FeatureMiner: A Tool for Interactive Feature Selection

Kewei Cheng, Jundong Li and Huan Liu
Computer Science and Engineering,
Arizona State University, Tempe, AZ 85281, USA
{Kewei.Cheng, jundong.li, huan.liu}@asu.edu

ABSTRACT

The recent popularity of big data has brought immense quantities of high-dimensional data, which presents challenges to traditional data mining tasks due to curse of dimensionality. Feature selection has shown to be effective to prepare these high dimensional data for a variety of learning tasks. To provide easy access to feature selection algorithms, we provide an interactive feature selection tool *FeatureMiner* based on our recently released feature selection repository *scikit-feature*[1]. *FeatureMiner* eases the process of performing feature selection for practitioners by providing an interactive user interface. Meanwhile, it also gives users some practical guidance in finding a suitable feature selection algorithm among many given a specific dataset. In this demonstration, we show (1) How to conduct data preprocessing after loading a dataset; (2) How to apply feature selection algorithms; (3) How to choose a suitable algorithm by visualized performance evaluation.

Keywords

Data Mining; Feature Selection; Interactive User Interface

1. INTRODUCTION

We are now in the era of big data, massive amounts of high-dimensional data are generated at an unprecedented rate. For example, 2.5 quintillion bytes of data are created every day and 90% of data in the world today was produced within the last two years[2]. These massive amount of high-dimensional data (e.g., social media data, images, videos) presents challenges to data mining techniques due to the curse of dimensionality. As a traditional and effective approach to prepare high-dimensional data, feature selection [5] aims to select a subset of relevant features to build a comprehensible learning model with good predictive performance.

In the past few decades, a large number of feature selection algorithms have been developed [3, 4]. These algorithms are designed to serve different purposes and have their own advantages and disadvantages. With so many feature selection algorithms in literature, there is a urgent need for a system to collect some widely used feature selection algorithms that have been developed to serve as a platform for facilitating their application, comparison and joint study [8, 9]. Recently, we release a open-source feature selection repository *scikit-feature* which includes around 40 representative feature selection algorithms. However, this repository is initially built for experienced users and programmers with some background knowledge of feature selection. It would be better for users to have some prior knowledge about these algorithms since different feature selection algorithms may require different inputs and provide different outputs. In this work, in order to provide easy access to these feature selection algorithms, we present a system *FeatureMiner*, which facilitates the use of these feature selection algorithms in practical usage by providing an interactive user interface.

2. FEATURE SELECTION

According to the availability of label information, feature selection algorithms can be either supervised [2] or unsupervised [1]. Supervised feature selection algorithms take advantage of the class labels to evaluate feature relevance by its ability to distinguish instances from different classes. A general framework of supervised feature selection is illustrated in Figure 1. It consists of two phases - the training phase and the prediction phase. In the training phase, supervised feature selection algorithms first select subset of highly discriminant features. With training data represented by these selected features, classifiers are trained under the guidance of label information. In the prediction phase, the trained classifier predicts class labels of testing data on the selected features.

However, due to the lack of label information, unsupervised feature selection exploits alternative criteria instead of class labels to define the relevance of features which include data similarity, local discriminative information and data reconstruction error. A general framework of unsupervised feature selection is illustrated in Figure 2. Different from supervised feature selection, unsupervised feature selection usually uses all instances in the feature selection phase then outputs the cluster structure of all data samples on the selected features by using a typical clustering algorithm.

[1]http://featureselection.asu.edu/
[2]http://www-01.ibm.com/software/data/bigdata/what-is-big-data.html

CIKM'16 October 24-28, 2016, Indianapolis, IN, USA

ACM ISBN 978-1-4503-4073-1/16/10.

DOI: http://dx.doi.org/10.1145/2983323.2983329

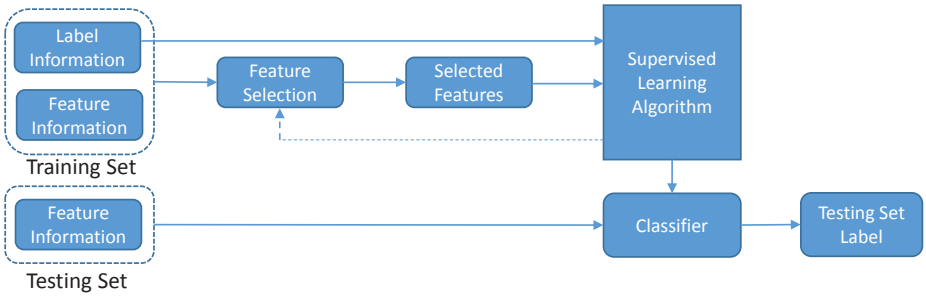

Figure 1: A general framework of supervised feature selection.

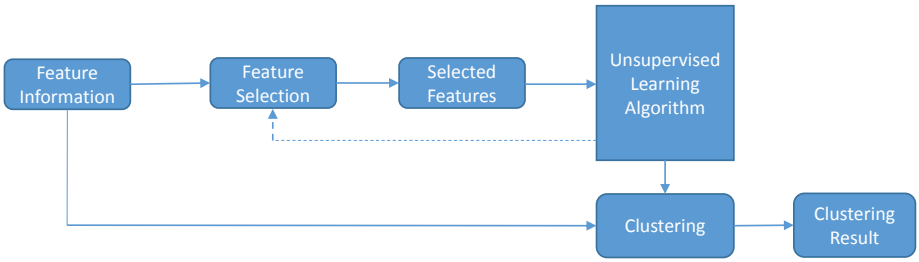

Figure 2: A general framework of unsupervised feature selection.

3. SYSTEM DESIGN

We develop the interactive feature selection tool *FeatureMiner* on the basis of our recently released feature selection repository *scikit-feature*, which consists of most the popular supervised and unsupervised feature selection algorithms. The tool is designed to simplify the usage of these feature selection algorithms for users without much experience of feature selection. The workflow of *FeatureMiner* is shown in Figure 3. As shown in Figure 3, *FeatureMiner* is a hierarchically-organized tool, with the low-level procedures at the bottom of the hierarchy as loading data and conducting data preprocessing. These low-level procedures are assembled into higher-level feature selection algorithms. As the highest-level procedures, the selected features are used to build a learning model for performance evaluation. There are three tabs in *FeatureMiner*, with the first allowing data preprocessing, the second providing access to supervised feature selection algorithms with performance evaluation and the third is designed for unsupervised feature selection and its performance evaluation. In the following subsections, we will introduce our interactive tool in detail. For more information on our tool, please visit the demo webpage[3].

3.1 Data Preprocessing

As shown in Figure 4, on the *preprocess* tab, users can obtain some basic information of dataset and prepare the dataset for certain types of feature selection algorithms. It currently support two kinds of formats: Matlab data format and CSV file format. Once the dataset has been loaded, some basic information of the dataset is presented to users, including the number of instances, the number of features, and the number of classes. To give users an overview of data distribution, *FeatureMiner* also offers a bar graph to show the class distribution of the dataset. It also supports other data preprocessing. Functions supported in data preprocessor are shown as follows:

(1) data normalization: It transforms features by scaling each feature to the range of $(0, 1)$.

(2) data discretization: With the number of bins as input, it partitions continuous values of features into a smaller number of bins.

3.2 Performing Feature Selection

As discussed in Section 2, unsupervised feature selection algorithms are distinct with supervised feature selection algorithms in both algorithm design and performance evaluation. Thus, we separate supervised feature selection and unsupervised feature selection into two tabs, *supervised* tab as shown Figure 5 and *unsupervised* tab as shown Figure 6.

For supervised feature selection, users can select a supervised feature selection algorithm from drop down list with its parameters and number of selected features as input. It should be noted that different feature selection algorithms may have different requirements for datasets. For example, some supervised feature selection algorithms can only tackle binary-class classification problems while others can tackle multi-class classification problems. Some methods can only deal with discrete feature values while others can also deal with continuous feature values. To ensure that the dataset is well prepared for a chosen algorithm, *FeatureMiner* checks dataset before conducting the feature selection. If the dataset is not well prepared, it reminds users to conduct corresponding preprocessing on the dataset first. As mentioned above, it is a difficult task to choose a suitable feature selection algorithm for a particular application [6]. To give users further guidance, we divide supervised feature selection algorithms into five categories: (1) information theoretical based methods [7] are proposed to maximize feature rele-

[3]http://featureselection.asu.edu/featureminer.php

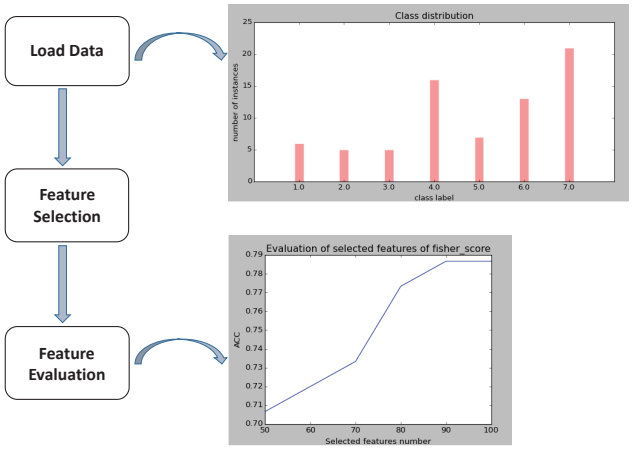

Figure 3: Overview of the *FeatureMiner* for interactive feature selection.

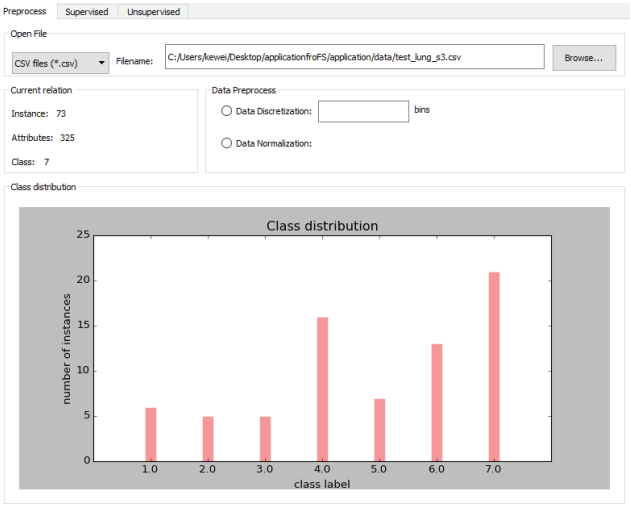

Figure 4: The preprocess tab of *FeatureMiner* showing example of loading lung cancer dataset.

vance and minimize feature redundancy (i.e., CIFE, CMIM, DISR, FCBF, ICAP, IGFS, JMI, MIFS, MIM, MRMR); (2) similarity based methods select features based on their ability to preserve data similarity (i.e., Fisher Score, ReliefF, Trace Ratio); (3) sparse learning based methods aim to minimize the fitting error with sparse regularization terms (i.e., RFS, Least square loss, Logistic loss); (4) statistical based methods evaluate importance of features based on different statistical measures (i.e., T-score, F-score, Chi-square, Gini Index) and (5) wrapper methods measure features by their predictive accuracy of a predetermined learning algorithm (i.e., Decision Tree based backward elimination, Decision Tree based forward selection, SVM based backward elimination, SVM based forward selection). To choose a suitable algorithm for a loaded dataset, user can first find a best category and then conduct deeper analysis with visualized performance evaluation.

Similar to supervised feature selection, to conduct unsupervised feature selection, users first choose an unsupervised feature selection algorithm from drop down list then specify its parameters and the number of selected features. Feature selection methods supported in *unsupervised* tab include: (1) similarity based methods (i.e., SPEC, Laplacian Score); and (2) sparse learning based methods (i.e., MCFS, NDFS, UDFS).

3.3 Evaluation

After we obtain the selected features as illustrated in Section 3.2, we evaluate these feature selection algorithms. There are some differences between evaluation of supervised and unsupervised feature selection algorithms.

As shown in Figure 1, to test performance of the supervised feature selection algorithms, the whole dataset is usually divided into two parts including the training set and testing set. With guidance of label information, feature selection algorithms will be first applied on the training set to select a subset of relevant features; then the testing set on the selected features are regarded as input to a classification model for the performance evaluation purpose. The predicted labels are compared with the label information of testing set by different classification evaluation metrics. In the system, the evaluation results will be visually plotted, which enables users to have a clear sense about how classification performance vary with respect to the number of selected features. The visual plots are very useful since users can determine the optimal value for the number of selected features. Besides, for the request of users, the system can also save the evaluation results to a file with the classification performance and the indexes of selected features.

A user can specify evaluation setting for supervised feature selection algorithms including: (1) classification models (e.g., Linear SVM, Nearest Neighbors, Decision Tree); (2) number of runs for the classification evaluation; (3) percentage of training/test data; and (4) classification evaluation metrics (e.g., classification accuracy and F1 score).

As shown in Figure 2, for performance evaluation of unsupervised feature selection algorithms, there is no need to divide the whole dataset into the training set and testing set. The label information(ground truth) is only used for the evaluation purpose. Each feature selection algorithm is first applied on the whole dataset to select features, then clustering models are performed based on the selected features. We repeat the clustering algorithm 10 times and report the average clustering results visually in the same way as the line chart for performance evaluation of supervised feature selection. Similarly, it can also save results to a file including the clustering performance and the indexes of selected features for the request of users.

A user can specify evaluation setting for unsupervised feature selection including: (1) clustering models (e.g., K-Means, spectral clustering); and (2) clustering evaluation metrics(e.g., clustering accuracy (ACC), normalized mutual information (NMI) and adjusted rand Index (ARI)).

4. DEMO SCENARIOS

During the demonstration, we will apply several feature selection algorithms to different kinds of datasets with various types of settings and show users: (1) How to conduct

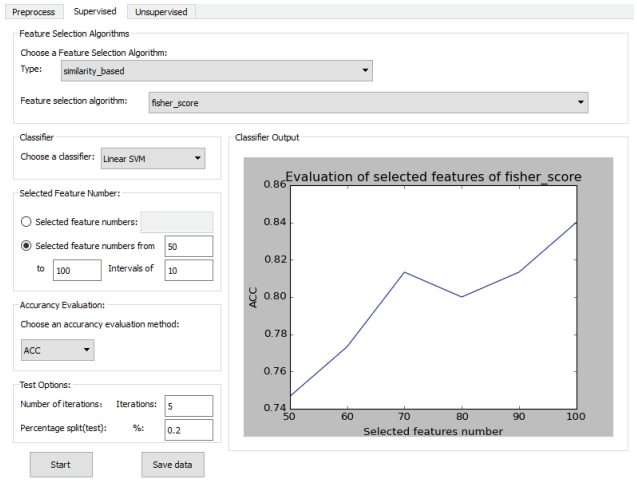

Figure 5: The supervised tab of *FeatureMiner* showing example of conducting fisher score algorithm.

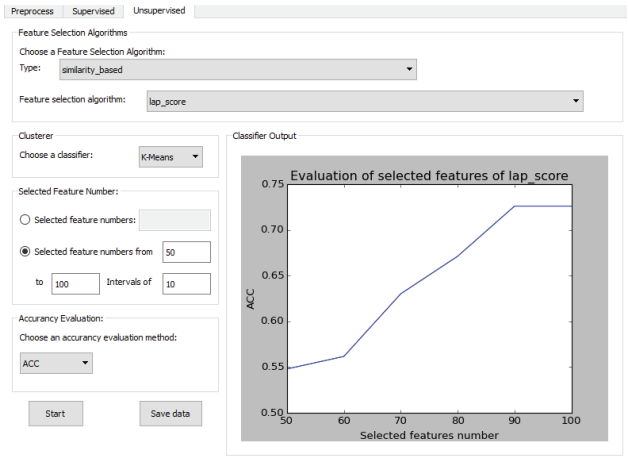

Figure 6: The unsupervised tab of *FeatureMiner* showing example of conducting laplacian score algorithm.

data preprocessing after loading a dataset; (2) How to apply feature selection algorithms; (3) How to choose a suitable algorithm by visualized performance evaluation. We have an example to demonstrate how to find a suitable feature selection algorithm for lung cancer dataset[4]. To find the most suitable supervised feature selection algorithm for this dataset, first we randomly choose an algorithm from each of the five categories mentioned in Section 3.2 and set the number of selected feature to the same range then apply these algorithms to the lung cancer dataset. After comparing the classification performance of these algorithms, the category of that shows the best classification performance is considered as the best algorithm category for this dataset. Then we apply each algorithms in this category to lung cancer dataset in order to find out the most suitable feature selection algorithm. Similarly, we use the same method to find a most suitable unsupervised feature selection algorithm for this dataset.

5. CONCLUSIONS AND FUTURE WORK

In this work we have built a user interactive feature selection tool *FeatureMiner* which provides easy access to different feature selection algorithms. The tool is built upon our recently released open-source feature selection repository *scikit-feature*. It facilitates the use of feature selection algorithms for users with limited background knowledge and gives them some practical guidance in finding a suitable feature selection algorithm among many given a specific dataset.

In the future, to expand the functionality of *FeatureMiner*, we would like to automatically recommend the most suitable algorithms to users based on the experimental results of similar datasets. Besides, *FeatureMiner* currently only provides users access to traditional feature selection algorithms. Due to the prevalence of heterogeneous data sources such as social media data, we would like to integrate some feature selection algorithms for heterogeneous data into our system.

[4]http://penglab.janelia.org/proj/mRMR/index.htm#data

Acknowledgments

This material is, in part, supported by the National Science Foundation (NSF) under grant number IIS-1217466.

6. REFERENCES

[1] S. Alelyani, J. Tang, and H. Liu. Feature selection for clustering: A review. *Data Clustering: Algorithms and Applications*, 29, 2013.

[2] M. Dash and H. Liu. Feature selection for classification. *Intelligent data analysis*, 1(3):131–156, 1997.

[3] J. Li, K. Cheng, S. Wang, F. Morstatter, R. Trevino, J. Tang, and H. Liu. Feature selection: A data perspective. 2016.

[4] J. Li and H. Liu. Challenges of feature selection for big data analytics. *IEEE Intelligent Systems*, 2016.

[5] H. Liu and H. Motoda. *Computational methods of feature selection*. CRC Press, 2007.

[6] H. Liu and L. Yu. Toward integrating feature selection algorithms for classification and clustering. *Knowledge and Data Engineering, IEEE Transactions on*, 17(4):491–502, 2005.

[7] H. Peng, F. Long, and C. Ding. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on pattern analysis and machine intelligence*, 27(8):1226–1238, 2005.

[8] P. Somol and P. Pudil. Feature selection toolbox. *Pattern Recognition*, 35(12):2749–2759, 2002.

[9] Z. Zhao, F. Morstatter, S. Sharma, S. Alelyani, A. Anand, and H. Liu. Advancing feature selection research. *ASU feature selection repository*, pages 1–28, 2010.