Chameleon: A Python Workflow Toolkit for Feature Selection

Diviya Thilakeswaran[1], Simon McManis[2], and X. Rosalind Wang[1,3][0000−0001−5454−6197]⋆

[1] Western Sydney University, Parramatta, NSW, Australia
diviya.thilak@outlook.com
rosalind.wang@westernsydney.edu.au
[2] University of Technology, Sydney
simon.mcmanis@gmail.com
[3] CSIRO Data61, Epping, Australia

Abstract. When considering classification problems in relation to high-dimensional data sets, such as biological data sets, the need for effective methods of dimensionality reduction by feature selection becomes apparent. Feature selection has been shown to significantly decrease computational cost and allow for classification models that are more easily interpretable. We present *Chameleon*, a Python-based toolkit that integrates all steps in a feature selection evaluation pipeline – from splitting data for cross-validation, to visualisation of classification results using various metrics. We implemented in *Chameleon* six existing feature selection methods, six common classification methods, and the classification results are evaluated using two different metrics. We also implemented an ensemble method which selects only common features from the different methods evaluated. Experimental results using four different data sets suggest that the common features method achieves improved or similar classification performance, compared to the individual feature selection algorithms, using smaller and thus more computationally efficient subsets of features.

Keywords: feature selection · biological data · classification

1 Introduction

The exponential growth in the volumes of data being generated in the world today has given rise to many data sets with extremely high dimensionality, posing the challenge of how to effectively extract the most relevant and valuable information from them. The application of data mining and machine learning techniques in order to uncover meaningful information from a data set becomes more complex as its dimensionality grows. A great increase in the number of features may lead to a significant amount of noise in the data, as well as the

⋆ Corresponding Author

over-fitting and poor performance of learning models, in what is known as the curse of dimensionality [13].

Yu and Liu [26] describe features as belonging to one of four categories; completely irrelevant features, weakly relevant and redundant features, weakly relevant and non-redundant features and strongly relevant features. Relevant features may have significant correlation with the predictors, and as such should be included in model building in order to increase prediction accuracy. Redundant features may correlate significantly with each other, and thus should be eliminated in order to improve model performance.

In order to simplify and improve the accuracy of classification models that are built upon these high-dimensional data sets, methods that can account for such irrelevant and redundant features in the data must be considered. This accounting for noise in a data set can be achieved by selecting a subset of features that are determined to be most relevant to the classification model being constructed. This effectively reduces the dimensionality of the data in a technique known as feature selection.

The importance of feature selection as a method of dimensionality reduction can be seen through its application in bioinformatics and medicine. In order to provide a more accurate understanding of underlying processes in such fields it may be imperative to retain the interactions of the original features [15]. Methods of feature selection have been shown to be effective in various applications, including improving classification performance with medical data [24].

1.1 Feature Selection for Classification

One of the most common applications for feature selection methods includes its use in classification problems [8,11]. Real-world data sets often contain a high multitude of different features, many of which are not relevant to the classification problem that is attempting to be solved. Training classifiers on high dimensional data sets with large amounts of irrelevant and redundant features is costly and unwieldy. Through the selection of an optimal subset of features, the subsequent process of building a classifier on these features becomes more cost efficient, and the resulting model is more generalisable.

Feature selection techniques for the purposes of classification can be categorised into two main categories: filter model and wrapper model. Filter models determine feature subsets based upon the general and intrinsic characteristics of the data [7]. Such methods are known to computationally efficient and are separate from classifier learning algorithms, thus they are able to be applied to many classifiers with better generalisation. The process of selecting feature subsets using filter methods is independent of learning algorithms, thus the selected subset may not provide the most optimal performance for the target classification model.

Wrapper models utilise specific learning algorithm and select optimal subsets based on classifier's prediction error [6]. As such, wrapper models are able to drive better classification performance compared to filter methods, though this comes at significant computational cost. Unlike filter methods, wrapper methods

are less generalisable to learning algorithms other than the one used to determine their evaluation criterion, and are more likely to over-fit.

1.2 Python Toolkit for Feature Selection and Classification

Currently, there are few existing open source Python libraries for feature selection. The Python *Scikit-Learn* library is an extremely popular machine learning library, however the feature selection module contained within the library implements only a few methods [22]. Currently, the largest feature selection library for Python to exist is *Scikit-Feature* developed by Arizona State University, which contains around 40 algorithms for feature selection [20]. However, *Scikit-Feature* is implemented in Python 2, which is no longer under development.

Despite the existence of a few Python tools which contain various feature selection algorithms, there does not evidently and currently exist a Python toolkit which is able to implement feature selection methods along with subsequent classification results and metrics to evaluate the efficacy of these algorithms.

1.3 Contribution

We present a Python toolkit named *Chameleon*, which streamlines the process of feature selection for classification and evaluates the classification performances. Our toolkit implements six existing feature selection methods and an ensemble method of using features common to the methods evaluated. Six classifiers were also implemented, along with two classification performance metrics.

We evaluated the methods using four biological data sets and showed that although the classification performance varies according to data sets, feature selection and classification algorithms, the common features selected in general give better classification performance with less number of features.

2 Methods

2.1 Chameleon Toolkit Structure

Chameleon requires the installation of Python 3.6+ as well as Java. Currently, data sets to be used with the package must contain features and targets with either `.mat` or `.arff` formats. The chosen data set is first imported with the command `chameleon-data`. This command contains two sub-commands `format` and k`-fold`, which formats the data and partitions it into k folds to make k train and test data sets respectively. The value of k is set to 5, data normalised and the k-fold method is stratified by default, however these parameters may be adjusted by the user.

The full pipeline (Figure 1) may then be configured using `chameleon-pipe`, whereby all feature selection methods and classifiers are added. Specific feature selection methods and classifiers may also be chosen and added individually. Finally, `chameleon-run` is used to run the configured pipeline. Feature selection

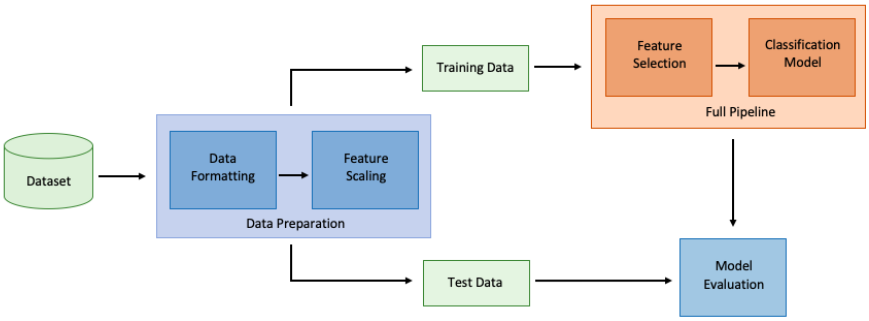


Fig. 1: General structure of Chameleon python toolkit.

methods are applied to the training data and the number of determined important features `n_features`, set to 50 by default, are used to build classification models. Subsequently, the test data is applied to the classification models in order to predict class labels, and classification evaluation metrics such as classification accuracy are able to be determined.

2.2 Feature Selection Methods

We included in *Chameleon* six feature selection algorithms and also implemented an ensemble method for feature selection:

Fisher Score The aim of the Fisher Score (FS) feature selection algorithm is to select a subset of features such that the Euclidean distances between data points in differing classes are maximised and the distances between data points within classes are minimised [24]. The FS for each feature is calculated and the features are ranked according to their scores, with these highly ranked features being selected. Features are evaluated independent of each other, thus the FS method is less effective in handling feature redundancy as it does not evaluate combinations of features. Despite this, it has been shown to be effective in various applications, such as improving classification performance with medical data.

ReliefF In the Relief feature selection algorithm, a feature statistic is calculated in order to help determine that feature's relevance to the classification problem [18]. The ReliefF algorithm improves the original Relief algorithm by iterating through a given number of training instances, and the feature statistics or scores are determined by feature value differences between nearest neighbour instances [19]. Thus, the Relief algorithm is able to determine relevant features in the context of others and take interactions between features into account.

Random Forest Feature Importance (RFFI) Random Forest (RF) classifiers train multiple decision trees on sub-samples, which are then aggregated for prediction. Such a classifier has shown to be effectively utilised as a method of feature selection [9]. When used in feature selection, Random Forests (RFs) were iteratively fitted to the data, with the least important features being discarded upon the creation of a new forest with each iteration. A RF feature importance algorithm is effective in obtaining small subsets of important features while maintaining high prediction accuracy.

SVM-Recursive Feature Elimination (SVM-RFE) Support Vector Machine (SVM) classifiers has been used with Recursive Feature Elimination (RFE) in order to select a small subset of features [16]. Here, a SVM classifier is used to train a model, where a decision boundary is determined in order to maximise the margin between classes, allowing for the weights of all the features to be determined accordingly. The algorithm then iteratively removes features with the smallest weights until a subset of relevant and non-redundant features remain, allowing for greater accuracy in the subsequent classification performance.

Iterative Mutual Information Selection In the case of classification problems, Mutual Information (MI) refers to the amount that the knowledge provided by features that reduces uncertainty about the class. An approach to feature selection proposed by Battiti [3] considers the use of MI in order to evaluate features based on their relation to previously selected features and the class. The aim is to maximise the MI between the selected subset of features and target class, thus providing a subset which contains the most relevant and non-redundant information of the class. With a large set of features, such an algorithm can provide a significantly larger set of possible feature combinations to evaluate, increasing computational cost significantly. In iterative MI method, we select the subsequent feature based on the features already selected and the conditional MI score. A significance test is also run to ensure the new feature does add significantly to the conditional MI value, otherwise a feature with the highest MI value will be added.

Simple Mutual Information Score In order to select a subset of optimal features using mutual information at a significantly lower computation cost than Battiti's algorithm, a simplified approach has been proposed in which information provided by features about the class are considered independent of each other [25]. The selected features are those with only the greatest amount mutual information with the class, taking into account features relevant to the classification problem but not allowing for the consideration of redundant features. As such, the results provided by the algorithm may be generally less accurate than alternative MI approaches, however such an approach benefits from significantly lowered computational expense.

Algorithm 1: Common Features Selection Method

input : Set of feature ranking arrays S
Size of feature subsets m
output: Common features subset list f

$R \longleftarrow \emptyset$;
for $u \in S$ **do**
  $R \longleftarrow R \cup \{u[:m]\}$
$f = R_1 \cap \ldots \cap R_n$

2.3 Feature Selection with Common Features

It has been shown that ensemble methods can improve individual feature selection outcomes [4]. We therefore implemented in *Chameleon* an approach to select an optimal subset of features that are the common features among subsets selected by several high performing feature selection algorithms. The high performing feature selection algorithms are identified by users from classification results of individual feature selection methods. The proposed approach to selecting subsets of features common to various high-performing feature selection methods is described in Algorithm 1. Feature subsets of size m are determined from lists of features ranked by importance in descending order for each feature selection algorithm. The intersection between these subsets is found, resulting in a reduced subset of highly important features.

2.4 Classification Methods

Six common classification methods were implemented in *Chameleon*.

Gaussian Naive Bayes The Gaussian Naïve Bayes (GNB) classifier is an algorithm which is based on Bayes' rule under the assumption of a Gaussian distribution in order to classify data [27]. Naïve Bayes (NB) algorithms operate under an assumption of conditional independence, which determines features to be independent of each other given the class variable.

K-Nearest Neighbours K-nearest neighbours (KNN) classification allows for samples to be classified according to the class of their k-nearest neighbours [2]. The default values for *Scikit-Learn* Library's [22] KNN classifier were implemented, including a K value of 5 and Euclidean distance to measure the distance between points.

Logistic Regression Logistic regression is a linear classifier that make use of the sigmoid function in order to classify data with binary target variables [10].

Logistic regression classifiers models the output probability that the input belongs to a certain class, transformed through the use of a logistic sigmoid function. The classifier allows for highly efficient training and high accuracy for linearly separable data sets. A significant limitation of the method is its assumption that the features and target variable share a linear relationship, as well as its tendency to over-fit when used with data sets containing a greater number of features than observations.

Neural Network (Multi-layer Perceptron) Multi-layer Perceptron (MLP) classifiers implement an underlying Neural Network (NN) in order to effectively classify data [23]. The parameters of the MLP classifier used include a limited-memory Broyden-Fletcher-Goldfarb-Shanno (lbfgs) solver to optimise the log-loss function, as well as hidden layer sizes of 100.

Random Forest The RF algorithm builds a classifier using multiple decision tree classifiers [5]. The decision trees are trained using various random subsamples and the results are subsequently aggregated for prediction of classes. The random forest algorithm uses bootstrap aggregation as well as random variable selection in order to build trees.

Support Vector Machine SVM algorithms work to classify data by finding an optimal separating hyperplane between classes. A regularisation parameter C is determined, a lower value ensuring that the distance of groups from the margin is maximised, while a higher C value minimises the amount of misclassifications. The default values for *Scikit-Learn* Library's SVM classifier were used, including a C value of 1, and a Radial basis function (RBF) kernel in order to account for non-linearity that may be found in real-world data sets.

2.5 Performance Metrics

Two performance metrices were implemented in *Chameleon*: accuracy and Area Under the ROC Curve (AUC). Classification accuracy is the proportion of correct results to the total number of instances. AUC is a value between 0 and 1 that give indication on how effectively the classifier is able to separate the data into differing classes. Statistical significance levels for classification are also calculated through either a t-test for binary classification, or randomly permuting the test data labels for multi-class classification.

2.6 Data Sets

Four high-dimensional microarray gene expression data sets (Table 1) were used to evaluate the feature selection methods. The data sets represent both discrete and continuous features, as well as binary and multiple class targets.

Table 1: Details of chosen high-dimensional gene expression data sets

Data Set	Instances	Features	Feature Type	Classes	Reference
Colon Cancer	62	2,000	Discrete	2	[1]
Leukemia Cancer	72	7,129	Continuous	2	[14]
Glioma Tumour	85	22,283	Continuous	2	[12]
Lung Cancer	203	3,312	Continuous	5	[17]

2.7 Feature Selection Evaluation

Stratified 10-fold cross-validation is implemented for all data sets, partitioning the data into training and test sets, whilst preserving the proportions of class labels for each fold. Measures of classification accuracy for the six toolkit feature selection methods are computed, with models built using feature subsets with sizes ranging from 1 to 100 of the most important ranked features.

Classification accuracy of the common features selection method is determined by taking the common features apparent from feature subsets of multiple high-performing feature selection methods. The size of the feature subsets from which the common features are selected range from 10 to 100 included features, increasing with increments of 5.

We also evaluated the classification performance over the whole data set to provide a baseline for performance of selected features.

2.8 Software

This work uses the Python packages *Scikit-Feature* for four of the feature selection methods (RF, ReliefF, RFFI and SVM-RFE) [20] and *Scikit-Learn* for classifiers. The MI feature selection methods used the JIDT package [21] for MI calculations, and Python wrapper code is written to call methods in the package. The ensemble methods were developed and implemented by us in Python. The code used in this work is on GitHub[4].

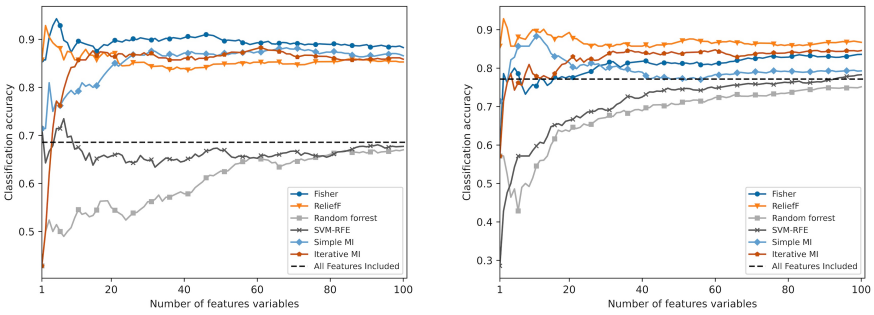
3 Results and Discussion

3.1 Comparing Toolkit Feature Selection Methods

The feature selection algorithms included in *Chameleon* were applied to the four data sets in order to compute lists of feature importance. Classifications were then performed on the selected subsets and their performances compared.

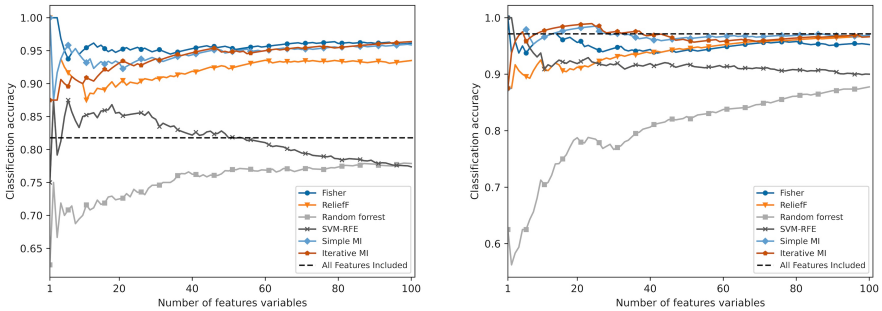
Example results from two of the toolkit classifiers are shown for each data set, as all methods of classification yielded similar results. Classification accuracy results across all classifiers using the subset of first 100 selected features are shown in Tables 2–4. For the colon cancer data set, we show results from NB and MLP classifiers (Fig. 2). For both classifiers, four of the feature selection methods

[4] `https://github.com/diviyat/chameleon`



(a) Naïve-Bayes Classifier **(b)** Multi-Layer Perceptron Classifier

Fig. 2: Colon cancer data set classification accuracy scores for six feature selection methods

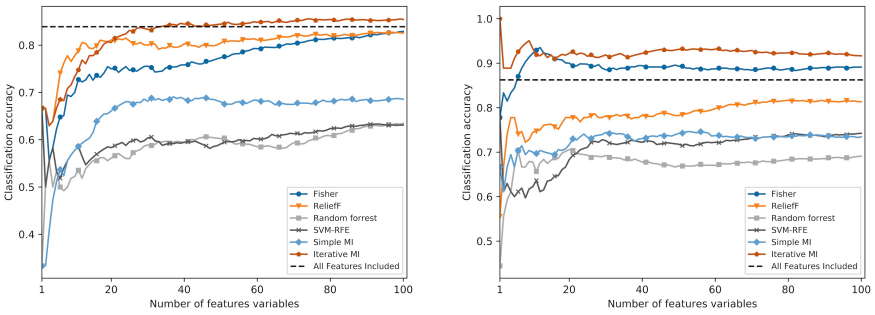


(a) KNN Classifier **(b)** Naive-Bayes Classifier

Fig. 3: Leukemia cancer data set classification accuracy scores for six feature selection methods

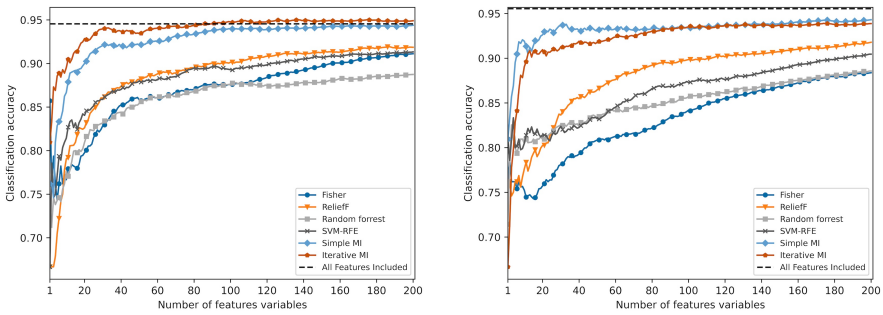
(FS, ReliefF, and both MI methods) picked subsets that performed better than the baseline accuracy of using all features. The FS algorithm provides the most relevant features and the best accuracy for the NB classifier, while the ReliefF algorithm provides the highest performing feature subsets for the MLP classifier. The same features selection methods also provided high classification accuracies from from KNN and NB for the Leukemia cancer data set (Fig. 3).

For the much higher dimensional Glioma brain tumour data set, three feature selection methods (FS and both MI methods) provided classification accuracy values that are better, or closely approach, the baseline accuracy (Fig. 4). A model fit to a SVM classifier with a subset of 40 features chosen using the iterative MI selection algorithm can be seen to provide better accuracy compared to a model built using all 22,283 features in the data set.



(a) SVM Classifier **(b)** Random Forest Classifier

Fig. 4: Glioma brain tumour data set classification accuracy scores for six feature selection methods

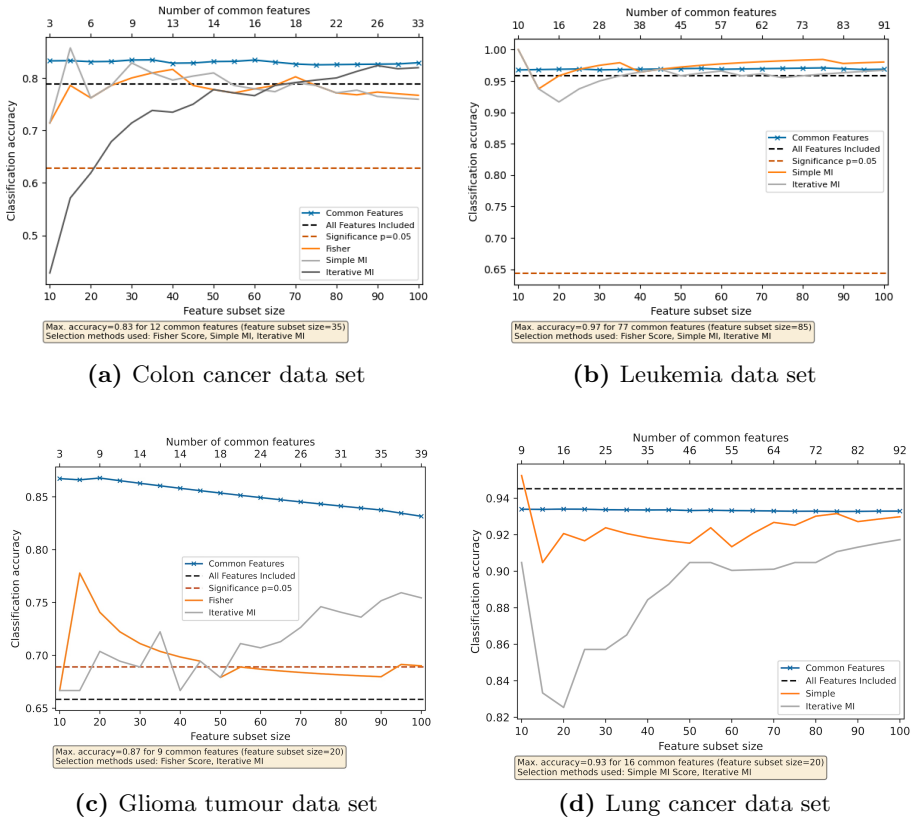


(a) Random Forest Classifier **(b)** Logistic Regression Classifier

Fig. 5: Lung cancer data set classification accuracy scores for six feature selection methods

Finally, for the lung cancer data set, both MI methods can be seen to select highly relevant smaller feature subsets which result in accuracy values which out-perform or closely approach the baseline accuracy values for random forest and logistic regression classifiers (Fig. 5).

The results confirm that different feature selection methods do not perform uniformly. Certain feature selection algorithms included in the toolkit are able to better select a smaller subset of features that gives similar or improved classification results compared to classification accuracy computed using all features in a high-dimensional data set. Feature selection by MI methods can be seen be classifier agnostic as the feature subsets they selected result in generally high classification performance across the six different classifiers. Other feature selection algorithms provide the best classification results for certain classifiers, such



(a) Colon cancer data set **(b)** Leukemia data set

(c) Glioma tumour data set **(d)** Lung cancer data set

Fig. 6: Classification accuracy for common features selection method using a neural network (multi-layer perceptron) classifier.

as the ReliefF algorithm used to select a subset of features from the colon cancer data set for the MLP classifier (Fig. 2(b)).

3.2 Evaluating Common Features Selection Method

We show here the classification results from MLP using the common features for all data sets (Fig. 6). Features from SVM-RFE, RF or ReliefF were not included, as in general these features did not give good classification performance. The results show that while the size of the common feature set is much smaller than the respective feature sets they were chosen from, yet the classification performance is almost always better. Further, the common features achieve a higher classification performance at a much smaller set size than the features selected from the individual methods.

Table 2: Classification performance in AUC using the first 100 selected features for the Colon Cancer data set. Twenty-eight (28) common features were identified from FS, ReliefF and iterative MI feature selection algorithms. For each classification method, we identify, in italic, the best performance.

Feature Selection	Naive-Bayes	KNN	Logistic Reg	Neural Net	Random Forest	SVM
Fisher Score	*0.93 ± 0.11*	*0.93 ± 0.10*	0.91 ± 0.11	0.86 ± 0.15	0.92 ± 0.10	0.92 ± 0.12
ReliefF	0.91 ± 0.11	0.92 ± 0.09	0.90 ± 0.09	0.89 ± 0.17	0.90 ± 0.13	0.95 ± 0.08
Random Forest	0.69 ± 0.32	0.74 ± 0.19	0.77 ± 0.19	0.77 ± 0.24	0.82 ± 0.18	0.81 ± 0.21
SVM-RFE	0.74 ± 0.27	0.85 ± 0.13	0.90 ± 0.12	0.89 ± 0.12	0.90 ± 0.15	0.89 ± 0.17
Iterative MI	0.82 ± 0.21	0.88 ± 0.17	0.94 ± 0.17	*0.89 ± 0.11*	0.91 ± 0.11	0.90 ± 0.15
Simple MI	0.90 ± 0.10	0.91 ± 0.15	*0.94 ± 0.08*	0.88 ± 0.12	*0.93 ± 0.07*	*0.95 ± 0.11*
Common Features	0.95 ± 0.11	0.92 ± 0.09	0.89 ± 0.16	0.86 ± 0.15	0.94 ± 0.08	0.92 ± 0.13

Table 3: Classification performance in AUC using the first 100 selected features for the Glioma Brain Tumour data set. Thirty-nine (39) common features were identified from FS, and iterative MI feature selection algorithms. For each classification method, we identify, in italic, the best performance.

Feature Selection	Naive-Bayes	KNN	Logistic Reg	Neural Net	Random Forest	SVM
Fisher Score	*0.97 ± 0.04*	0.93 ± 0.10	*0.95 ± 0.08*	0.85 ± 0.15	*0.96 ± 0.06*	*0.97 ± 0.04*
ReliefF	0.89 ± 0.11	0.92 ± 0.10	0.91 ± 0.11	0.80 ± 0.19	0.96 ± 0.07	0.94 ± 0.05
Random Forest	0.77 ± 0.15	0.79 ± 0.13	0.84 ± 0.13	0.81 ± 0.15	0.95 ± 0.08	0.77 ± 0.21
SVM-RFE	0.65 ± 0.22	0.76 ± 0.12	0.87 ± 0.10	0.67 ± 0.17	0.94 ± 0.08	0.89 ± 0.12
Iterative MI	0.94 ± 0.08	*0.95 ± 0.07*	0.94 ± 0.12	*0.85 ± 0.10*	0.96 ± 0.07	*0.97 ± 0.04*
Simple MI	0.81 ± 0.19	0.85 ± 0.17	0.80 ± 0.17	0.78 ± 0.18	0.86 ± 0.13	0.86 ± 0.07
Common Features	0.96 ± 0.04	0.90 ± 0.16	0.86 ± 0.11	0.81 ± 0.13	0.96 ± 0.06	0.99 ± 0.02

We compared the classification performance of all classifiers using the first 100 selected features from all feature selection methods and their common features subsets for colon cancer (Table 2), Glioma brain tumour (Table 3) and lung cancer (Table 4) data sets. 10-fold cross-validation evaluation was performed on the four high-dimensional data sets to gather the AUC values and the variance in the performance.

For the colon cancer data set (Table 2), the feature selection algorithms determined to provide the best performing were FS, ReliefF and iterative MI algorithms. Using these methods to generate subsets including 100 of the most important features, 28 features were identified as common between them. It can be seen that the common features provide similar (within the margin of error) classification performance for all classifiers.

For the Glioma brain tumour data set (Table 3), the best performing feature selection algorithms were FS and iterative MI algorithms, which generated a subset with 39 features using the common features method. Classification results show the reduced sized common features subset has similar performance against for all classifiers except Logistic Regression.

Using both MI features selections, 88 common features were selected from 100 feature subsets for the lung cancer data set (Table 4). The classification performance using the common features were better than using the subsets from individual feature selection methods for all classifiers.

Table 4: Classification performance in AUC using the first 100 selected features for the Lung Cancer data set. Eighty-eight (88) common features were identified from iterative and simple MI feature selection algorithms. For each classification method, we identify, in italic, the best performance.

Feature Selection	Naive-Bayes	KNN	Logistic Reg	Neural Net	Random Forest	SVM
Fisher Score	0.83 ± 0.11	0.94 ± 0.06	0.97 ± 0.02	0.95 ± 0.03	0.97 ± 0.03	0.96 ± 0.03
ReliefF	0.94 ± 0.04	0.97 ± 0.03	0.97 ± 0.02	0.95 ± 0.05	0.98 ± 0.02	*0.99 ± 0.01*
Random Forest	0.95 ± 0.02	0.95 ± 0.04	0.96 ± 0.02	0.93 ± 0.04	0.97 ± 0.04	0.97 ± 0.02
SVM-RFE	0.90 ± 0.04	0.96 ± 0.02	0.96 ± 0.02	0.96 ± 0.04	0.97 ± 0.03	0.97 ± 0.02
Iterative MI	0.98 ± 0.02	*0.98 ± 0.02*	0.98 ± 0.02	*0.97 ± 0.02*	*0.99 ± 0.01*	*0.99 ± 0.01*
Simple MI	*0.99 ± 0.01*	0.97 ± 0.04	*0.98 ± 0.01*	0.96 ± 0.02	*0.99 ± 0.01*	*0.99 ± 0.01*
Common Features	0.99 ± 0.01	0.98 ± 0.03	0.98 ± 0.02	0.97 ± 0.03	0.99 ± 0.01	0.99 ± 0.02

It can be seen that by utilising the best performing feature selection algorithms to select the features common between their determined feature subsets, improved or similar classification results are able to be achieved with smaller, and thus more computationally efficient, subsets of relevant and non-redundant features.

4 Conclusion

We presented *Chameleon*, a Python based toolkit for feature selection based on categorical variables and subsequent classification performance evaluation. *Chameleon* integrates various necessary steps including data pre-processing, k-fold cross validation, six feature selection algorithms, six different well-established classification methods, two classification performance metrics, as well as easy to understand visualisation of the results.

A rigorous analysis was conducted by means of combining the six classifiers with the six toolkit feature selection algorithms, as well as the introduced common features method, for binary and multi-class high-dimensional data sets. It was found that certain feature selection algorithms were able to select smaller subsets of features which yielded better classification performance compared to baseline accuracy computed with all features in the data sets. Using such high performing algorithms to select the common features between them resulted in improved or similar classification performance, compared to the individual methods of feature selection, using significantly smaller subsets of features.

It can be seen that the high-dimensional data sets tested result in generally high classification accuracy results. In future work, the toolkit's feature selection algorithms, as well as the proposed common features method, may be tested using more challenging data sets that are not so easily classified. Future development of *Chameleon* will include extending to more feature selection algorithms, adding other performance metrics, and extending the data formats permitted by the package beyond `.mat` and `.arff` formats.

Acknowledgement

Simon McManis was supported by 2019/2020 CSIRO Data61 Summer Scholarship for this project.

References

1. Alon, U., Barkai, N., Notterman, D.A., Gish, K., Ybarra, S., Mack, D., Levine, A.J.: Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. Proceedings of the National Academy of Sciences **96**(12), 6745–6750 (1999)
2. Altman, N.S.: An introduction to kernel and nearest-neighbor nonparametric regression. The American Statistician **46**(3), 175–185 (1992)
3. Battiti, R.: Using mutual information for selecting features in supervised neural net learning. IEEE Trans. Neural Networks **5**(4), 537–550 (1994)
4. Bolón-Canedo, V., Alonso-Betanzos, A.: Ensembles for feature selection: A review and future trends. Information Fusion **52**, 1–12 (2019). https://doi.org/10.1016/j.inffus.2018.11.008
5. Breiman, L.: Random forests. Machine learning **45**(1), 5–32 (2001)
6. Caruana, R., Freitag, D.: Greedy attribute selection. In: Machine Learning Proceedings 1994, pp. 28–36. Elsevier (1994)
7. Dash, M., Choi, K., Scheuermann, P., Liu, H.: Feature selection for clustering-a filter solution. In: 2002 IEEE International Conference on Data Mining, 2002. Proceedings. pp. 115–122. IEEE (2002)
8. Dash, M., Liu, H.: Feature selection for classification. Intelligent data analysis **1**(3), 131–156 (1997)
9. Díaz-Uriarte, R., De Andres, S.A.: Gene selection and classification of microarray data using random forest. BMC bioinformatics **7**(1), 3 (2006)
10. Dreiseitl, S., Ohno-Machado, L.: Logistic regression and artificial neural network classification models: a methodology review. Journal of biomedical informatics **35**(5-6), 352–359 (2002)
11. Dy, J.G., Brodley, C.E.: Feature selection for unsupervised learning. Journal of machine learning research **5**(Aug), 845–889 (2004)
12. Freije, W.A., Castro-Vargas, F.E., Fang, Z., Horvath, S., Cloughesy, T., Liau, L.M., Mischel, P.S., Nelson, S.F.: Gene expression profiling of gliomas strongly predicts survival. Cancer research **64**(18), 6503–6510 (2004)
13. Friedman, J., Hastie, T., Tibshirani, R.: The elements of statistical learning, vol. 1. Springer series in statistics New York (2001)
14. Golub, T.R., Slonim, D.K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J.P., Coller, H., Loh, M.L., Downing, J.R., Caligiuri, M.A., et al.: Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. science **286**(5439), 531–537 (1999)
15. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. Journal of machine learning research **3**(Mar), 1157–1182 (2003)
16. Guyon, I., Weston, J., Barnhill, S., Vapnik, V.: Gene selection for cancer classification using support vector machines. Machine learning **46**(1-3), 389–422 (2002)
17. Hong, Z.Q., Yang, J.Y.: Optimal discriminant plane for a small number of samples and design method of classifier on the plane. pattern recognition **24**(4), 317–324 (1991)

18. Kira, K., Rendell, L.A., et al.: The feature selection problem: Traditional methods and a new algorithm. In: Aaai. vol. 2, pp. 129–134 (1992)
19. Kononenko, I., Šimec, E., Robnik-Šikonja, M.: Overcoming the myopia of inductive learning algorithms with relieff. Applied Intelligence **7**(1), 39–55 (1997)
20. Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R.P., Tang, J., Liu, H.: Feature selection: A data perspective. ACM Computing Surveys (CSUR) **50**(6), 94 (2018)
21. Lizier, J.T.: Jidt: An information-theoretic toolkit for studying the dynamics of complex systems. Frontiers in Robotics and AI **1**, 11 (2014). https://doi.org/10.3389/frobt.2014.00011
22. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research **12**, 2825–2830 (2011)
23. Ruck, D.W., Rogers, S.K., Kabrisky, M.: Feature selection using a multilayer perceptron. Journal of Neural Network Computing **2**(2), 40–48 (1990)
24. Sun, L., Zhang, X.Y., Qian, Y.H., Xu, J.C., Zhang, S.G., Tian, Y.: Joint neighborhood entropy-based gene selection method with fisher score for tumor classification. Applied Intelligence **49**(4), 1245–1259 (2019)
25. Wang, X.R., Lizier, J.T., Nowotny, T., Berna, A.Z., Prokopenko, M., Trowell, S.C.: Feature selection for chemical sensor arrays using mutual information. PLoS ONE **9**(3), e89840 (03 2014). https://doi.org/10.1371/journal.pone.0089840
26. Yu, L., Liu, H.: Efficient feature selection via analysis of relevance and redundancy. Journal of machine learning research **5**(Oct), 1205–1224 (2004)
27. Zhang, H.: The optimality of naive bayes, flairs conference (2004)