequences with the numbers *i, j*. The size of the matrix is n^2, where *n* is the number of sequences. Accordingly, the calculation time has a quadratic dependence on the number of sequences.

The following functions for calculating sequence similarity without discontinuity were used: ACS, LCS and CP. Functions are written in Python.

The values of the functions ranged from 0 to 1. The value 1 is obtained for equal sequences, in particular, for the diagonal elements of the matrix. The value 0 is obtained when there is no similarity between sequences.

To evaluate classification quality by different methods, the class of sex for the test sample was used. The initial data were divided into training and test sets according to the ratio 80/20. Previously, the rows of data were shuffled for a more even distribution between the training and test sets. Calculation time and accuracy results for similarity functions are presented in Table 1.

Table 1. Classification using special functions of the SVM method with kernels based on sequence similarity measures without discontinuity

Parameter	CP	ACS	LCS
Model fitting time, sec	400.97	1580.86	1544.21
Prediction time, sec	98.66	394.20	388.06
Total time, sec	499.62	1975.06	1932.27
Accuracy	0.648	**0.659**	0.490

Classification accuracy of functions CP and ACS differ slightly, and the calculation time for the CP is much less; the quality of the LCS prediction is not satisfactory.

In this table and in all of the following tables methods are compared by accuracy, time is shown for information.

Classification by features using the SVM method. For comparison, we will classify not according to the sequences, but on the basis of the respondents' features: type of education, place of residence, religiosity, frequency of church attendance and generation. We use the SVM method with default parameters (kernel function - RBF). Results are presented in Table 2.

Table 2. Classification by features using the SVM method

Parameter	Value
Model fitting time, sec	3.62
Prediction time, sec	0.52
Total time, sec	4.14
Accuracy	0.615

The results in the table show that the accuracy is worse than with the classification by sequence of events. The built-in function is much faster, since it is implemented in C and does not calculate the sequences similarity function.

Classification by sequences, by features and by weighted sum of probabilities of sequences and features. We can try to improve the result by combining two methods of classification: by sequences and by features. This can be done using the probabilities of referring to a certain class, calculated by the SVM method. To get the probability values, you need to specify the qualifier parameter "probability = True":

```
clf = svm.SVC(probability=True)
```

As a result, the method returns a matrix with the number of columns equal to the number of classes. In each position, there will be a probability of assigning a sequence to the corresponding class.

Having obtained the probability tables for each method, we can classify based on the weighted sum of the probabilities of the two methods. Since the methods give different classification accuracies, the final probability of assigning an object to a class is calculated by the formula:

$$P = \frac{As \cdot Ps + Af \cdot Pf}{As + Af} \tag{7}$$

where
As — accuracy by sequences,
Af — accuracy by features,
Ps — probability by sequences,
Pf — probability by features.

Formula (7) takes into account the accuracy of the method for the final probability calculation. The probability calculated by each method will be included in the final result with a coefficient equal to the method accuracy. Results are presented in Table 3.

Table 3. Classification by sequences, by features and by weighted sum of probabilities of sequences and features

Parameter	CP	ACS	LCS
Accuracy of sequence classification (SVM, custom kernel functions: CP, ACS, LCS)	0.648	0.659	0.490
Accuracy of classification by features (SVM default - RBF)	0.615	0.615	0.615
Accuracy of classification by weighted sum of probabilities (7)	**0.678**	0.670	0.612

The table shows a noticeable improvement in the final result when two methods are combined.

Classification by features and sequences transformed into features. Another possible method of classification by sequences is by bringing each sequence to a set of features. After that, existing methods of classification by features could be used.

We consider as features a set of subsequences without discontinuity no greater than a certain length, such that one can compose all the sequences from them. We compose a dictionary from all possible subsequences without discontinuity for the available sequences. Such subsequences will be regarded as features of the sequence. We replace each sequence with a set of attributes corresponding to the subsequences that appear in it. It is possible to apply the SVM method to the set of attributes.

The maximum number of different subsequences of sequences without discontinuity is

$$\frac{n \cdot (n + 1)}{2} \tag{8}$$

where n is the length of the sequence.

The dependence of the number of subsequences without discontinuity on the length of the sequence is quadratic. For large sequences, it is necessary to introduce constraints. Let us consider subsequences without discontinuity no greater than a certain length.

This algorithm is similar to the ACS method used in the kernel function; however, in the kernel function only the number of common subsequences without discontinuity between the two sequences is considered. This algorithm takes into account their quality – each unique subsequence is considered as a feature of the sequence. Let us investigate the work of algorithms with existing sequences of demographic events. In the provided initial data, the maximum length of the sequence is eight, hence the maximum possible number of subsequences without discontinuity is 36, according to (8).

We will not consider all subsequences, but only one subsequence of maximum length for each sequence. In this case, there will be only one feature for each sequence, which will significantly reduce the amount of computation. Results are presented in Table 4.

Table 4. Classification by features and sequences transformed into features

Parameter	Classification by sequences only (as features)	Classification by features only	Classification by sequences and features
Number of sequences	6626	0	6626
The number of unique sequences of maximum length (number of features values)	1228	0	1228
Number of initial features	0	5	5
Number of generated features (from sequences)	1	0	1
Model fitting time, sec	4.80	3.62	5.79
Prediction time, sec	0.79	0.52	0.91
Total time, sec	5.59	4.14	6.70
Accuracy	0.675	0.615	**0.716**

It turned out that the classification according to the sequences transformed into features, in combination with the initial features, renders the best result in comparison with the above methods.

3.2 Recurrent neural network algorithms

We performed classification according to sequences using recurrent neural networks from Keras and Tensorflow software [7]. Keras, a top-level software, is used to describe the structure of a neural network as an add-on over the Tensorflow software, which performs the simulation of the neural network. The simulation was performed on the GeForce GT 710 GPU.

Recurrent Neural Network - RNN allows us to reveal regularities in sequences [8]. Three types of recurrent layers were compared in Keras: SimpleRNN, GRU and LSTM [5]. GRU and LSTM, in comparison with SimpleRNN, have more complex algorithms for detecting regularities. However, on sequences in the original demographic data, because of their small lengths, LSTM and GRU did not show any advantage in classification over SimpleRNN. At the same time, the simulation time for GRU and LSTM was several times larger. Therefore for subsequent classification by sequences, together with the features, only the SimpleRNN algorithm was used. The results are shown in the Table 5.

Table 5. Classification using recurrent neural networks (Keras, Tensorflow)

Method of classification	By sequences			By features	By sequences and features
	Neural network layers (number)				
Parameter	SimpleRNN Dense	GRU Dense	LSTM Dense	Dense Dropout	SimpleRNN Dense Dropout
The number of events in the sequences (maximum)	8	8	8	0	8
Number of features	0	0	0	5	5
Model fitting time, sec	168.80	452.27	585.73	348.49	418.05
Prediction time, sec	2.28	3.38	3.93	0.68	1.36
Total time, sec	171.07	455.66	585.73	349.17	419.40
Accuracy	0.676	0.672	0.675	0.626	**0.754**

Recurrent neural networks give the best result, since they are better than other algorithms at accounting for the dependencies in the sequences.

4 Comparison of all methods

Table 6 shows a comparison of the accuracies of classification methods. The table is supplemented by the result obtained by the algorithm of article [9] on the same data.

Table 6. Comparison of the accuracy of classification methods

Methods	Classification by sequences only	Classification by features only	Classification by sequences and features
Methods based on the SVM kernel custom functions			
Common Prefix similarity (CP)	0.648	0.615	0.678
All Common Subsequences similarity (ACS)	0.659	0.615	0.670

Longest Common Subsequence similarity (LCS)	0.490	0.615	0.612
Using sequences transformed into features	0.675	0.615	**0.716**
Recurrent neural networks (Keras, Tensorflow)			
SimpleRNN, Dense	0.676	0.626	**0.754**
GRU, Dense	0.672	0.626	
LSTM, Dense	0.675	0.626	
Decision trees [9]			
Time coding			0.661

Thus, the best result of classification of demographic data is given by recurrent neural networks (Keras, Tensorflow) on sequences with features, and the accuracy is 0.754.

5 Conclusion

In the course of the work, we derived formulas for calculating measures of sequence similarity without discontinuity; these were then incorporated to the kernels of the SVM method. We have studied several methods for classifying demographic data by sequences of events without discontinuities, namely variants of a custom kernel in the SVM method and recurrent neural networks. Also, we made a comparison of these methods with the algorithm from the earlier paper [9]. To complete our work, we wrote programmes in Python, with the help of which we processed the initial demographic data. We obtained solid classification results by using the custom kernel function in SVM by transforming sequences into features and even better results with recurrent neural network SimpleRNN. These two methods take into account event regularities in the sequences, unlike most other methods which work only with features. This work can be applied to the analysis of various sequences. Of course, many other classification methods based on different similarity measures of demographic sequences can be used. These may be statistical methods or other types of neural networks, such as convolutional neural networks (CNN). Those methods may be investigated in future research.

Acknowledgments. We would like to thank our colleagues from the research and study group “Models and Methods of Demographic Sequence Analysis” Dmitry Ignatov and Danil Gizdatullin for their piece of advice and Ekaterina Mitrofanova for the obtained data.

This article was prepared within the framework of the Academic Fund Program at the National Research University Higher School of Economics (HSE) in 2016-2017

(grant № 16-05-0011 “Development and testing of demographic sequence analysis and mining techniques”) and by the Russian Academic Excellence Project "5-100".

References

1. Elzinga, C.H., Liefbroer A.C.: De-standardization of Family-Life Trajectories of Young Adults. A Cross-National Comparison Using Sequence Analysis. European Journal of Population 23(3), 225-250 (2007).
2. Elzinga, C.H., Rahmann, S., Wang, H.: Algorithms for subsequence combinatorics. Theoretical Computer Science 409(3), 394-404 (2008).
3. Egho, E., Raïssi, C., Calders, T., Jay, N., Napoli, A.: On measuring similarity for sequences of itemsets. Data Mining Knowledge Discovery 29(3), 732-764 (2015).
4. Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N., Watkins, C.: Text Classification using String Kernels. Journal of Machine Learning Research 2, 419-444 (2002).
5. Understanding LSTM Networks, http://colah.github.io/posts/2015-08-Understanding-LSTMs/, last accessed 2017/02/15.
6. Scikit-learn: Scientific library for Machine Learning in Python, http://scikit-learn.org/, last accessed 2017/01/28.
7. Keras: Deep Learning library for Theano and TensorFlow, https://keras.io/, last accessed 2017/02/17.
8. The Unreasonable Effectiveness of Recurrent Neural Networks, http://karpathy.github.io/2015/05/21/rnn-effectiveness/, last accessed 2016/12/20.
9. Ignatov, D.I., Mitrofanova, E.S., Muratova A.A., Gizdatullin D.K.: Pattern Mining and Machine Learning for Demographic Sequences. In: Knowledge Engineering and Semantic Web: 6th International Conference, KESW 2015, vol. 518, pp. 225-243. Springer, Switzerland (2015).
10. Buzmakov, A., Egho, E., Nicolas, J., Kuznetsov, S.O., Napoli, A., Raïssi, Ch.: On mining complex sequential data by means of FCA and pattern structures. Int. J. General Systems 45(2), 135-159 (2016)
11. Ganter, B., Kuznetsov, S.O.: Pattern structures and their projections. In: Delugach, H.S., Stumme, G. (eds.) ICCS 2001. LNCS (LNAI), vol. 2120, pp. 129–142. Springer, Heidelberg (2001)
12. Ganter, B., Wille, R.: Formal Concept Analysis. Springer, Berlin (1999)
13. Gizdatullin, D., Baixeries, J., Ignatov, D., Mitrofanova, E., Muratova, A., Thomas H. Espy: Learning Patterns from Demographic Sequences. In.: Intelligent Data Processing, IDP 2016, Springer (to appear)
14. Gizdatullin,D., Ignatov, D., Mitrofanova, E., Muratova, A.: Classification of Demographic Sequences Based on Pattern Structures and Emerging Patterns. In.:14th International Conference on Formal Concept Analysis, Supplementary proceedings, ICFCA 2017, Rennes, France (2017)
15. Aggarwal, Ch. C., Han, J.: Frequent Pattern Mining. Springer (2014)