Abstract

Feature selection is a process that aims to reduce the number of variables when building a prediction model or performing a machine learning procedure. In this paper, we propose an automated machine learning mechanism for the task of feature selection and that relies on the comparison between two methods: Random Forest and XGBoost classifier. We present both backward and forward approaches for the feature selection process and test the proposed algorithm on four different datasets. In all cases, the results show that the number of features for building the model can be significantly reduced while retaining high model accuracy. The proposed automated feature selection method presents an effective and efficient strategy for users to adopt in order to choose accurate algorithms and features that significantly influence the predicted variable.

Feature selection, AutoML, Random forest, XGBoost

1 Introduction

Feature selection is one of the most important tasks and a core concept in machine learning, especially in predictive models. Using irrelevant features when training a model may affect the performance of the model, reduce accuracy, and cause overfitting. By choosing wisely the best and most significant features from the data when building the model, one avoids overfitting, improves prediction accuracy, and reduces the training time. Feature selection has been widely studied in the literature (see, e.g. [**?**], [**?**], [**?**], [**?**], [**?**], [**?**] and many references therein) and is used in many fields, such as statistical pattern recognition [**?**], [**?**], [**?**], face recognition [**?**], data mining and machine learning [**?**], [**?**], [**?**], [**?**], text categorization [**?**], customer relationship management [**?**], bioinformatics [**?**], genomics [**?**], and cross-project defect prediction [**?**]. Furthermore, Ref. [**?**] provides a comprehensive survey of online feature selection with streaming features (i.e., when features are generated dynamically).

Feature selection methods are mainly divided into filter methods, wrapper methods, and embedded methods. Filter methods use variable ranking techniques and ranking criteria to decide whether a variable should be removed from the model. In wrapper methods, a subset of features is evaluated by using a machine learning algorithm that employs a search strategy to look through the space of possible feature subsets. Each subset is evaluated based on the quality of the performance of a given algorithm. Embedded methods

perform feature selection during the execution of the modeling algorithm. For a review of these methods, see Ref. [**?**].

This paper presents an automated feature selection mechanism. After receiving the data, the mechanism first executes two feature selection methods: Random Forest [**?**] and XGBoost [**?**]. Next, according to each method, it determines the importance of each feature and, as a result, which features should be used in the model.

Automated machine learning (AutoML) is an artificial-intelligence-based method that automates the process of machine learning by building efficient and high-quality machine learning algorithms. A recent comprehensive survey of AutoML can be found in Ref. [**?**] and references therein.

We focus herein on the Random Forest classifier and the XGBoost algorithm. Reference [**?**] reports that a feature selection based on the Random Forest classifier provides multivariate feature importance scores, which are relatively cheap to obtain and which have been successfully applied to high-dimensional data. Random Forest performs an implicit feature selection by using a small subset of "useful variables" for the classification only. This provides, eventually, an indicator of feature relevance. XGBoost is a scalable machine learning system that is commonly applied in tree boosting [**?**]. Reference [**?**] states that the XGBoost algorithm provides a trained predictive model that automatically estimates the trained feature importance. The XGBoost algorithm improves the performance of the model by alleviating the effects of redundant features and noise. Moreover, the algorithm prevents overfitting through feature subsampling or column subsampling.

Naturally, one of the most interesting issues when performing variable selection is accuracy, see Ref. [**?**]. In other words, we are interested in whether the accuracy achieved by using all features in the machine learning model significantly exceeds the accuracy of the model when using only the selected (most important) variables. Put another way, we are interested in whether it suffices to use a small (but how small) number of features without reducing the accuracy.

The proposed automated mechanism iteratively performs the Random Forest and XGBoost algorithms. In each iteration, we keep the most important features according to their rank in the Random Forest and XGBoost classifier and only use them when solving some given classification problem. We then calculate the accuracy of this model and compare it with the accuracy of the full model (i.e., a Random Forest or an XGBoost classifier with all features). In the following iteration, we add another feature to the model (according to the ranks of the features) and calculate its accuracy. This procedure stops when the accuracy of the full model (with all features)

differs only negligibly from that of the partial model (with only the selected features).

The rest of the paper is organized as follows: Section 2 describes the proposed algorithm, and Section 3 presents the implementation steps. The results and comparisons between the Random Forest classifier and the XGBoost algorithm are given in Section 4. Section 5 concludes the paper.

2 The Method

In this paper, we define the AutoML method that performs the automated feature selection and reduction. The underlying process is as follows:

1. Run a selected algorithm on a full dataset D, i.e., with all features (in this paper, we apply both the Random Forest classifier and the XGBoost algorithm).

2. Let $AC(D)$ be the accuracy of step 1.

3. Use a well-defined feature-importance method $f(D)$ (in this work we use the Random Forest classifier and the XGBoost algorithm).

4. Sort the $f(D)$ features list by importance. Let $X_1(D)$ denote the first feature in the list of ordered features (i.e., the most "important" feature), and let $X_n(D)$ denote the last feature in the list of ordered features (i.e., the most "unimportant" feature).

5. **Option A:** Use the backward approach; that is, remove variables until the accuracy between a full model and a partial model exceeds some pre-determined error E. This approach uses the following main steps:

   (a) Let n be the number of features in the dataset D.

   (b) Omit $X_n(D)$ from dataset D and create $D_{new} = D[-X_n(D)]$.

   (c) Run the selected algorithm from step 1 on D_{new}.

   (d) Let $AC(D_{ne}w)$ be the accuracy of step 5.A.c.

   (e) While $[AC(D) - AC(D_{new}) \leq E$ and $n > 0]$ do

      i. $n = n - 1$

      ii. $D_{new} = D_{new}[-X_n(D)]$

      iii. Run the selected algorithm on D_{new}

      iv. $AC(D_{new})$ = the accuracy of step (e)iii.

Option B: Use the forward approach; that is, start with a model consisting of only the predicted (dependent) variable and add (independent) features to the model, as long as the difference between the accuracy of the full model and the partial model is greater than some error E. Once the difference is less than E, we stop and use the model with only the selected features. This approach has the following main steps:

(a) Let n be the number of features in the dataset D and let $b = 1$.
(b) Create a new empty dataset D_{new} that contains only the (single) dependent variable.
(c) Add $X_1(D)$ to D_{new}.
(d) Run the selected algorithm from step 1 on D_{new}.
(e) Let $AC(D_{new})$ be the accuracy of step 5.B.d.
(f) While $[AC(D) - AC(D_new) > E$ and $b < n]$ do
    i. $b = b + 1$
    ii. $D_{new} = D_{new}[+X_b(D)]$
    iii. Run the selected algorithm from step 1 on D_{new}
    iv. $AC(D_{new})$ = the accuracy of step (f)iii.

Note that the parameter E determines a threshold level for error accuracy, which should be modified according to various factors and considerations, such as

- the research domain (for example, in the health care domain, the prediction must be very high);
- quality of the data (sample size, missing values, outliers, etc.);
- use-case analysis;
- other statistical measures and factors (dependencies, multi-collinearity, bias, etc.);
- model flexibility.

We present both the backward and forward approaches because, depending on the research domain, one approach might be more suitable than the other. For example, if we assume that an accuracy of 80% suffices, we apply the forward approach (i.e., add features gradually to the model until this level of accuracy is achieved). Conversely, if we want to reduce the number of features but maintain some minimum deviation from the accuracy of the full model, we will prefer the backward approach.

3 Implementation

To illustrate the proposed mechanism, we perform the following implementation procedures:

1. We test the proposed mechanism on four different datasets, which are presented and detailed in the sequel.**[AU: Do you mean "which will be detailed in a forthcoming presentation"?]**In each dataset, we solve some classification problem.

2. We use the Random Forest and XGBoost algorithms (i) for feature selection and (ii) as the prediction model for the classification problem and calculate the accuracy. To this end, we use the libraries sklearn.ensemble.RandomForestClassifier (see Ref. [**?**] and the xgboost import XGBClassifier).

3. We use pandas [**?**] for handling with our datasets and derive the statistical results and measures.

4. We test the proposed procedure on the following datasets:

   (a) Dataset 1 is Wine Quality [**?**]. This dataset consists of 4898 records, 11 features, and a categorical target variable with 11 different classes.

   (b) Dataset 2 is the Cleveland Heart Disease Dataset [**?**]. We use the processed.cleveland.data dataset, which contains 303 records with a total of 14 features, including the classification target (with 5 classes).

   (c) Dataset 3 is breast-cancer-Wisconsin [**?**]. This dataset consists of 699 records, 10 features, and a categorical target variable (with 2 classes).

   (d) Dataset 4 is the Internet Firewall (see, e.g., Ref. [**?**]). This dataset consists of 65 532 records and twelve features including the classification categorical target variable (with four different classes).

5. We implement both backward and forward approaches (as described in Section 2) on each of the selected datasets detailed above. For each dataset, we provide the following results:

   (a) feature importance sorted list derived from the Random Forest classifier and from the XGBoost algorithm;

(b) one comparative accuracy graph per model of the backward approach;

(c) one comparative accuracy graph per model of the forward approach.

At the end of the process, this procedure returns the best model with the optimal number of features selected for each dataset. Figure 1 describes the flow of the AutoML implementation steps. We start our implementation by splitting the data into a training set and a test set. Next, we run the Random Forest algorithm and generate the feature-importance list. If the list is not empty, we drop one feature and rerun the algorithm to generate a new list. We then calculate the accuracy and save it in the algorithm's feature-accuracy list. We compile the Random Forest feature accuracy list if the importance is not greater than zero. Furthermore, we successively iterate the procedure by using the XGBoost algorithm and compare the accuracy obtained by using the features from the two lists. The final output is the accuracy needed and the optimal number of features.

Figure 1: Flow chart of implementation steps.

4 Results

This section presents the results of the proposed mechanism for each of the four datasets described in Section 2.

4.1 Dataset 1: Wine-quality dataset

Table 1 presents the sorted feature importance list based on the outcomes of both the Random Forest algorithm and the XGBoost algorithm. The results for dataset 1 show that the accuracy of a full Random Forest model (consisting of all features) is 0.6020, while that of the full XGBoost model is 0.6562. Figure 2 presents the accuracy of the fitted Random Forest and XGBoost models under the backward approach. In other words, we start with a full model with all eleven features and then remove features according to their importance given in Table 1. In this case, reducing the number of features to only five (out of eleven) does not dramatically change the accuracy of the model. However, Figure 2 shows that the XGBoost method is more accurate than the Random Forest method.

Figure 3 depicts the accuracy for the forward approach. We start with a model consisting only of the most important feature, which, for both Random Forest and XGBoost, results in a low accuracy of about 0.51. We then add features according to their importance until reaching the desired accuracy. Again, good accuracy is obtained with only five features. Both Figures 2 and 3 show that, for dataset 1, the XGBoost model is more accurate than the Random Forest classifier.

Table 1: Feature importance for dataset 1 according to Random Forest and XGBoost.

Feature name	**Importance Random Forest**	**Feature name**	**Importance XGBoost**
Alcohol	0.242851	Alcohol	0.201177
Sulphates	0.140236	Total sulfur dioxide	0.105005
Total sulfur dioxide	0.115642	sulphates	0.101907
Volatile acidity	0.111605	Volatile acidity	0.09821
Density	0.092982	Free sulfur dioxide	0.07577
Chlorides	0.057417	Fixed acidity	0.075138
Citric acid	0.053522	pH	0.074227
Fixed acidity	0.052005	Residual sugar	0.072228
pH	0.045732	Citric acid	0.065855
Residual sugar	0.044457	Density	0.065293
Free sulfur dioxide	0.043558	Chlorides	0.06519

Figure 2: Model accuracy for the backward approach applied to dataset 1.

Figure 3: Model accuracy for the forward approach applied to dataset 1.

4.2 Dataset 2: Cleveland heart disease dataset

Table 2 presents the results of the feature importance process executed on dataset 2, as obtained by Random Forest and XGBoost. Note that the accuracy of the full model according to Random Forest (XGBoost) is 0.5604 (0.4945). Figure 4 shows that, according to the Random Forest classifier, eliminating variables from the model increases the accuracy. This often occurs since having many variables in the model may cause overfitting and increase the variance. A model with two features attains the best accuracy when using Random Forest, whereas four features are required when using XGBoost. This is also shown in Figure 5, where the accuracy is given for the forward approach (i.e., when adding features). A model with a single independent feature gives poor accuracy with Random Forest, but, surprisingly, when using a single feature, it does not give worse accuracy than when using XGBoost. Adding only a single extra feature to the model with Random Forest significantly improves the accuracy, while the accuracy for the XGBoost model rises in a more moderate manner. Overall, it is evident from Figures 4 and 5 that Random Forest results in better accuracy for dataset 2.

4.3 Dataset 3: Breast-cancer-Wisconsin dataset

Table 3 presents the order of feature importance for the breast-cancer dataset. The accuracies of the full Random Forest model and the full XGBoost model

Table 2: Feature importance for dataset 2 when using Random Forest and XGBoost.

Feature name	**Importance Random Forest**	**Feature name**	**Importance XGBoost**
feature 2	0.177796	feature 11	0.153479
feature 11	0.146085	feature 2	0.143723
feature 1	0.139925	feature 1	0.12426
feature 6	0.101466	feature 5	0.083033
feature 4	0.098772	feature 3	0.076648
feature 5	0.082157	feature 4	0.064499
feature 13	0.079420	feature 12	0.063392
feature 3	0.045154	feature 16	0.059724
feature 9	0.042083	feature 8	0.058084
feature 12	0.037607	feature 13	0.052149
feature 10	0.035161	feature 9	0.048032
feature 7	0.013243	feature 10	0.044038
feature 8	0.001131	feature 7	0.028939

Figure 4: Model accuracy when applying the backward approach to dataset 2.

are both about 0.9714. For the backward approach, Figure 6 shows that a model with three features (out of ten), reaches a very good accuracy for the Random Forest classifier (almost as good as for the full model), whereas six features provide good accuracy for the XGBoost model. This phenomenon also appears in the lower part of Figure 7, which shows the accuracy for the forward approach.

Figure 5: Model accuracy when applying the forward approach to dataset 2.

Figure 6: Model accuracy when applying the backward approach to dataset 3.

4.4 Dataset 4: Internet firewall dataset

In this final example, we consider the firewall data set. Feature importance is given in Table 4. The accuracy of a full Random Forest (XGBoost) model with all eleven features is 0.9984 (0.9986). However, the results in Figures 8 and 9 show that even a model with only two features reaches almost the same accuracy using either Random Forest or XGBoost.

Table 3: Feature importance for dataset 3 according to Random Forest and XGBoost.

Feature name	**Importance Random Forest**	**Feature name**	**Importance XGBoost**
feature 7	0.256161	feature 7	0.565556
feature 8	0.233745	feature 8	0.231707
feature 4	0.155182	feature 3	0.056456
feature 3	0.128431	feature 4	0.050602
feature 5	0.092941	feature 2	0.044598
feature 2	0.080215	feature 9	0.024274
feature 9	0.034542	feature 6	0.010903
feature 6	0.015189	feature 5	0.010789
feature 10	0.002378	feature 10	0.005116
feature 1	0.001223	feature 1	0

Figure 7: Model accuracy when applying the forward approach to dataset 3.

5 Concluding remarks

This paper presents an automated feature importance method based on the Random Forest and XGBoost algorithms. For a given dataset, the proposed mechanism suggests which features should be used in the model and which should be omitted while maintaining high accuracy. Reducing the number of features may reduce the complexity of the model and, as shown in our examples, does not drastically affect performance (i.e., model accuracy). Specifically, we test the proposed method on four different datasets by solving a classification problem. For each dataset, we first apply the Random Forest and the XGBoost algorithms to derive the feature importance. Next,

Table 4: Feature importance for dataset 4 according to Random Forest and XGBoost.

Feature name	**Importance Random Forest**	**Feature name**	**Importance XGBoost**
Destination Port	0.225071	Elapsed Time	0.793872
Elapsed Time	0.192756	Destination Port	0.083326
NAT Source Port	0.144005	Bytes	0.077337
NAT Destination Port	0.120887	Packets	0.03966
Packets	0.074335	NAT Source Port	0.00164
Bytes	0.065065	Bytes Received	0.001225
pkts received	0.050558	Bytes Sent	0.001096
Bytes Sent	0.046179	NAT Destination Port	0.001009
Source Port	0.040731	Source Port	0.000374
Bytes Received	0.038632	pkts received	0.000265
pkts sent	0.001781	pkts sent	0.000197

Figure 8: Model accuracy when applying the backward approach to dataset 4.

according to the importance of features, we use the backward approach (i.e., starting with a full model and removing variables according to accuracy criteria) and the forward approach (i.e., starting with an empty model and adding variables according to pre-determined criteria). The measured accuracy is referred to as a classification model, which we implement using Random Forest. For all datasets, we conclude that the number of features used for building the model may be reduced by half (and even by more than that), while keeping the model accuracy very close to the accuracy of a full model (with all features). The results also show that, for some datasets, the Random Forest classifier outperforms XGBoost (datasets 2 and 3), whereas,

Figure 9: Model accuracy when applying the forward approach to dataset 4.

for dataset 1, XGBoost gives better accuracy. For dataset 4, both methods yield quite the same accuracy, except for the case when only a single feature is used, in which case Random Forest is better. This automated feature selection method is an effective process for selecting the optimal number of features for predictive machine learning models, thus enhancing the accuracy of fit. Implementing a machine learning model with the appropriate features increases the model's performance and reduces computational costs. Overall, the method is efficient and states which features strongly influence the response variable.