DiscoVars: A New Data Analysis Perspective - Application in Variable Selection for Clustering

by Ayhan Demiriz

Abstract We present a new data analysis perspective to determine variable importance regardless of the underlying learning task. Traditionally, variable selection is considered an important step in supervised learning for both classification and regression problems. The variable selection also becomes critical when costs associated with the data collection and storage are considerably high for cases like remote sensing. Therefore, we propose a new methodology to select important variables from the data by first creating dependency networks among all variables and then ranking them (i.e. nodes) by graph centrality measures. Selecting Top-n variables according to preferred centrality measure will yield a strong candidate subset of variables for further learning tasks e.g. clustering. We present our tool as a Shiny app which is a user-friendly interface development environment. We also extend the user interface for two well-known unsupervised variable selection methods from literature for comparison reasons.

Introduction

Applications of machine learning and related technologies are dependent on successful implementations of learning algorithms and strong data analytic skills of practitioners. Often complexity and black box nature of the algorithms disengage practitioners from the knowledge discovery process. Recently, high dimensional data and uninterpretable complex models are two major reasons that practitioners have started shying away from interacting with learning models and have preferred AutoML Martinez-Plumed et al. (2021); Truong et al. (2019). Incorporating domain knowledge and expertise into algorithmic models certainly allows practitioners to manage the learning process.

Feature (variable) selection is an important task and can be performed by employing a large array of methods in the literature Li et al. (2018); Guyon and Elisseeff (2003); Caruana and de Sa (2003); Forman (2003). Depending on learning task, supervised and unsupervised variable selection are possible. Considering that machine learning problems have been studied for several decades, many well known methods were originally developed under very limited computational resource constraints. Naturally, dimension reduction is one way of reducing problem complexity for the underlying methods. Feature extraction and feature selection are two alternative choices for dimension reduction Li et al. (2018). The first alternative is typically used for projecting the full input space to a lower dimension. Feature extraction requires the usage of full input data in data preparation stage of the underlying learning method. Moreover, such methods may still require more computational resources and full data at later stages of the learning process. Feature selection, on the other hand, picks a subset of all variables presumably without any sacrifice in the performance of learning process. The purpose of feature selection is primarily to reduce the computational complexity and then to increase the performance of learning process which could be in supervised, unsupervised and semi-supervised fashions. Overfitting is also another reason to utilize feature selection to prevent very poor generalization in learning processes.

Categorization of the feature selection methods is done from various perspectives in practice Li et al. (2018); Guyon and Elisseeff (2003). Supervision and selection strategy perspectives are the most common categorization of feature selection methods Li et al. (2018). Depending on level of label data, supervised, unsupervised and semi-supervised approaches are utilized in feature selection. Specifically, label information is used in classification and regression problems. Subset of features are selected based on some performance measures such as accuracy and R^2. In addition to level of supervision, selection strategy is another categorization of feature selection methods. Selection could be implemented as a wrapper method which calls the learning model like a black box with different combinations of input data and searches for the best possible subset of variables. Since there are 2^d different input possibilities for d-dimensional data, the combinatoric nature of this approach may slow down the search. However, introducing some evaluation criteria may speed up the process. For example, traditional stepwise, backward and forward selection methods can be considered as wrapper methods in multivariate regression problems and these methods simply pick a feature based on its contribution to the overall R^2 value at each iteration. Usually one feature is added/subtracted from the regression model at each iterative search step. Obviously, supervision is also used. The second approach for selection strategy is filter methods. The main idea is to select a subset of variables by ranking them prior to running the learning algorithm. Potentially, filter methods can be used irrespective of underlying learning task i.e. supervised, unsupervised or semi-supervised. Ranking could be based on an intrinsic measure such as correlation and mutual information Chandrashekar and Sahin (2014). Filter methods are composed of two steps: the first step ranks the variables according to some measure either univariate or multivariate (i.e. multiple features) way, the second step removes low ranked features Li et al. (2018). The third way of selection strategy is to embed feature selection into model learning Li et al. (2018). This is somewhat a hybrid approach by combining best parts of previous two approaches.

We propose an interactive multivariate filter method that first creates a dependency network Heckerman et al. (2000) among all features then the nodes of this network are ranked based on centrality measure chosen by the user. For commonly used network centrality measures, interested user is referred to Hansen et al. (2020). Finally, Top-n variables are selected by the user (modeler). Our approach enables modeler to discover candidate variable subset through an interactive method by utilizing dependency networks and graph centrality measures. Note that each node in dependency networks corresponds to a variable. Thus our method is named as DiscoVars (Discover Variables). Since supervised feature selection methods are established on concrete performance measures, we opt to implement our feature selection method for clustering analysis. Nevertheless, our approach can directly be used in classification problems as well. Novelty of our approach is to utilize graph centrality metrics to determine importance of variables on dependency networks constructed by efficient and proven variable selection methods such as stepwise, forward, and Lasso Tibshirani (1996).

Throughout the paper, $\mathcal{X}$ represents $m \times d$ dimensional dataset, i-th dimension of $\mathcal{X}$ is practically the random variable X_i, Top-n variables are the selected n features by the user (modeler), $\mathcal{S}$ represents $m \times n$ dimensionally reduced dataset, and k is the number of clusters set for clustering algorithms, a dependency network is a directed graph $\mathcal{G} = (V, E)$ of vertices V, edges E. The organization of the paper is as follows. Section 2 introduces the methodology proposed in this paper. As an application domain, unsupervised feature selection is considered suitable for our methodology. Section 3 reviews some related work in feature selection for clustering problems. Comparison of performances of clustering methods is very subjective by its nature. We report the detail implementation of our proposed approach DiscoVars in Section 4. For comparison reasons we also implemented two-well known feature selection methods on Shiny framework in Section 5. Section 6 then concludes our paper.

Methodology

Graphical models Koller et al. (2007) are considered as a popular tool for represent real world problems by combining uncertainty and logical structure i.e. conditional independence of variables. Bayesian and Markov Networks are the most common graphical models studied in the literature. Bayesian Networks are directed graphs. Markov Networks, on the other hand are undirected graphs. Graphical models primarily help on probabilistic inference. Thus, the goal is to derive a joint distribution P over set of random variables $\mathscr{X} = \{X_1, \dots, X_d\}$. Conditional independence is an important structural concept that simplifies underlying graphical models. According to Definition 2.1 of Koller et al. (2007), conditional independence is defined as $P(X = x, Y = y \mid Z = z) = P(X = x \mid Z = z)P(Y = y \mid Z = z)$ for all x, y, and z values. Assuming conditional independence of variables $\{X_1, \dots, X_d\}$ in a Bayesian Network (BN) $\mathcal{G}$, one can factorize joint probability distribution of BN as,

$$P_B(X_1, \dots, X_d) = \prod_{i=1}^{d} P(X_i \mid Pa_i),$$

where Pa_i are the parents of random variable X_i. Note that this rule is also called as chain rule of BN. A greedy algorithm that combines local and global structure search is proposed in Chickering et al. (1997) which utilizes conditional probability distributions. We implicitly assume that a random variable X_i and its non-descendants are conditionally independent given X_i's parents Pa_i. Markov Networks can also be constructed by factorizing over structural sub-graphs that are independent. Graphical models are computationally expensive to construct. In addition, it is very hard to interpret these models by untrained practitioners. Modelers prefer to interpret connections in these networks as relationships. However, relationships are casual and may not necessarily correspond to predictive ones.

Dependency networks (DNs) are considered as computationally efficient to construct Heckerman et al. (2000). The primary intention to develop DNs was to visualize predictive relationships. Commercial version was available with MSSQL Server 2000. It is claimed in Heckerman et al. (2000) that it was very useful to discover dependency relationships as a graphical tool. By varying (increasing) the strength threshold of relationships, sparser networks can be drawn. The simplicity of a consistent DN comes from the fact that conditional distributions of variables X_i can be found as:

$$P(X_i \mid Pa_i) = P(X_i \mid \mathscr{X} \backslash X_i),$$

where $\mathscr{X} \backslash X_i$ means all the variables in $\mathscr{X}$ except X_i and DN is consistent in a way that local probability distributions can be obtained from $P(\mathscr{X})$. It is shown in Heckerman et al. (2000) that consistent DNs are equivalent to Markov Networks. In other words, one can learn the structure of Markov Network from a consistent DN. This is an important result that makes DNs as practical graphical models for probabilistic inference, predicting preferences (collaborative filtering) Heckerman et al. (2000), and sequential data inference (prediction) Carlson et al. (2008). Constructing a DN may require using classification/regression models for estimating the local distributions Heckerman et al. (2000). In commercial setting, probabilistic decision trees were implemented as a default learning method because of their simplicity and computational efficiency. Note that DNs are primarily used for supervised prediction problems. So, the underlying graph represents the significant relationships for the target (label) variable in consideration.

Our approach is primarily based on constructing a DN by fitting each variable X_i with the remaining variables $\mathscr{X} \backslash X_i$ to represent all the significant relationships among all the variables. A statistical variable selection method M should be used for determining significant variables for local distributions. Assume that the set s represents the indices of significant variables for X_i and the variables X_s are parents Pa_i of X_i. In other words, X_i depends on variables X_s. All of these relationships form the DN G. Note that we assume that all X_i variables are continuous variables for this paper. Categorical data and classification methods can subsequently be incorporated into constructing DNs easily. So Stepwise, Forward, Akaike Information Criterion (AIC) and Lasso selection methods can be utilized by linear regression models (lm) in our approach at this time. The pseudo code of our methodology is given in Algorithm 1.

Algorithm 1 DiscoVars: Construct DN $\mathcal{G}$

```
1: Given $\mathcal{X}$ and $M$
2: for $i \leftarrow 1, d$ do
3:     $Fit\ X_i \leftarrow lm(\mathscr{X} \backslash X_i, M)$
4:     $s \leftarrow \{significant\ variables\}$
5:     $E_{is} \leftarrow 1$
6: end for
7: $c \leftarrow Select\ Centrality\ Measure$
8: $Rank(\mathcal{G}, c, \mathscr{X})$
9: $Select\ Top - n\ variables$
```

Interactive nature of our approach is based on choices of variable selection method M, centrality measure c and n for Top-n important variables. The methodology given in Algorithm 1 is parallelizable due to the nature of `for` loop. However, one can argue that it is computationally expensive to construct d regression models in the first place. Note that we can still apply filter methods to screen very high dimensional data ahead of constructing DNs if there is a consideration of computational resources. A stochastic search algorithm is proposed for constructing DNs on gene expression data for the purpose of supervised variable selection in Dobra (2009). Underlying graph structure among features can improve classification models for genome data Sun et al. (2020). By coupling graph structure with feature selection performs better than traditional feature selection methods Sun et al. (2020).

Centrality measures are used in our approach to rank nodes (variables) in DNs. It is a well established research topic to quantify structural importance of actors in a network Borgatti (2006). The centrality measures suitable for directed graphs such as betweenness, closeness, degree, eigenvector, and pagerank etc. can be used in our methodology. In common social network analysis problems, networks are composed of objects, people, or events. But features (variables) are the center of attention in our approach. In general, network science aims to study the structural relationships and importance in networks. Network topology determines the structural importance of nodes Roddenberry and Segarra (2020). Centrality measures are usually computed based on full network topology. Some recent work enables inference of eigenvector rankings from data directly without inferring the full network topology Roddenberry and Segarra (2020, 2021). So one can argue that it is technically possible to rank variables without first constructing DNs, but this is not the scope of this paper. The full network topology is constructed first in this study to calculate centrality measures.

In Meinshausen and Bühlmann (2006), local neighborhood structures are found by utilizing Lasso Tibshirani (1996). The main idea in Meinshausen and Bühlmann (2006) is to estimate the graph structure of covariance matrix for high dimensional data. Similar to our approach local dependencies are found by Lasso. Note that that Lasso is an option in our implementation. Our approach basically differs from Meinshausen and Bühlmann (2006) once DN is constructed. In Meinshausen and Bühlmann (2006), Lasso is used for constructing a sparse graph structure that all remaining variables in the graph become relevant for the learning task. Note that our approach is independent from learning task. Centrality measures are further used to filter most important variables (nodes) in the graph. There exist also distributional assumptions in Meinshausen and Bühlmann (2006). The size of neighborhood in Lasso is very sensitive to the selection of regularization parameter λ. Generally, cross-validation is common way of setting the parameter. In extreme cases, very sparse and full connections may be found by Lasso. Some remedies are offered in Meinshausen and Bühlmann (2006) for Lasso to result in robust neighborhood formation i.e. connections in network. Therefore neighborhoods are formed in a more stable way compared to original Lasso formulation. Same situation may also be an issue in our approach. If network is extremely sparse or almost full in connections, graph centrality measures will not yield decisive rankings.

Structural properties of variable neighborhoods can also help computing Laplacian Scores He et al. (2005). Variables can be ranked based on this score. This is another filter method that shares similar perspective with our work. Moreover, traveling salesman problem (TSP) is a universal test bed of ideas. In regular TSP, nodes (cities) have full connections. In other words, one can travel from any city to any other city directly. However, if sparse connections exist meaning that one can only move over physical road networks, we can rank cities based on underlying graph (network) structure. Thus, TSP can be solved by ranking cities first Demiriz (2009). This is also very similar to our idea in this paper.

Our approach can be used as a filter method irrespective of underlying learning task. There are widely used and proven feature selection methods for supervised learning problems due to availability of robust performance measures. We think that applying our approach may make an impact on unsupervised feature selection. Therefore, $\mathcal{S}$ can easily be clustered by any clustering algorithm after applying Algorithm 1.

Unsupervised feature selection

In practice, there exist some metrics to compare clustering results such as Davies-Bouldin Index (DBI) Davies and Bouldin (1979) and Adjusted Rand Index (ARI) Hubert and Arabie (1985). It is technically possible to devise a wrapper algorithm by incorporating some intelligence in search mechanism to run clustering methods to come up with best possible feature subset. These clustering indices are easy to interpret but they are not guaranteed to be useful for a robust search optimization. They are not only dependent on features selected but also number of partitions at the same time. Certainly, some heuristics methods can be deployed to find a suitable subset of features. However this is not within the scope of our paper.

An excellent review of feature selection methods for model-based clustering is given in Fop and Murphy (2018). Technically, the aim of variable selection is to determine the set of relevant variables for clustering. Logically, the remaining variables are called irrelevant. Two major assumptions are local conditional independence assumption of relevant variables within clusters and global independence assumption of irrelevant and relevant variables. Model-based clustering assumes that each observation comes from a finite mixture of G probability distributions. Obviously, each distribution represents a different group Raftery and Dean (2006); Scrucca and Raftery (2018); Scrucca et al. (2016). Bayesian Information Criterion (BIC) is commonly used in model (i.e. performance) comparison for clustering. BIC can be calculated as follows Raftery and Dean (2006).

$$BIC = 2 * \log(\text{maximized likelihood}) - (\text{no. of parameters}) * \log(m).$$

Technically, BIC values of inclusion and exclusion of variable X_i can be compared and a decision is made regarding that variable Scrucca and Raftery (2018).

A Lasso like approach is proposed in Witten and Tibshirani (2010). Technically, between cluster sum of squares for feature X_i is optimized and regularization terms L_1 (Lasso) and L_2 are applied on weights of features. L_2 penalty term guarantees non-zero solution. L_1 penalty term forces for sparser solutions Witten and Tibshirani (2010).

Witten and Tibshirani (2010) maximizes between cluster sum of squares. Alternatively, it is also possible to reduce within group variance Andrews and McNicholas (2014). VSCC method Andrews and McNicholas (2014) selects iteratively those variables that have smaller within-group variance and are correlated. We believe that feature selection for model-based clustering is well-studied. Interested readers are referred to Fop and Murphy (2018). In Section 5, two well-known methods are implemented in Shiny framework for comparison and reproducibility purposes.

DiscoVars

Our approach is implemented by using Shiny [1] framework in R to design and develop interactive applications (Figure 1). The current version of DiscoVars[2] can be accessed at Github repository [3]. As outlined in Algorithm 1, DiscoVars consists of two main steps:

- Constructing DN
- Ranking and selecting Top-n variables interactively according to choice of network centrality measure.

DiscoVars can import data from various sources by utilizing **datamods**[4] R package (see Figure 2). **datamods** is also a Shiny application. As seen in Figure 2, user can import datasets available from other R packages such as **mlbench**. Notice that `BostonHousing` Dataset from **mlbench** is used for illustration purposes.

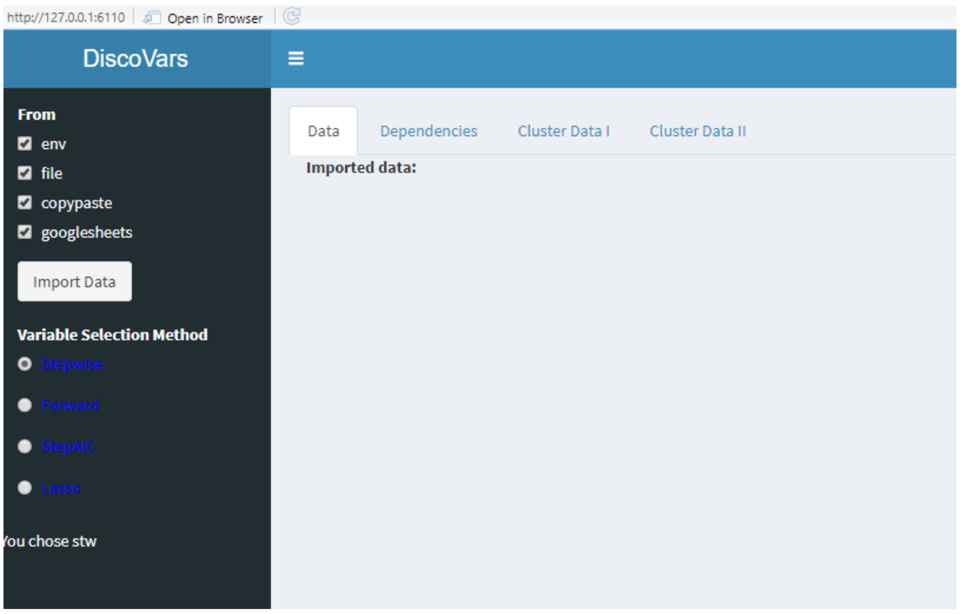
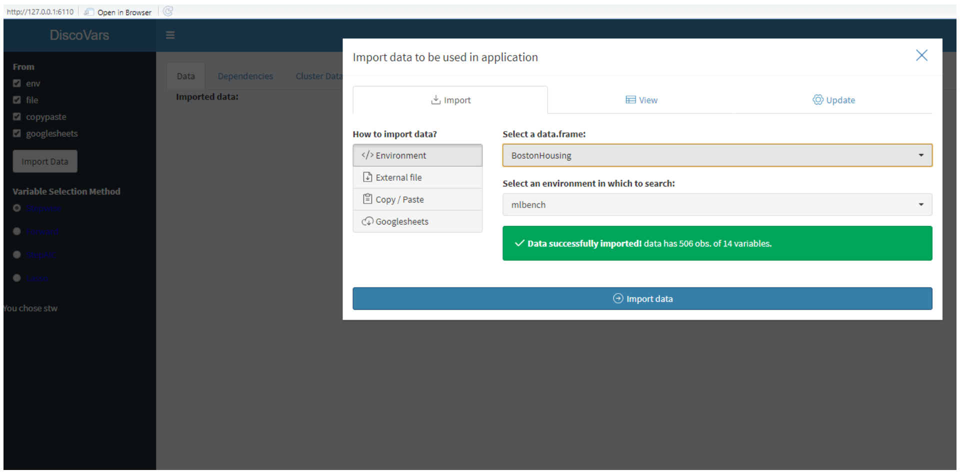


Figure 1: DiscoVars Opening Screen

Note that only numeric variables are included in our implementation for the time being. Categorical variables can also be included via probabilistic decision tree models to construct DNs. Once the dataset selected by the user as in Figure 2, user can filter data and/or exclude some unrelated variables (such as numerical ID variable) from dataset. Once dataset is imported, only numeric variables are used in DiscoVars. Imported dataset is presented to the user via `dataTable` object (Figure 3). At this moment, user can run dependency discoverer by choice of variable selection method M. Four well-known variable selection methods are available in DiscoVars. The default method is `Stepwise`; `Forward`, `stepAIC` and `Lasso` are also presented to the user as alternatives. `Stepwise` and `Forward` methods are available from `olsrr` package, `stepAIC` is available from **MASS** package and `Lasso` is available from **glmnet** package in R. For `stepwise` selection, $p \leq 0.1$ entry and $p \geq 0.25$ exit parameters are set. For `forward` selection, $p \leq 0.1$ entry parameter is set. Default parameter settings are used for `stepAIC`. With the default settings, **glmnet** runs Lasso with a varying number of λ values. Therefore, a model selection is required. The parameter `s` in **glmnet** is set to $16/m$ where m is the number of rows. Cross-validation can be used in **glmnet** to pick the best model. This option is not use to avoid extended run times.

DN can be constructed by running d regression models as given in step 3 of Algorithm 1. Since this step is fully parallelizable, d regression models are run by calling `parSapply` function in **doParallel** R package. Parallelization obviously speeds up this step significantly. Figure 4 depicts the DN constructed. User is also able to redraw the network according to choice of centrality measures - `alpha`, `authority`, `betweenness`, `closeness`, `degree`, `eigen`, `hub`, `pagerank`, and `power`. The default measure is `alpha`. Once user finalizes the centrality measure and n for the variable selection, Top-n variables are listed in a `dataTable` object. n is set by a `sliderInput` object (see Figure 5).

[1] https://shiny.rstudio.com/
[2] Software runs on RStudio.
[3] https://github.com/ademiriz/DiscoVars
[4] https://dreamrs.github.io/datamods/index.html



Import data to be used in application

Import | View | Update

How to import data?
- Environment
- External file
- Copy / Paste
- Googlesheets

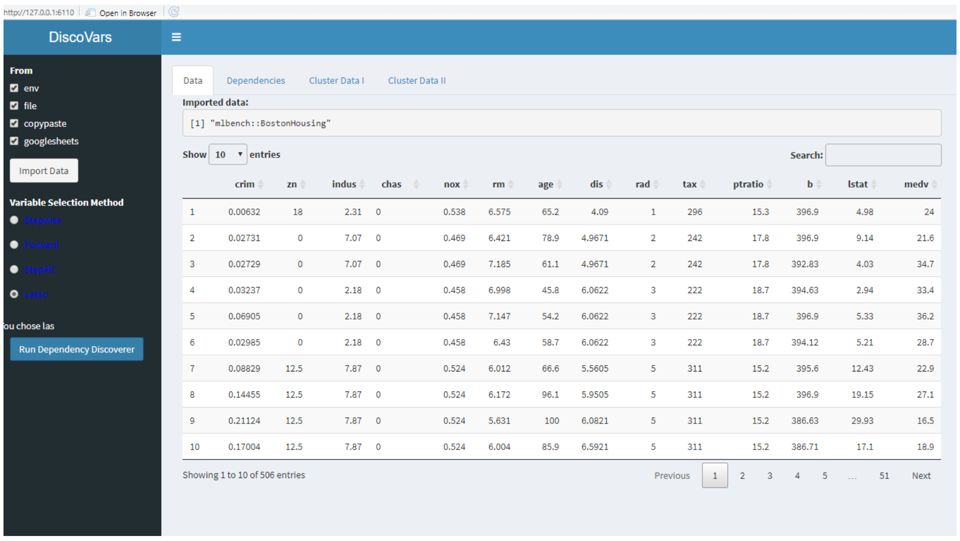
Select a data.frame: BostonHousing

Select an environment in which to search: mlbench

Data successfully imported! data has 506 obs. of 14 variables.

Import data

Figure 2: Data Import Screen



Imported data:

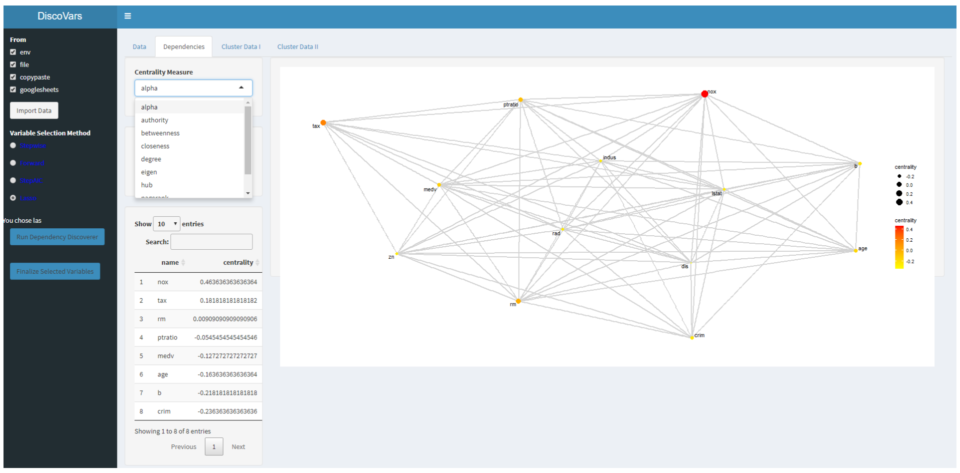
```
[1] "mlbench::BostonHousing"
```

Show 10 entries   Search:

	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	b	lstat	medv
1	0.00632	18	2.31	0	0.538	6.575	65.2	4.09	1	296	15.3	396.9	4.98	24
2	0.02731	0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.9	9.14	21.6
3	0.02729	0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03	34.7
4	0.03237	0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	33.4
5	0.06905	0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.9	5.33	36.2
6	0.02985	0	2.18	0	0.458	6.43	58.7	6.0622	3	222	18.7	394.12	5.21	28.7
7	0.08829	12.5	7.87	0	0.524	6.012	66.6	5.5605	5	311	15.2	395.6	12.43	22.9
8	0.14455	12.5	7.87	0	0.524	6.172	96.1	5.9505	5	311	15.2	396.9	19.15	27.1
9	0.21124	12.5	7.87	0	0.524	5.631	100	6.0821	5	311	15.2	386.63	29.93	16.5
10	0.17004	12.5	7.87	0	0.524	6.004	85.9	6.5921	5	311	15.2	386.71	17.1	18.9

Showing 1 to 10 of 506 entries   Previous 1 2 3 4 5 … 51 Next

Figure 3: Data Table



Centrality Measure: alpha

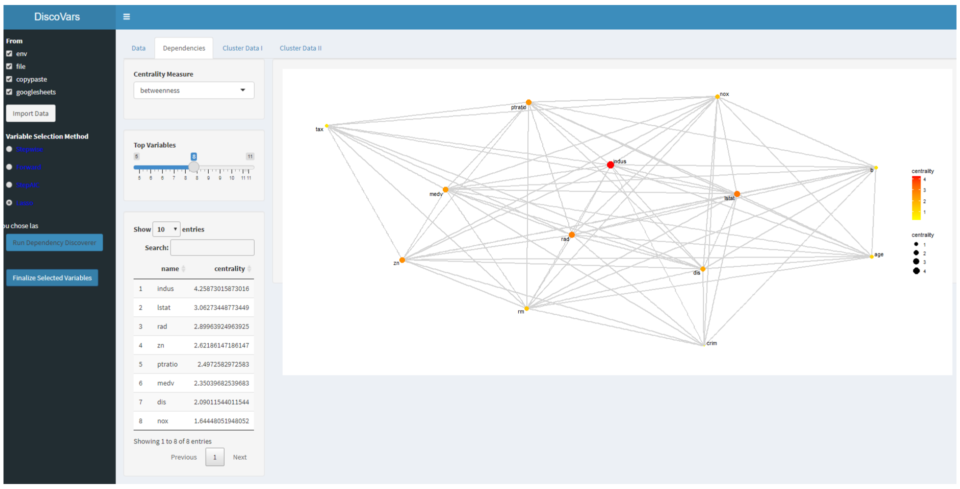
Show 10 entries

Search:

	name	centrality
1	nox	0.463636363636364
2	tax	0.181818181818182
3	rm	0.00909090909090906
4	ptratio	-0.0545454545454546
5	medv	-0.127272727272727
6	age	-0.163636363636364
7	b	-0.218181818181818
8	crim	-0.236363636363636

Showing 1 to 8 of 8 entries   Previous 1 Next

Figure 4: Dependency Network and Setting Centrality Measure



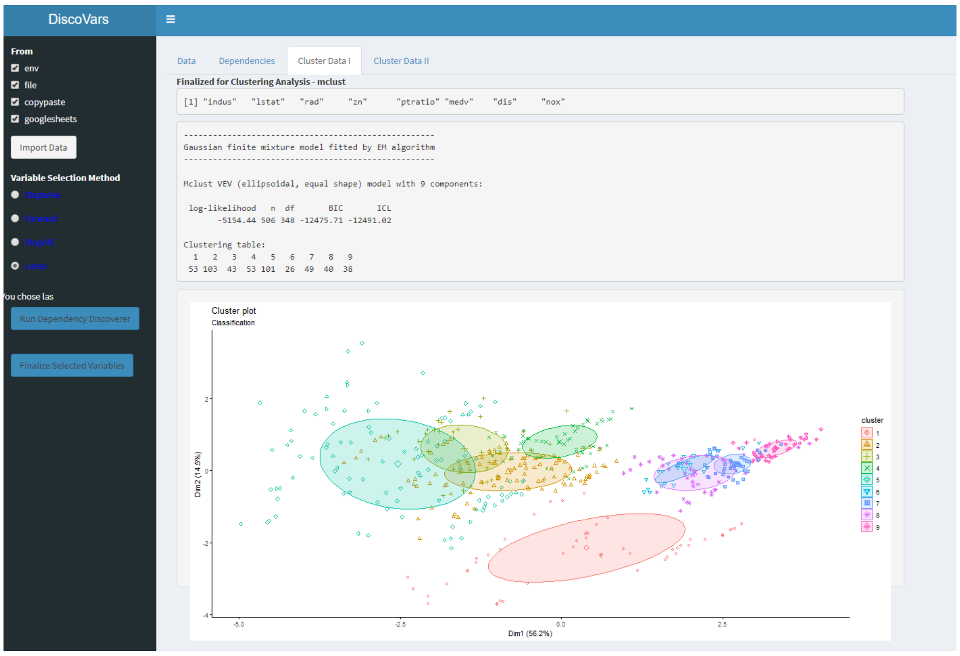
Centrality Measure: betweenness

Show 10 entries

	name	centrality
1	indus	4.25873015873016
2	lstat	3.06273448773449
3	rad	2.89963924963925
4	zn	2.62186147186147
5	ptratio	2.4972582972583
6	medv	2.35039682539683
7	dis	2.09011544011544
8	nox	1.64448051948052

Showing 1 to 8 of 8 entries

Figure 5: Finalizing Top-*n* Variables



Finalized for Clustering Analysis - mclust

```
[1] "indus"   "lstat"   "rad"     "zn"      "ptratio" "medv"    "dis"     "nox"
```

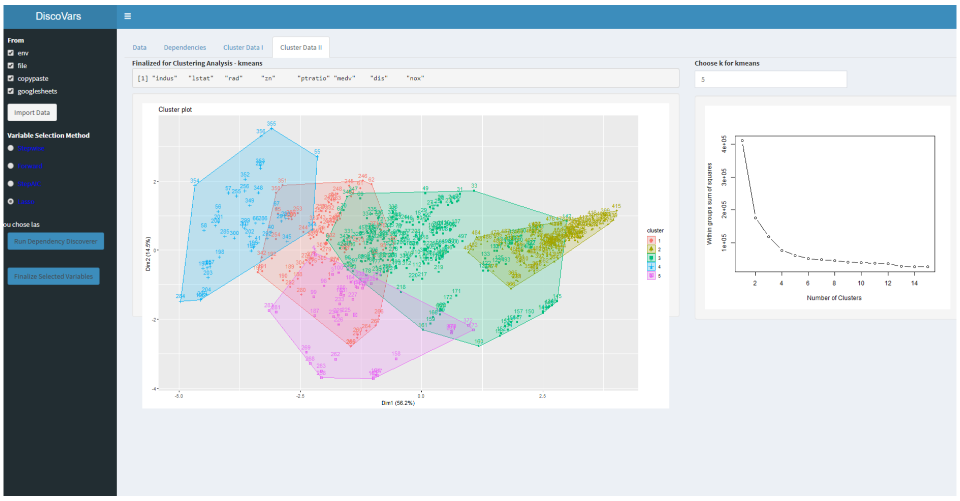
```
----------------------------------------------------
Gaussian finite mixture model fitted by EM algorithm
----------------------------------------------------

Mclust VEV (ellipsoidal, equal shape) model with 9 components:

 log-likelihood   n  df       BIC       ICL
       -5154.44 506 348 -12475.71 -12491.02

Clustering table:
  1   2   3   4   5   6   7   8   9
 53 103  43  53 101  26  49  40  38
```

Figure 6: Results of mclust Algorithm

After finalizing Top-*n* variables, various clustering algorithms can be deployed to group data. **mclust** Scrucca et al. (2016) and k-means algorithms are utilized in DiscoVars. Figures 6 and 7 depict outputs of **mclust** and k-means respectively by using Top-*n* variables. The default number of cluster parameter for **mclust** is 9 and it picks the best grouping with this initial condition. User can pick number of groups k for k-means algorithm (see Figure 7). Elbow plot is also shown in k-means output screen. Clustering results are plotted based on first two principal components in Figures 6 and 7. **factoextra** package is used for this purpose. Notice that the whole process is interactive. Even if variable selection method, M, is changed DN will be reconstructed and Top-*n* variables will be updated accordingly.

The most critical and time consuming part of DiscoVars is constructing DNs via d regression models. DiscoVars can easily be used to filter important variables for numerical datasets. In order to report running times of DiscoVars for DN construction, we chose two different domains: anthropometric data and crypto currency market data. Anthropometry is the study of forms, measures and functional capacities of human body. We used two different datasets of anthropometric surveys on US military personnel [5]. Measuring different parts of human body will cost proportionally to the number of measurements taken. The obvious question would be what measurements are most critical? The answer to this question requires expert domain knowledge. Note that this type of data is neither a

[5] https://www.openlab.psu.edu/data/



```
[1] "indus"   "lstat"   "rad"     "zn"      "ptratio" "medv"    "dis"     "nox"
```

Figure 7: Results of k-means Algorithm

Table 1: DN Construction Times for Various Variable Selection Methods on Representative Data

Dataset	m	d	Stepwise (sec.)	Forward (sec.)	stepAIC (sec.)	Lasso (sec.)
Ansur Men[1]	1174	131	4881	6039	4751	5
Ansur Women[1]	2208	131	5780	7008	7805	6
Ansur2 Men[2]	4082	93	2835	3391	3381	5
Ansur2 Women[2]	1986	93	1972	2286	1001	5
Coin I	1087	24	16	16	4	4
Coin II	926	18	8	8	3	3

[a] https://www.openlab.psu.edu/ansur/
[b] https://www.openlab.psu.edu/ansur2/

classification nor a regression problem. Similar situation may arise in case of remote sensing data: which measurements are critical? We think that DiscoVars may yield reasonable results for this kind of cases without expert domain knowledge.

In the second set of domain, crypto currency market data are collected[6]. Daily return rate of an asset, r_t can be calculated as:

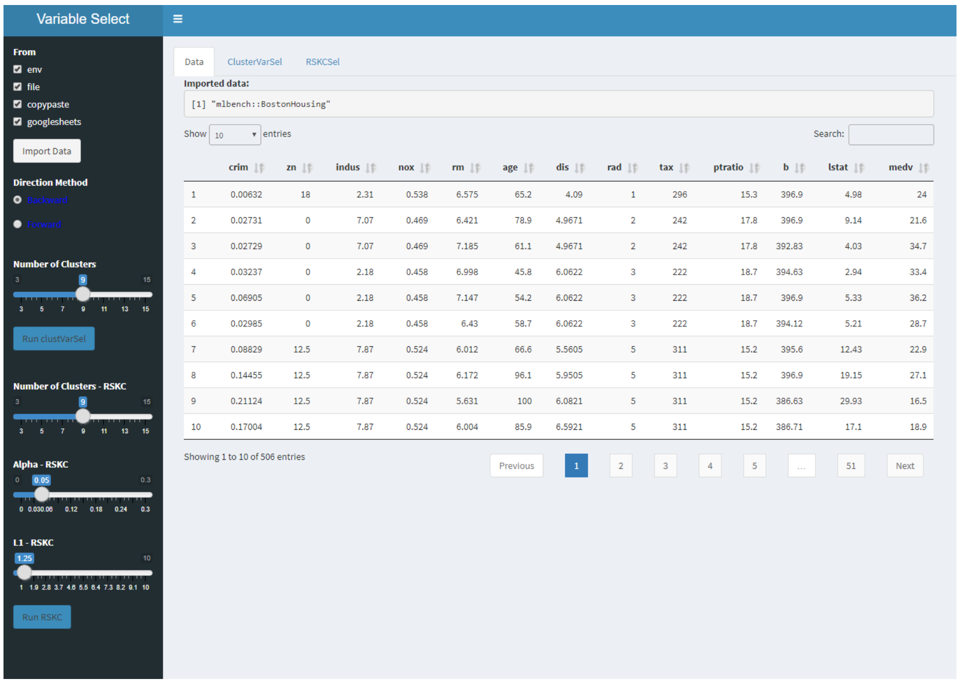
$$r_t = \frac{v_t - v_{t-1}}{v_t},$$

where v_t and v_{t-1} are daily closing values of an asset at dates t and $t-1$ respectively.

We collected data and calculated daily return rates of two different sets of crypto currencies. In the first set, eight crypto coins' daily returns, first and second lags of these returns between 10-04-2017 and 09-24-2020 are calculated. In the second set, six crypto coins' daily returns, first and second lags of these returns between 08-10-2015 and 02-20-2018 are calculated. DiscoVars can principally discover major influencers among digital coins. Dimensions of both anthropometric and crypto currency market data are given in Table 1. These datasets are also provided with the software for reproducibility purpose[7].

Table 1 also summarizes running times of variable selection methods on anthropometric and crypto currency market data. DN construction times are reported in seconds. These experiments were run on a Windows 10 machine with fourth generation i7 processor, 16GB ram and R version 3.6.1. It is apparent that Lasso has a far better performance than the remaining methods. For small datasets, all methods can interactively be used in DiscoVars. For larger datasets, users are advised to utilize Lasso. Notice that parameters of variables selection methods are not tuned extensively in our implementation. Even stepwise regression can be sped up by lowering entry and exit p values. Interested users can

[6] https://coinmarketcap.com/
[7] https://github.com/ademiriz/DiscoVars



Imported data:

```
[1] "mlbench::BostonHousing"
```

Show 10 entries Search:

	crim	zn	indus	nox	rm	age	dis	rad	tax	ptratio	b	lstat	medv
1	0.00632	18	2.31	0.538	6.575	65.2	4.09	1	296	15.3	396.9	4.98	24
2	0.02731	0	7.07	0.469	6.421	78.9	4.9671	2	242	17.8	396.9	9.14	21.6
3	0.02729	0	7.07	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03	34.7
4	0.03237	0	2.18	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	33.4
5	0.06905	0	2.18	0.458	7.147	54.2	6.0622	3	222	18.7	396.9	5.33	36.2
6	0.02985	0	2.18	0.458	6.43	58.7	6.0622	3	222	18.7	394.12	5.21	28.7
7	0.08829	12.5	7.87	0.524	6.012	66.6	5.5605	5	311	15.2	395.6	12.43	22.9
8	0.14455	12.5	7.87	0.524	6.172	96.1	5.9505	5	311	15.2	396.9	19.15	27.1
9	0.21124	12.5	7.87	0.524	5.631	100	6.0821	5	311	15.2	386.63	29.93	16.5
10	0.17004	12.5	7.87	0.524	6.004	85.9	6.5921	5	311	15.2	386.71	17.1	18.9

Showing 1 to 10 of 506 entries

Previous 1 2 3 4 5 … 51 Next

Figure 8: Tool for Unsupervised Feature Selection Methods from Literature

easily modify provided code to reflect tuning variable selection methods.

By utilizing DiscoVars on `Coin I` dataset with `eigen` centrality measure, Top-5 variables are `BNP_RTN`, `ETH_RTN_LG2`, `BNP_RTN_LG1`, `ETH_RTN_LG1`, and `BTC_RTN`. Note that crypto currency datasets also include lagged returns.

Implementation of clustvarsel and sparcl on Shiny

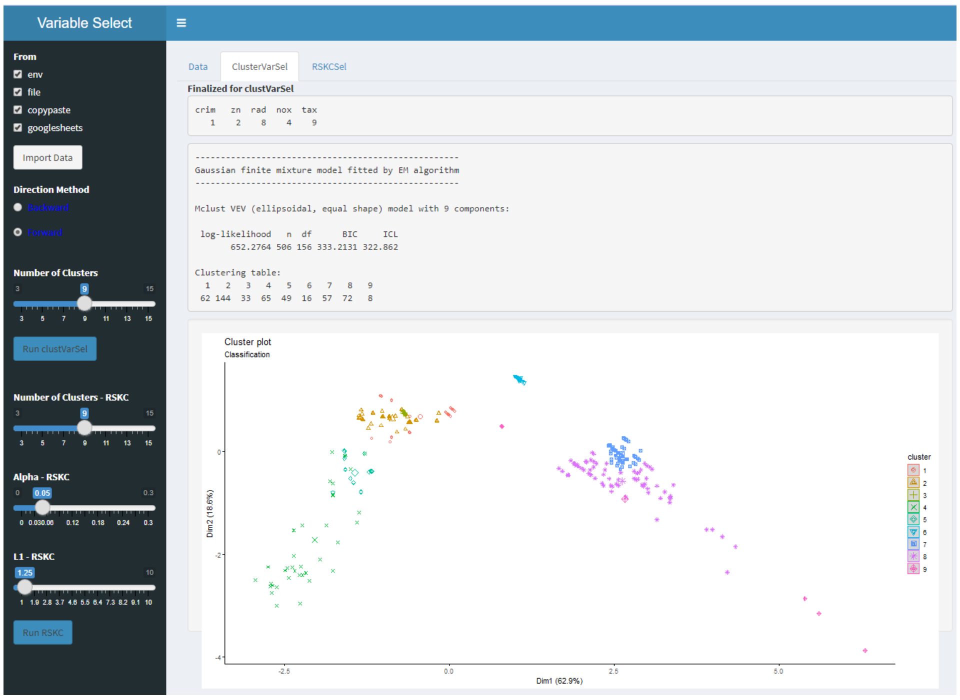
In order to have a comparison with our methodology, Shiny implementations of both **clustvarsel** Scrucca and Raftery (2018) and `sparcl` Witten and Tibshirani (2010) are also provided. Some information about these methods are already given in Section 3. We use a similar design framework for the implementations of **clustvarsel** and `sparcl`. User needs to import dataset first. Only numerical variables are included for analysis in both implementations. User is advised to remove univariate and ID variables again.

clustvarsel is based on **mclust** Scrucca et al. (2016) clustering method. Forward and backward directions i.e. variable inclusion and exclusion are available in this unsupervised feature selection method. The number of models parameter, G can also be specified. The default value is 9. As seen in Figure 8, user can pick search direction and number of clusters (models) parameters for our Shiny implementation. Once **clustvarsel** is run, the output is shown to the user in Figure 9 like in **mclust** results (see Figure 6). Note that variable selected by **clustvarsel** are shown in the text box at the top of Figure 9. Since it is a wrapper method on top of **mclust**, **clustvarsel** may take longer times for high dimensional datasets. Inherently **mclust** is slower than `k-means`.

As a second method, `sparcl` Witten and Tibshirani (2010) is implemented in Shiny via **RSKC** Package Kondo et al. (2016) in R. `sparcl` is available as stand alone R package, but **RSKC** is more preferable due to seamless integration with Shiny objects. As seen in Figure 8, user is able to choose number of clusters, α and L_1 parameters for **RSKC** in our implementation interactively. Our initial experiments have indicated that results are highly sensitive to choice of parameters.

Once **RSKC** is run, Figure 10 is shown to the user. The selected variables, standard output of **RSKC** and clustering plot (first two principal components with cluster labels) are presented. `sparcl` formulation can be coupled with hierarchical clustering methods too, but `k-means` is chosen as underlying clustering method. As user changes L_1 parameter, different variables may be chosen. For example, if we reduce L_1 to 1.15 from 1.25, we will get a sparser solution as in Figure 11. If only variables `tax` and `b` are used for clustering BostonHousing dataset, we can get well-partitioned clusters. A similar result can be achieved with the `authority` centrality measure by DiscoVars. If n is set to 2, `tax` and `rad` will be chosen (see Figure 12).

The methods provided in this section are comparison reason. User can easily compare our



From
- [✓] env
- [✓] file
- [✓] copypaste
- [✓] googlesheets

Direction Method
- Backward
- Forward

Number of Clusters: 9

Number of Clusters - RSKC: 9

Alpha - RSKC: 0.05

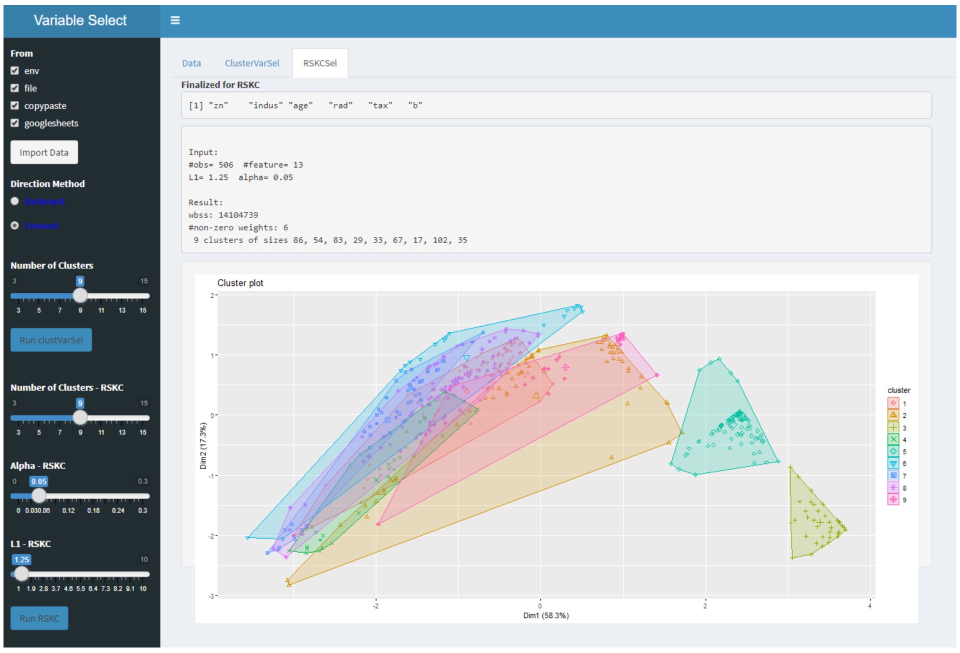
L1 - RSKC: 1.25

Finalized for clustVarSel

```
crim   zn  rad  nox  tax
   1    2    8    4    9
```

```
----------------------------------------------------
Gaussian finite mixture model fitted by EM algorithm
----------------------------------------------------

Mclust VEV (ellipsoidal, equal shape) model with 9 components:

 log-likelihood   n  df      BIC     ICL
       652.2764 506 156 333.2131 322.862

Clustering table:
  1   2   3   4   5   6   7   8   9
 62 144  33  65  49  16  57  72   8
```

Figure 9: clustvarsel Results



From
- [✓] env
- [✓] file
- [✓] copypaste
- [✓] googlesheets

Direction Method
- Backward
- Forward

Number of Clusters: 9

Number of Clusters - RSKC: 9

Alpha - RSKC: 0.05

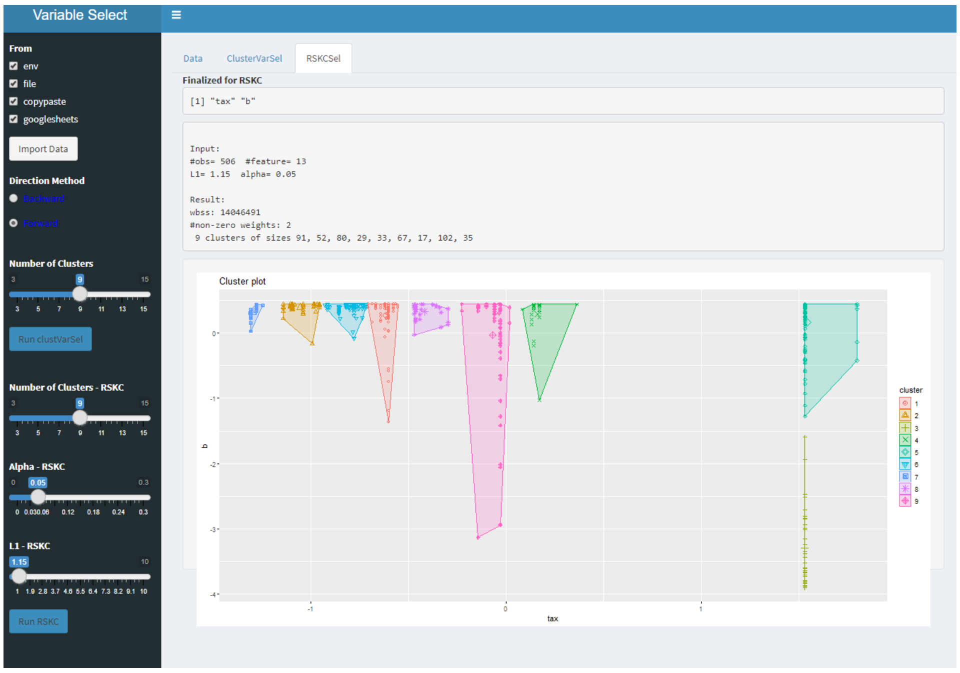
L1 - RSKC: 1.25

Finalized for RSKC

```
[1] "zn"    "indus" "age"   "rad"   "tax"   "b"
```

```
Input:
#obs= 506  #feature= 13
L1= 1.25  alpha= 0.05

Result:
wbss: 14104739
#non-zero weights: 6
 9 clusters of sizes 86, 54, 83, 29, 33, 67, 17, 102, 35
```

Figure 10: RSKC Results

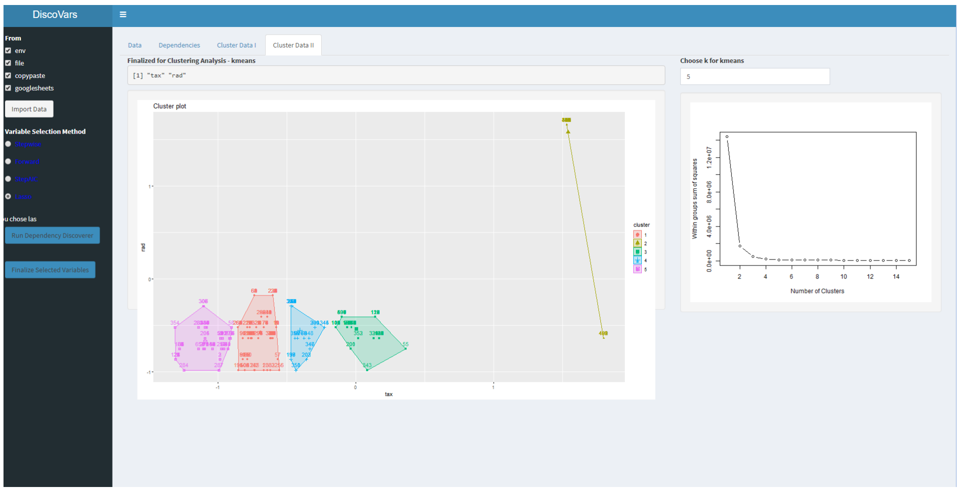


Finalized for RSKC

```
[1] "tax" "b"
```

```
Input:
#obs= 506  #feature= 13
L1= 1.15  alpha= 0.05

Result:
wbss: 14046491
#non-zero weights: 2
 9 clusters of sizes 91, 52, 80, 29, 33, 67, 17, 102, 35
```

Figure 11: Sparser RSKC Results



Finalized for Clustering Analysis - kmeans

```
[1] "tax" "rad"
```

Choose k for kmeans: 5

Figure 12: Top-2 Variable Results from DiscoVars

approach, DiscoVars with methods from literature. User can import several datasets to try these methods interactively.

Conclusions

In this paper, we presented a novel filtering method for learning algorithms. The steps of our general approach are

i- construction of a graphical model from the data set,

ii- ranking of variables based on a centrality metric calculated by using the constructed graphical model, and

iii- using the Top-*n* variables for the learning algorithm.

In the first step, we employ the well-known graphical model construction methods in the literature. Although there are studies concentrating on graphical model construction or centrality rankings (i.e. step (i) or (ii)), to the best of our knowledge, there is no paper combining both steps for the purpose of filtering for a learning algorithm. In our paper, we applied our new method to two real data sets (an anthropometric data set and a time series of crypto currency returns) and showed that our new method successfully returns satisfactory results in a reasonable computing time.

Bibliography

J. L. Andrews and P. D. McNicholas. Variable selection for clustering and classification. *Journal of Classification*, 31(2):136–153, 2014. [p4]

S. P. Borgatti. Identifying sets of key players in a social network. *Computational & Mathematical Organization Theory*, 12(1):21–34, 2006. [p3]

J. M. Carlson, Z. L. Brumme, C. M. Rousseau, C. J. Brumme, P. Matthews, C. Kadie, J. I. Mullins, B. D. Walker, P. R. Harrigan, P. J. Goulder, et al. Phylogenetic dependency networks: inferring patterns of ctl escape and codon covariation in hiv-1 gag. *PLoS computational biology*, 4(11):e1000225, 2008. [p3]

R. Caruana and V. R. de Sa. Benefitting from the variables that variable selection discards. *Journal of Machine Learning Research*, 3:1245–1264, 2003. [p1]

G. Chandrashekar and F. Sahin. A survey on feature selection methods. *Journal of Computers and Electrical Engineering*, 40:16–28, 2014. ISSN 0045-7906. doi: 10.1016/j.compeleceng.2013.11.024. [p2]

D. M. Chickering, D. Heckerman, and C. Meek. A bayesian approach to learning bayesian networks with local structure. In D. Geiger and P. P. Shenoy, editors, *UAI '97: Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence*, pages 80–89, Brown University, Providence, Rhode Island, USA, Aug. 1997. Morgan Kaufmann. [p2]

D. Davies and D. Bouldin. "A Cluster Separation Measure". *IEEE Transcations on Pattern Analysis and Machine Intelligence*, 1(2):224–227, 1979. [p4]

A. Demiriz. Solving traveling salesman problem by ranking. In M. van Zaanen, J. Stehouwer, and M. van Erp, editors, *The 18th Annual Belgian-Dutch Conference on Machine Learning*, Tilburg University, May 2009. [p4]

A. Dobra. Variable selection and dependency networks for genomewide data. *Biostatistics*, 10(4): 621–639, 2009. [p3]

M. Fop and T. B. Murphy. Variable selection methods for model-based clustering. *Statistics Surveys*, 12: 18–65, 2018. [p4]

G. Forman. An extensive empirical study of feature selection metrics for text classification. *Journal of Machine Learning Research*, 3:1289–1305, 2003. [p1]

I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003. [p1]

D. L. Hansen, B. Shneiderman, M. A. Smith, and I. Himelboim. Chapter 6 - calculating and visualizing network metrics. In D. L. Hansen, B. Shneiderman, M. A. Smith, and I. Himelboim, editors, *Analyzing Social Media Networks with NodeXL (Second Edition)*, pages 79–94. Morgan Kaufmann, second edition edition, 2020. ISBN 978-0-12-817756-3. doi: https://doi.org/10.1016/B978-0-12-817756-3.00006-6. [p2]

X. He, D. Cai, and P. Niyogi. Laplacian score for feature selection. In *Advances in Neural Information Processing Systems 18 [Neural Information Processing Systems, NIPS 2005, December 5-8, 2005, Vancouver, British Columbia, Canada]*, pages 507–514, 2005. [p4]

D. Heckerman, D. M. Chickering, C. Meek, R. Rounthwaite, and C. Kadie. Dependency networks for inference, collaborative filtering, and data visualization. *Journal of Machine Learning Research*, 1(Oct): 49–75, 2000. [p2, 3]

L. Hubert and P. Arabie. Comparing partitions. *Journal of Classification*, 2:193–218, 1985. [p4]

D. Koller, N. Friedman, L. Getoor, and B. Taskar. *Introduction to Statistical Relational Learning*, chapter Graphical models in a nutshell, pages 13–55. MIT press, Cambridge, MA, 2007. [p2]

Y. Kondo, M. Salibian-Barrera, and R. Zamar. Rskc: An r package for a robust and sparse k-means clustering algorithm. *Journal of Statistical Software*, 72(5):1–26, 2016. doi: 10.18637/jss.v072.i05. [p9]

J. Li, K. Cheng, S. Wang, F. Morstatter, R. P. Trevino, J. Tang, and H. Liu. Feature selection: A data perspective. *ACM Computing Surveys*, 50(6):1–45, 2018. doi: 10.1145/3136625. URL https://doi.org/10.1145/3136625. [p1, 2]

F. Martinez-Plumed, L. Contreras-Ochando, C. Ferri, J. Hernandez-Orallo, M. Kull, N. Lachiche, M. J. Ramirez-Quintana, and P. Flach. Crisp-dm twenty years later: From data mining processes to data science trajectories. *IEEE Transactions on Knowledge and Data Engineering*, 33:3048–3061, 2021. ISSN 2326-3865. doi: 10.1109/TKDE.2019.2962680. [p1]

N. Meinshausen and P. Bühlmann. High-dimensional graphs and variable selection with the lasso. *The annals of statistics*, 34(3):1436–1462, 2006. [p3, 4]

A. E. Raftery and N. Dean. Variable selection for model-based clustering. *Journal of the American Statistical Association*, 101:473:168–178, 2006. ISSN 0162-1459. doi: 10.1198/016214506000000113. [p4]

T. M. Roddenberry and S. Segarra. Blind inference of centrality rankings from graph signals. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5335–5339. IEEE, 2020. [p3]

T. M. Roddenberry and S. Segarra. Blind inference of eigenvector centrality rankings. *IEEE Transactions on Signal Processing*, 69:3935–3946, 2021. [p3]

L. Scrucca and A. E. Raftery. clustvarsel: A package implementing variable selection for gaussian model-based clustering in r. *Journal of Statistical Software*, 84, 2018. [p4, 9]

L. Scrucca, M. Fop, T. Murphy, and A. Raftery. mclust 5: Clustering, classification and density estimation using gaussian finite mixture models. *The R Journal*, 8(1):289–317, 08 2016. doi: 10.32614/RJ-2016-021. [p4, 7, 9]

W. Sun, C. Chang, and Q. Long. Joint bayesian variable selection and graph estimation for non-linear svm with application to genomics data. In *2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA)*, pages 315–323, 2020. doi: 10.1109/DSAA49011.2020.00045. [p3]

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996. ISSN 00359246. URL http://www.jstor.org/stable/2346178. [p2, 3]

A. Truong, A. Walters, J. Goodsitt, K. Hines, C. B. Bruss, and R. Farivar. Towards automated machine learning: Evaluation and comparison of automl approaches and tools. In *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 1471–1479, 2019. doi: 10.1109/ICTAI.2019.00209. [p1]

D. M. Witten and R. Tibshirani. A framework for feature selection in clustering. *Journal of the American Statistical Association*, 105(490):713–726, 2010. doi: 10.1198/jasa.2010.tm09415. PMID: 20811510. [p4, 9]

Ayhan Demiriz
Verikar Software
Pendik, Istanbul, 34912
Turkey
(ORCiD 0000-0002-5731-3134)
ademiriz@gmail.com