Generalized Variable Conversion Using K-means Clustering and Web Scraping

Kourosh Modarresi(✉) and Abdurrahman Munir

Adobe Inc., San Jose, CA, USA
kouroshm@alumni.stanford.edu, munir@adobe.com

Abstract. The world of AI and Machine Learning is the world of data and learning from data so the insights could be used for analysis and prediction. Almost all data sets are of mixed variable types as they may be quantitative (numerical) or qualitative (categorical). The problem arises from the fact that a long list of methods in Machine Learning such as “multiple regression”, “logistic regression”, “k-means clustering”, and “support vector machine”, all to be as examples of such models, designed to deal with numerical data type only. Though the data, that need to be analyzed and learned from, is almost always, a mixed data type and thus, standardization step must be undertaken for all these data sets. The standardization process involves the conversion of qualitative (categorical) data into numerical data type.

Keywords: Mixed variable types · NLP · K-means clustering

1 Introduction

1.1 Why this Work is Needed

AI and machine learning are mathematical modeling methods for learning from data and producing intelligent models based on this learning. The data these models need to deal with, is normally a mixed data type of both numerical (continuous) variables and categorical (non-numerical) data types. Most models in AI and machine learning accept only numerical data as their input and thus, standardization of mixed data into numerical data is a critical step when applying machine learning models. Having data in the standard shape and format that models require is often a time consuming, nevertheless very significant step of the process.

As an example, when we have a data set (below) combined of many variables where all variables are numerical ones except two variables of categorical type (gender and marital status) as following (Table 1):

Table 1. Original mixed variables

User	Age	Income	Gender	Marital status
1	31	90,000	M	Single
2	45	45,000	M	Married
3	63	34,000	M	Divorced
4	33	65,000	F	Divorced
5	47	87,000	F	Single
6	38	39,000	M	Married
7	26	120,000	M	Married
8	25	32,000	F	Married
9	29	55,000	F	Single
10	44	33,000	F	Single

When applying many machine learning models, the models need the data to be numerical data type. Thus, the categorical data should be converted into numerical type. The most efficient way of converting the categorical variable is the introduction of dummy variables (one hot encoding) for which a new (dummy) variable is created for each category (except the last category – since it'd be dependent on the rest of dummy variables, i.e., its value could be determined when all other dummy variables are known) of the categorical variable. These dummy variables are binary variables Queryand could assume only two values, 1 and 0. The value 1 means the sample has the value of that variable and 0 means the opposite.

Here, for this example, we have two categorical variables:

1. Gender: there are only two categories, so we need to create one dummy variable.
2. Marital Status: there are three categories so we need to create two new dummy variables.

The result after the creation of dummy variables is shown in Table 2.

Table 2. The original variables after the introduction of dummy variables.

User	Age	Income	Dummy variable-1 (Female)	Dummy variable-2 (Married)	Dummy variable-3 (Single)
1	31	90000	0	0	1
2	45	45000	0	1	0
3	63	34000	0	0	0
4	33	65000	1	0	0
5	47	87000	1	0	1
6	38	39000	0	1	0
7	26	120000	0	1	0
8	25	32000	1	1	0
9	29	55000	1	0	1
10	44	33000	1	0	1

Now, we could use any machine learning model for this data set as all its variables are of the numerical type.

In general, for any categorical variable of “m” categories (classes), we need to create “m − 1” dummy variables. The problem arises when any specific categorical variable has large (based on our work, that means larger than 8) number of categories. The reason is that, in these cases, the number of dummy variables need to be created becomes too large causing the data to become of high dimension. The high dimensionality of data leads to “curse of dimensionality” problem and thus all related issues related to “curse of dimensionality” such as the need of “exponential increase in the number of data rows” and “difficulties of distance computation” would appear. Obviously, one needs to avoid the situation since, in addition to these problems, curse of dimensionality also leads to misleading results from any machine learning models such as finding false patterns discovered based on noise or random chance. Besides all of that, higher dimension leads to higher “computational cost” and “slow model response and lower robustness”, all of which should be avoided. Therefore, in the process of transformation of categorical data into numerical data types, we must reduce the number of newly created numerical variables to reduce the dimension of data.

2 The Model

2.1 The Problem of Mixed Variables

The Vast majorities of the models in machine learning are models that use only numeric data. Though, practically all data that are used in machine learning are mixed type, numerical and categorical data. When used for machine learning models that could use only numerical data, mixed data types are handled using three different approaches: first approach is trying to, instead, using models that could handle mixed data type, second approach is to ignore (drop) categorical variables. The last approach is converting categorical variables to numerical type by introducing dummy variables or one hot encoding. The first approach introduces many limitations as there are only a limited number of models that could handle mixed data and those models may not the best model fitting the data sets. The second approach leads to ignoring much of the information in the data sets, i.e., the categorical data.

The practical approach is the third one, i.e., conversion of categorical data into numerical data. As we explained above, this can be done correctly only when all categorical variables have only limited number of categories. Else, it leads to high dimensional data that causes, among other problems, machine learning models to produce meaningless (biased) results. In other words, when the variable has many classes, this approach becomes infeasible because the number of variables will be too high for the numeric models to handle.

We can classify categorical variables into three types of variables. The first type is the ones without any clear and explicit features (like url, concatenated data, acronyms and so on). The second type of categorical variable occur when we have features (attributes) readily available as a part of data sets (or metadata). This is rarely seen in the data sets of the real world. In these cases that we have features for all categories or classes of any variable, we could use k-means clustering directly and follow it with the rest of the steps in this work. The third categorical data type is the case of categorical data without those readily available features. This paper addresses this last type of data where, quite often, there is no attributes information about these classes in the data sets and thus this we use NLP, Natural Language Processing [2, 13, 18–20, 40, 44, 45, 52, 56], models to establish these attributes. For our invention, we use web scraping to detect all features or attributes for our data sets. Then using these features, we use k-means clustering to compute a limited number of clusters that would represent the number of newly created features for the categorical data.

In this work, we also determine the upper bound for the number of new numerical variable created for conversion and representation of categorical variable. Besides, we define our way of testing the correctness and validation of our approach.

Therefore, to address these types of problem, this work establishes a new approach of reducing the number of categories (when the number of categories in a categorical variable in larger than 10) to K categories for K ≤ 10. We do it by clustering the categories of each of such categorical variable into k clusters, using k-means clustering. We compute the number of clusters, k, using silhouette method. We also use Silhouette method also to verify correctness of our models simultaneously. Then, the number of dummy variable needs to be created for any categorical variable of such will be reduced to K dummy variables, one for each cluster. Thereafter, the standardization is done by introducing K dummy variables.

Using the method explained above, this work detects a much smaller number of "latent classes", that in general could be some of the original attributes or some linear or non-linear combination of the original attributes, that are the underpinning classes or categories for the original categories of each categorical variable. This way, the high dimensionality is avoided and thus, we can use these latent classes to perform the dummy variable generation procedure that is described above to be used for any machine learning model. The small number of latent categories are detected using k-means clustering.

The basic idea is that categorical variables that have many values (or unique values for each sample) provide little information for other samples. To maintain the useful information from these variables, the best method may be to keep that useful (latent) information. This paper does it by finding the latent categories by clustering all categories into similar groups.

2.2 Computing the Number of Cluster K and Testing the Model

In this work, including for the three examples, to compute the optimal number of clusters, the upper bound for the number of clusters, and for testing and validation of our model, we use Silhouette method which is based on minimizing the dissimilarities inside a cluster and maximizing the dissimilarities among clusters:

The Silhouette model computes s(i) for each data point in the data set for each K:

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

Where $a(i)$ is the mean distance of point i to all the other points in its cluster. Also, $b(i)$ is the mean distance to all the points in its closest cluster, i.e., $b(i)$ is the minimum mean distance of point i to all clusters that i is not a member of.

The optimal K is the K that maximizes the total score s(i) for all data set. The score values lie in the range of [−1, 1] with −1 to be the worst possible score and +1 to be the optimal score. Thus, the closest (average score of all points) score to +1 is the optimal one and the corresponding K is the optimal K. Our experiments show that the value of K has upper bound of 10. Here, we use not only the score but the maximum separation and compactness of the clusters, as measured by distance between clusters and uniformity of the width of clusters, to test and validate our model simultaneously when computing optimal K.

In this work, we display the application of our model using three examples of categorical variables of large categories or classes. The first example is "country of residence" where there are over 175 categories or classes (countries). Secondly, we consider "city of residence (in the US)" as the second example where we use 183 most populated cities in the US. The third example of categorical variable with large categories that we use as an application of our model is "vegetables". For the vegetables, we have found records of 52 different classes (types of vegetables). In these examples, we show, that using our approach, we can find a small number of grouping within these variables and that these groupings can then be appended to the original data as dummy numeric variables to be used alongside the numeric variables.

2.3 The First Example of Categorical Variable, "Country of Residence"

Again, the issue is that there are so many categories for this categorical variable (country of residence), i.e., 175 categories. So, we need to create 174 dummy variables that would lead to a very high dimensional data and hence to "curse of dimensionality", as explained above. Here, we used clustering to group a list of 175 countries. For this

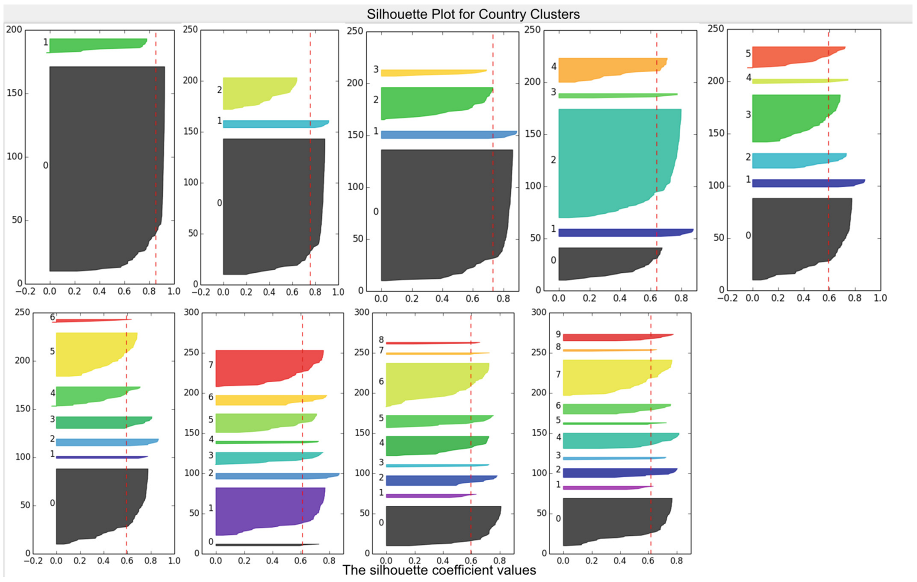
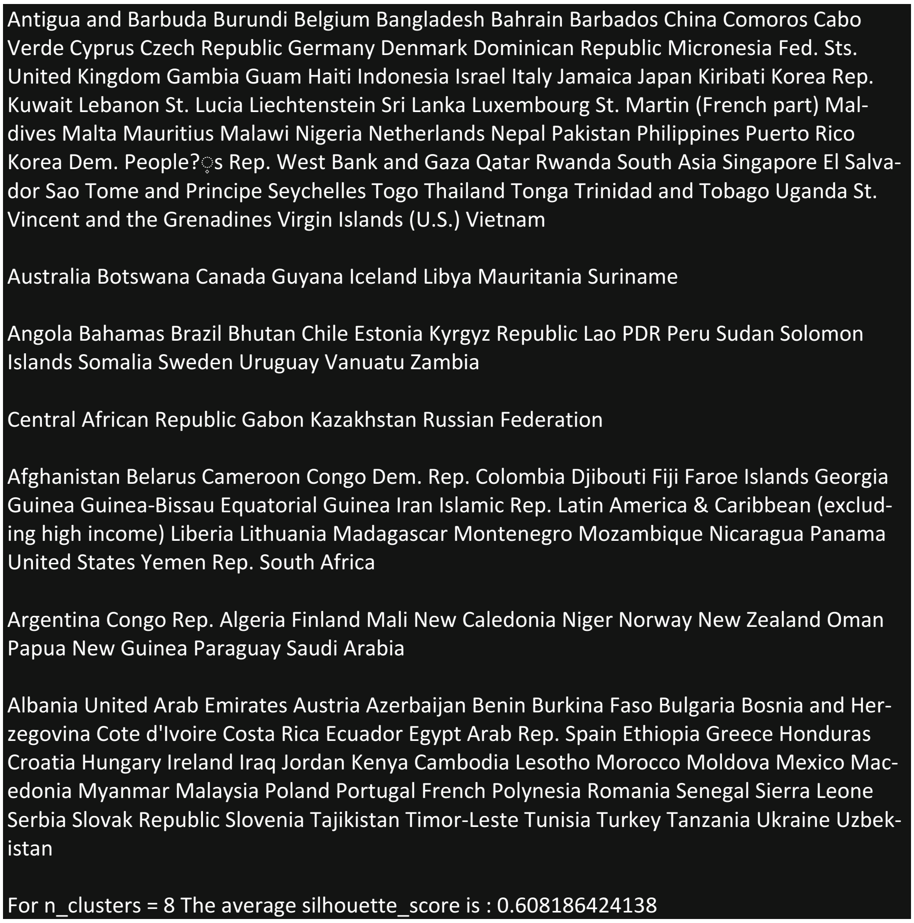


Fig. 1. The Silhouette plots displaying the optimal K to be 8.

case, syntactic similarity is useless since the name of a country has no relation to its attributes. Thus, we extracted the features from "www.worldbank.com". The seven features that we extracted, for each country, were: population, birth rate, mortality rate, life expectancy, death rate, surface area and forest area. These features were first normalized then K-means clustering was performed on the samples, again with a range of K from 2 to 10. Based off the silhouette plots in the following figure, Fig. 1, we can see that the algorithm performed well with K equal to 8:

country clustering output after k-means clustering is:



```
Antigua and Barbuda Burundi Belgium Bangladesh Bahrain Barbados China Comoros Cabo
Verde Cyprus Czech Republic Germany Denmark Dominican Republic Micronesia Fed. Sts.
United Kingdom Gambia Guam Haiti Indonesia Israel Italy Jamaica Japan Kiribati Korea Rep.
Kuwait Lebanon St. Lucia Liechtenstein Sri Lanka Luxembourg St. Martin (French part) Mal-
dives Malta Mauritius Malawi Nigeria Netherlands Nepal Pakistan Philippines Puerto Rico
Korea Dem. People?s Rep. West Bank and Gaza Qatar Rwanda South Asia Singapore El Salva-
dor Sao Tome and Principe Seychelles Togo Thailand Tonga Trinidad and Tobago Uganda St.
Vincent and the Grenadines Virgin Islands (U.S.) Vietnam

Australia Botswana Canada Guyana Iceland Libya Mauritania Suriname

Angola Bahamas Brazil Bhutan Chile Estonia Kyrgyz Republic Lao PDR Peru Sudan Solomon
Islands Somalia Sweden Uruguay Vanuatu Zambia

Central African Republic Gabon Kazakhstan Russian Federation

Afghanistan Belarus Cameroon Congo Dem. Rep. Colombia Djibouti Fiji Faroe Islands Georgia
Guinea Guinea-Bissau Equatorial Guinea Iran Islamic Rep. Latin America & Caribbean (exclud-
ing high income) Liberia Lithuania Madagascar Montenegro Mozambique Nicaragua Panama
United States Yemen Rep. South Africa

Argentina Congo Rep. Algeria Finland Mali New Caledonia Niger Norway New Zealand Oman
Papua New Guinea Paraguay Saudi Arabia

Albania United Arab Emirates Austria Azerbaijan Benin Burkina Faso Bulgaria Bosnia and Her-
zegovina Cote d'Ivoire Costa Rica Ecuador Egypt Arab Rep. Spain Ethiopia Greece Honduras
Croatia Hungary Ireland Iraq Jordan Kenya Cambodia Lesotho Morocco Moldova Mexico Mac-
edonia Myanmar Malaysia Poland Portugal French Polynesia Romania Senegal Sierra Leone
Serbia Slovak Republic Slovenia Tajikistan Timor-Leste Tunisia Turkey Tanzania Ukraine Uzbek-
istan

For n_clusters = 8 The average silhouette_score is : 0.608186424138
```

Fig. 2. The K-means clustering output for the first example.

In this example, the features extracted were not from only one domain, such as economic features only or just physical features. The advantage, of having a diverse domain features, is that the clusters that are formed will be more meaningful as they represent higher variation of data. For example, if our only feature was country size then the clustering algorithm would cluster algorithms with similar size. Additionally, if our only feature was country population then the algorithm would cluster countries with similar sizes. However, by using the different types of features, the algorithm could find clusters of countries that have both similar sizes and similar populations. For example, big countries with small populations could be in the same cluster as well as small countries that have large populations - - based their overall similarities computed using many various features.

2.4 The Second Example of Categorical Variable, "City of Residence" Using Web Scraping

To extract features for our categorical data (cities), we web scraped Wikipedia pages because of their abundant and concise data. The extraction came from the infobox on Wikipedia pages which contain quick facts about the article. We used five features which mainly pertained to the various attributes of the cities: land area, water area, elevation, population, and population density. For the most part, this was the only information available for direct extraction via Wikipedia pages. We extracted features for 183 U.S. cities then performed the same K-means clustering as in the previous examples to group the set into similar cities in each cluster. The most important aspect of this example is the web scraping. Whereas in the previous example, the features

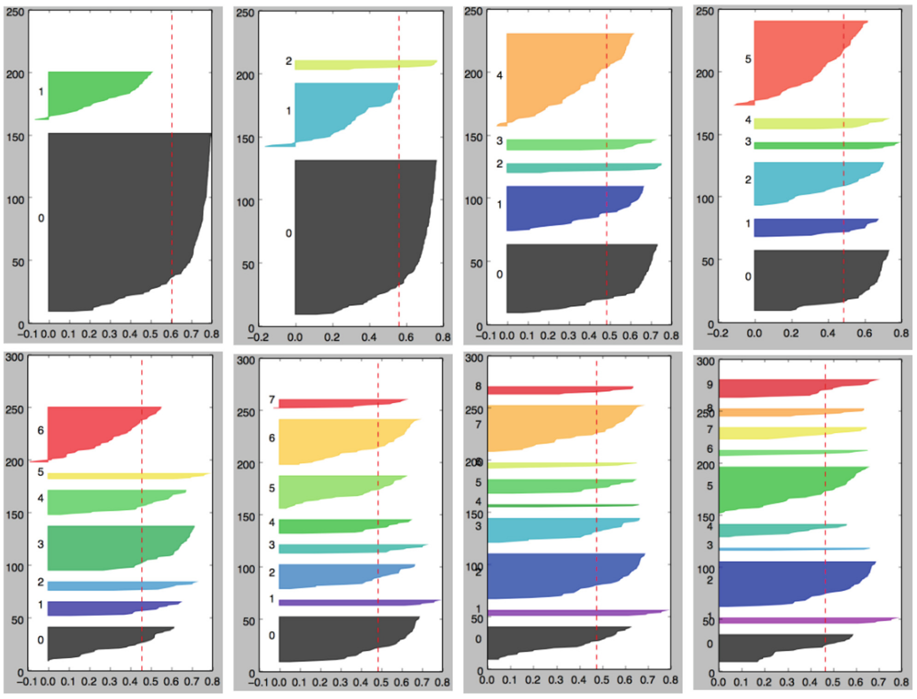
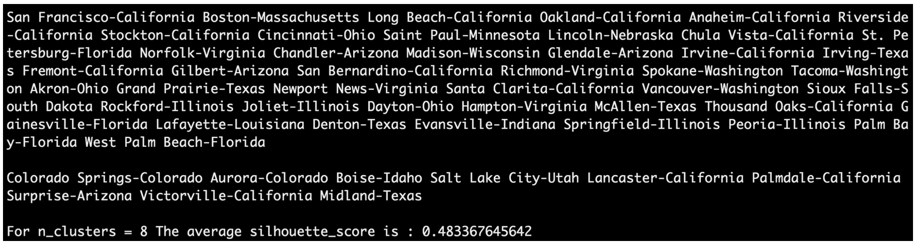


Fig. 3. The Silhouette model applied to this example. The plots display the optimal number of cluster to be K = 8.



```
San Francisco-California Boston-Massachusetts Long Beach-California Oakland-California Anaheim-California Riverside
-California Stockton-California Cincinnati-Ohio Saint Paul-Minnesota Lincoln-Nebraska Chula Vista-California St. Pe
tersburg-Florida Norfolk-Virginia Chandler-Arizona Madison-Wisconsin Glendale-Arizona Irvine-California Irving-Texa
s Fremont-California Gilbert-Arizona San Bernardino-California Richmond-Virginia Spokane-Washington Tacoma-Washingt
on Akron-Ohio Grand Prairie-Texas Newport News-Virginia Santa Clarita-California Vancouver-Washington Sioux Falls-S
outh Dakota Rockford-Illinois Joliet-Illinois Dayton-Ohio Hampton-Virginia McAllen-Texas Thousand Oaks-California G
ainesville-Florida Lafayette-Louisiana Denton-Texas Evansville-Indiana Springfield-Illinois Peoria-Illinois Palm Ba
y-Florida West Palm Beach-Florida

Colorado Springs-Colorado Aurora-Colorado Boise-Idaho Salt Lake City-Utah Lancaster-California Palmdale-California
Surprise-Arizona Victorville-California Midland-Texas

For n_clusters = 8 The average silhouette_score is : 0.483367645642
```

Fig. 4. The city clustering output after K-means clustering.

were taken from prebuilt online datasets, in this example we automatically built our own dataset by web scraping Wikipedia pages and constructing the features from this dataset. This shows that despite having a variable with many classes and no available information about the classes, we can extract the information necessary to perform the clustering. The following figure shows the silhouette model outcome:

As indicated, the silhouette plot for city clusters shows the number of newly variables, replacing 183 cities (categories), should be 8. Some of these clusters are shown here:

2.5 The Third Example: Categorical Variable, "Vegetables" Using Web Scraping

For the final example, we again use web scraping on a list of 52 vegetables to extract features. The features we extracted were: calories, protein, carbohydrates, and dietary

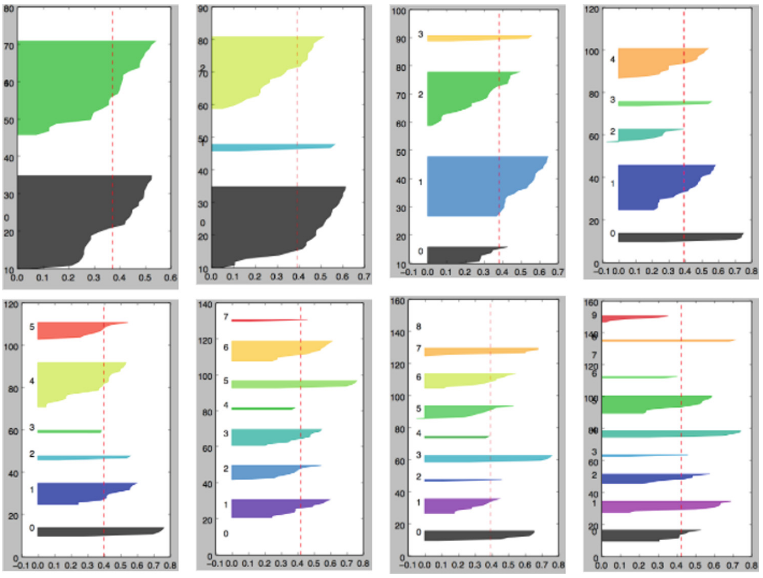
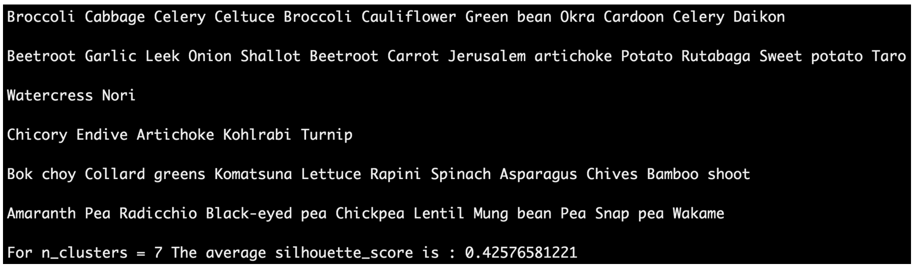


Fig. 5. The Silhouette plot indicating the optimal number of cluster is 7.

fiber. Like the previous example, we used Wikipedia articles to extract the features. Once again, this example shows the practicality of using web scraping as a means of automatically collecting features to build features for a dataset and then perform clustering on the dataset. The clustering of vegetables demonstrates the wide variety of variable types that our method can be applied to. The Silhouette plots is shown below with the optimal k to be 7:

Some of the clusters are shown below:



```
Broccoli Cabbage Celery Celtuce Broccoli Cauliflower Green bean Okra Cardoon Celery Daikon

Beetroot Garlic Leek Onion Shallot Beetroot Carrot Jerusalem artichoke Potato Rutabaga Sweet potato Taro

Watercress Nori

Chicory Endive Artichoke Kohlrabi Turnip

Bok choy Collard greens Komatsuna Lettuce Rapini Spinach Asparagus Chives Bamboo shoot

Amaranth Pea Radicchio Black-eyed pea Chickpea Lentil Mung bean Pea Snap pea Wakame

For n_clusters = 7 The average silhouette_score is : 0.42576581221
```

Fig. 6. Some of the clusters for the example three.

As shown by the images above, our algorithm is able to cluster the list of vegetables into groups based on similar nutritional benefit.

3 Conclusion

This work deals with the problem of converting categorical variables (to numerical ones) when the variables have high number of classes. We have shown the application of our model using three examples: countries, cities and vegetables. We use NLP plus clustering to show that even when there is no available information about the attributes, we could still perform clustering for the purpose of standardization of data. In the second example, we extracted external information about the values and then applied clustering using the information (features). In the second and third examples, we automatically extracted features from online resources. This information is needed for clustering. These three examples show that as long as there exists information about a variable, somewhere online, this information can be extracted and used for clustering. The final objective is to use the clustering method to drastically reduce the number of dummy variables that must be created in place of the categorical data type. Our model is practical and easy to use. It is an essential step in pre-processing data for many machine learning models.

References

1. Ahn, D., Jijkoun, V., Mishne, G., Müller, K., de Rijke, M., Schlobach, S.: Using Wikipedia at the TREC QA track. In: Proceedings of TREC (2004)
2. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.: DBpedia: a nucleus for a web of open data. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ASWC/ISWC -2007. LNCS, vol. 4825, pp. 722–735. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-76298-0_52
3. Backstrom, L., Leskovec, J.: Supervised random walks: predicting and recommending links in social networks. In: ACM International Conference on Web Search and Data Mining (WSDM) (2011)
4. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. In: International Conference on Learning Representations (ICLR) (2015)
5. Baudiš, P.:YodaQA: a modular question answering system pipeline. In: POSTER 2015-19th International Student Conference on Electrical Engineering, pp. 1156–1165 (2015)
6. Baudiš, P., Šedivý, J.: Modeling of the question answering task in the YodaQA system. In: Mothe, J., Savoy, J., Kamps, J., Pinel-Sauvagnat, K., Jones, Gareth J.F., SanJuan, E., Cappellato, L., Ferro, N. (eds.) CLEF 2015. LNCS, vol. 9283, pp. 222–228. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24027-5_20
7. Becker, S., Bobin, J., Candès, E.J.: NESTA: a fast and accurate first-order method for sparse recovery. SIAM J. Imaging Sci. **4**(1), 1–39 (2009)
8. Bjorck, A.: Numerical Methods for Least Squares Problems. SIAM, Philadelphia (1996)
9. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet allocation. J. Mach. Learn. Res. **3**, 993–1022 (2003)
10. Bollacker, K., Evans, C., Paritosh, P., Sturge, T., Taylor, J.: Freebase: a collaboratively created graph database for structuring human knowledge. In: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, pp. 1247–1250. ACM (2008)
11. Brill, E., Dumais, S., Banko, M.: An analysis of the AskMSR question-answering system. In: Empirical Methods in Natural Language Processing (EMNLP), pp. 257–264 (2002)
12. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press, Cambridge (2004)
13. Buscaldi, D., Rosso, P.: Mining knowledge from Wikipedia for the question answering task. In: International Conference on Language Resources and Evaluation (LREC), pp. 727–730 (2006)
14. Candès, E.J., Recht, B.: Exact matrix completion via convex optimization. Found. Comput. Math. **9**, 717–772 (2008)
15. Candès, E.J.: Compressive sampling. In: Proceedings of the International Congress of Mathematicians, Madrid, Spain (2006)
16. Candès, E.J., Tao, T.: Near-optimal signal recovery from random projections: universal encoding strategies. IEEE Trans. Inform. Theor. **52**, 5406–5425 (2004)
17. Caruana, R.: Multitask learning. In: Thrun, S., Pratt, L. (eds.) Learning to Learn, pp. 95–133. Springer, Boston (1998). https://doi.org/10.1007/978-1-4615-5529-2_5
18. Chen, D., Bolton, J., Manning, C.D.: A thorough examination of the CNN/daily mail reading comprehension task. In: Association for Computational Linguistics (ACL) (1998). 2016
19. Chen, D., Fisch, A., Weston, J., Bordes, A.: Reading Wikipedia to Answer Open-Domain Questions, arXiv:1704.00051 (2017)

20. Collobert, R., Weston, J.: A unified architecture for natural language processing: deep neural networks with multitask learning. In: International Conference on Machine Learning (ICML) (2008)
21. d'Aspremont, A., El Ghaoui, L., Jordan, M.I., Lanckriet, G.R.G.: A direct formulation for sparse PCA using semidefinite programming. SIAM Rev. **49**(3), 434–448 (2007)
22. Efron, B., Hastie, T., Johnstone, I., Tibshirani, R.: Least angle regression. Ann. Stat. **32**, 407–499 (2004)
23. Elden, L.: Algorithms for the regularization of Ill-conditioned least squares problems. BIT **17**, 134–145 (1977)
24. Elden, L.: A note on the computation of the generalized cross-validation function for Ill-conditioned least squares problems. BIT **24**, 467–472 (1984)
25. Engl, H.W., Groetsch, C.W. (eds.): Inverse and Ill-Posed Problems. Academic Press, London (1987)
26. Fader, A., Zettlemoyer, L., Etzioni, O.: Open question answering over curated and extracted knowledge bases. In: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1156–1165 (2014)
27. Fazel, M., Hindi, H., Boyd, S.: A rank minimization heuristic with application to minimum order system approximation. In: Proceedings American Control Conference, vol. 6, pp. 4734–4739 (2001)
28. Golub, G.H., Van Loan, C.F.: Matrix Computations, 4th edn. Computer Assisted Mechanics and Engineering Sciences, Johns Hopkins University Press, US (2013)
29. Golub, G.H., Van Loan, C.F.: An analysis of the total least squares problem. SIAM J. Numer. Anal. **17**, 883–893 (1980)
30. Golub, G.H., Heath, M., Wahba, G.: Generalized cross-validation as a method for choosing a good ridge parameter. Technometrics **21**, 215–223 (1979)
31. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning. SSS. Springer, New York (2009). https://doi.org/10.1007/978-0-387-84858-7
32. Hastie, T.J., Tibshirani, R.: Handwritten digit recognition via deformable prototypes. Technical report, AT&T Bell Laboratories (1994)
33. Hein, T., Hofmann, B.: On the nature of ill-posedness of an inverse problem in option pricing. Inverse Prob. **19**, 1319–1338 (2003)
34. Hewlett, D., Lacoste, A., Jones, L., Polosukhin, I., Fandrianto, A., Han, J., Kelcey, M., Berthelot, D.: Wikireading: a novel large-scale language understanding task over Wikipedia. In: Association for Computational Linguistics (ACL), pp. 1535–1545 (2016)
35. Hill, F., Bordes, A., Chopra, S., Weston, J.: The goldilocks principle: reading children's books with explicit memory representations. In: International Conference on Learning Representations (ICLR) (2016)
36. Hua, T.A., Gunst, R.F.: Generalized ridge regression: a note on negative ridge parameters. Comm. Stat. Theor. Methods **12**, 37–45 (1983)
37. Jolliffe, I.T., Trendafilov, N.T., Uddin, M.: A modified principal component technique based on the LASSO. J. Comput. Graph. Stat. **12**, 531–547 (2003)
38. Kirsch, A.: An Introduction to the Mathematical Theory of Inverse Problems. Springer, New York (1996). https://doi.org/10.1007/978-1-4419-8474-6
39. Mardia, K., Kent, J., Bibby, J.: Multivariate Analysis. Academic Press, New York (1979)
40. Manning, C.D., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S.J., McClosky, D.: The stanford corenlp natural language processing toolkit. In: Association for Computational Linguistics (ACL), pp. 55–60 (2014)
41. Marquardt, D.W.: Generalized inverses, ridge regression, biased linear estimation and nonlinear estimation. Technometrics **12**, 591–612 (1970)

42. Mazumder, R., Hastie, T., Tibshirani, R.: Spectral regularization algorithms for learning large incomplete matrices. JMLR **11**, 2287–2322 (2010)
43. McCabe, G.: Principal variables. Technometrics **26**, 137–144 (1984)
44. Miller, A.H., Fisch, A., Dodge, J., Karimi, A.-H., Bordes, A., Weston, J.: Key-value memory networks for directly reading documents. In: Empirical Methods in Natural Language Processing (EMNLP), pp. 1400–1409 (2016)
45. Mintz, M., Bills, S., Snow, R., Jurafsky, D.: Distant supervision for relation extraction without labeled data. In: Association for Computational Linguistics and International Joint Conference on Natural Language Processing (ACL/IJCNLP), pp. 1003–1011 (2009)
46. Modarresi, K., Golub, G.H.: An adaptive solution of linear inverse problems. In: Proceedings of Inverse Problems Design and Optimization Symposium (IPDO2007), 16–18 April, Miami Beach, Florida, pp. 333–340 (2007)
47. Modarresi, K.: A local regularization method using multiple regularization levels, Stanford, CA, April 2007
48. Modarresi, K.: Algorithmic approach for learning a comprehensive view of online users. Procedia Comput. Sci. **80C**, 2181–2189 (2016)
49. Modarresi, K.: Computation of recommender system using localized regularization. Procedia Comput. Sci. **51**, 2407–2416 (2015)
50. Rajpurkar, P., Zhang, J., Lopyrev, K., Liang, P.: SQuAD: 100,000+ questions for machine comprehension of text. In: Empirical Methods in Natural Language Processing (EMNLP) (2016)
51. Ryu, P.-M., Jang, M.-G., Kim, H.-K.: Open domain question answering using Wikipedia-based knowledge model. Inf. Process. Manag. **50**(5), 683–692 (2014)
52. Seo, M., Kembhavi, A., Farhadi, A., Hajishirzi, H.: Bidirectional attention flow for machine comprehension. arXiv preprint arXiv:1611.01603 (2016)
53. Tarantola, A.: Inverse Problem Theory. Elsevier, Amsterdam (1987)
54. Tibshirani, R.: Regression shrinkage and selection via the LASSO. J. Roy. Stat. Soc. Ser. B **58**(1), 267–288 (1996)
55. Tikhonov, A.N., Goncharsky, A.V. (eds.): Ill-Posed Problems in the Natural Sciences. MIR, Moscow (1987)
56. Wang, Z., Mi, H., Hamza, W., Florian, R.: Multi-perspective context matching for machine comprehension. arXiv preprint arXiv:1612.04211 (2016)
57. Witten, R., Candès, E.J.: Randomized algorithms for low-rank matrix factorizations: sharp performance bounds. To appear in Algorithmica (2013)
58. Zhou, Z., Wright, J., Li, X., Candès, E.J., Ma, Y.: Stable principal component pursuit. In: Proceedings of International Symposium on Information Theory, June 2010
59. Zou, H., Hastie, T., Tibshirani, R.: Sparse principal component analysis. J. Comput. Graph. Stat. **15**(2), 265–286 (2006)