Practical Application 1

Machine Learning

Cerezo Pomykol, Jan
j.cerezo@alumnos.upm.es

October 18, 2022

Contents

1 Introduction

The goal of this practical application is to study how feature subset selection (FSS) affects different machine learning models. Specifically how they perform with a dataset with all variables and datasets obtained from a univariate filter, a multivariate filter, and a multivariate wrapper. The selected classification models are the following:

- k-Nearest Neighbours.
- Rule Induction - RIPPER algorithm.
- Support Vector Machines.
- Neural Network - Multilayer Perceptron.
- Classification Trees - C4.5 algorithm.

For this assignment, the dataset chosen [?][?] contains data about instances of dry beans, which was created from images of 13,611 grains of 7 different varieties.

The rest of the document has the following structure. Firstly, de dataset is analysed in the Problem Description, later the methodology is specified in the Methodology section. Subsequently the results are shown in the Results section, and finally they are analysed in the Discussion and Conclusion sections.

2 Problem Description

The dataset contains 13,611 instances created from images of dry beans of 7 different varieties. There are a total of 17 attributes (16 plus the class column) and their significance is the following:

1 **Area** (A): The area of a bean zone and the number of pixels within its boundaries.

2 **Perimeter** (P): Length of its border.

3 **Major axis length** (L): Length of the longest line that can be drawn from a bean.

4 **Minor axis length** (l): Length of the longest line that can be drawn from the bean while standing perpendicular to the main axis.

5 **Aspect ratio** (K): Relationship between L and l.

6 **Eccentricity** (Ec): Eccentricity of the ellipse having the same moments as the region.

7 **Convex area** (C): Number of pixels in the smallest convex polygon that can contain the area of a bean seed.

8 **Equivalent diameter** (Ed): The diameter of a circle having the same area as a bean seed area.

9 **Extent** (Ex): The ratio of the pixels in the bounding box to the bean area.

10 **Solidity** (S): Also known as convexity. The ratio of the pixels in the convex shell to those found in beans.

11 **Roundness** (R): Calculated with the following formula: $\frac{4\pi A}{P^2}$

12 **Compactness** (CO): Measures the roundness of an object: $\frac{Ed}{L}$

13 **ShapeFactor1** (SF1)

14 **ShapeFactor2** (SF2)

15 **ShapeFactor3** (SF3)

16 **ShapeFactor4** (SF4)

17 **Class**: Seker, Barbunya, Bombay, Cali, Dermosan, Horoz and Sira.

All attributes are numeric, except the class which is nominal. Additionally, there are no *meta-attributes* such as instance or object identifiers, so there is no need to remove any column. Table 1 shows more information about the ranges of values and Table 2 shows the number of instances of each class.

No.	Name	Min.	Max.
1	Area	20420	254616
2	Perimeter	524.736	1985.37
3	Major axis length	183.601	738.86
4	Minor axis length	122.513	460.198
5	Aspect ratio	1.025	2.43
6	Eccentricity	0.219	0.911
7	Convex area	20684	263261
8	Equivalent diameter	161.244	569.374
9	Extent	0.555	0.866
10	Solidity	0.919	0.995
11	Roundness	0.49	0.991
12	Compactness	0.641	0.987
13	ShapeFactor1	0.003	0.01
14	ShapeFactor2	0.001	0.004
15	ShapeFactor3	0.41	0.975
16	ShapeFactor4	0.948	1
17	Class	-	-

Table 1: Value ranges of each variable.

Class	n
SEKER	2027
BARBUNYA	1322
BOMBAY	522
CALI	1630
HOROZ	1928
SIRA	2636
DERMASON	3546

Table 2: Number of instances per class.

Since there are some variables with high values compared to others (for example Area compared to Extent); normalization is necessary.

3 Methodology

3.1 Software

For this practical application, Weka [**?**] has been used to perform the training and evaluation of all models, as well as the normalization of the dataset. Besides from normalization of the dataset, there is no need to perform more pre-processing. There are no missing values and all variables are numerical (except the class).

3.2 Evaluation

The evaluation of each model will be conducted with cross validation with 10 folds. Because each class has different weight (different number of instances) the confusion matrix will be discussed as well. Besides the accuracy, training time will also be considered.

3.3 Classification Algorithms

The following machine learning algorithms have been evaluated. Note that some of them have the option to normalize de input data, which was set to *false* because the data was already normalized in the pre-processing stage.

- **Logistic Regression**. Default parameters.
- **Naive Bayes**. Default parameters.
- **Tree Augmented Naive Bayes**. The search algorithm is set to *TAN*.
- **Linear Discriminant Analysis**. .
- **Fusion**. With Naive Bayes, TAN and LDA.
- **Stacking**. With (TAN and LDA) and Naive Bayes.
- **Bagging**. With Naive Bayes.
- **Random Forest**. .
- **Boosting**. With Naive Bayes.
- **Naive Bayes Tree**. .
- **Logistic Model Trees**. .

Table 3 shows the Weka function associated to each algorithm.

Algorithm	Weka Function
Logistic Regression	functions.Logistic
Naive Bayes	bayes.NaiveBayes
Tree Augmented Naive Bayes	bayes.BayesNet
Linear Discriminant Analysis	functions.LDA
Fusion	meta.Vote
Stacking	meta.Stacking
Bagging	meta.Bagging
Random Forest	trees.RandomForest
Boosting	meta.AdaBoostM1
Lazy Bayesian Rules	lazy.LBR
Naive Bayes Tree	trees.NBTree
Logistic Model Trees	trees.LMT

Table 3: Weka implementation of each algorithm.

3.4 Feature Subset Selection

The selection of features has been performed with the following algorithms. Table 4 shows the Weka function associated to each method.

- **No FSS**. The original dataset was used.
- **Univariant Filter**. Evaluates the worth of each attribute by measuring the information gain with respect to the class. The threshold used for this dataset is 1.2.
- **Multivariant Filter**. Evaluates the worth of a subset of attributes by considering the individual predictive ability of each feature along with the degree of redundancy between them.
- **Wrapper Approach**. Evaluates each subset of attributes with the estimated performance of a classifier built with this subset of attributes.

FSS algorithm	Weka Function
No FSS	-
Univariant Filter	attributeSelection.InfoGainAttributeEval
Multivariant Filter	attributeSelection.CfsSubsetEval
Wrapper Approach	attributeSelection.WrapperSubsetEval

Table 4: Weka implementation of each FSS algorithm.

Note that the optimization of each classification model and feature selection method is not part of this practical application.

4 Results

This section includes the scores obtained with each classifier (section 3.3) and each Feature Subset Selection approach (section 3.4). Table 5 shows the attributes selected by each method.

Table 6 shows the score of each classification algorithm with all datasets obtained from each Feature Subset Selection method described, and Table 7 shows the training time of each model. The full version of each result report including the confusion matrices can be found in [**?**].

Note that images of the trees obtained by the C4.5 algorithm, or the neural networks generated are not shown because there are too many nodes to show and the representation is not understandable. All the rules generated from the RIPPER algorithm and the support vectors from the SVMs can be found in the complete result reports. In the same way, a representation of the outputs of the kNN algorithm is not included because it is not possible to visualize a 16-dimensional space.

5 Discussion

Generally, the models trained obtained a score of at least 90% in most cases. However, the results do not differ greatly from each other. The greatest precision obtained is 92.7852%, which was achieved with the original dataset and a Multilayer Perceptron classification algorithm.

Overall, the best performing models are the Multilayer Perceptron, the Support Vector Machine, and the k-Nearest Neighbour. Note that the latter has a training time hundreds of times faster compared to the first two.

Attribute	No FSS	Univariant	Multivariant	Wrapper (Logistic)	Wrapper (Naive Bayes)	Wrapper (TAN)	Wrapper (LDA)	Wrapper (Fusion)	Wrapper (Stacking)	Wrapper (Bagging)	Wrapper (Random Forest)	Wrapper (Boosting)	Wrapper (NBTree)	Wrapper (LMT)
Area		●					●			●	●	●		●
Perimeter	●	●	●	●	●	●	●	●	●	●		●	●	●
MajorAxisLength	●	●	●	●					●		●			●
MinorAxisLength	●	●	●	●		●		●					●	
AspectRatio	●		●			●			●					
Eccentricity	●								●					
ConvexArea	●	●	●	●			●		●					●
EquivDiameter	●	●		●					●					●
Extent	●		●	●		●	●	●	●	●	●	●		
Solidity	●			●				●		●	●		●	●
Roundness	●		●	●	●	●		●	●	●	●		●	●
Compactness	●		●		●	●	●	●		●	●		●	
ShapeFactor1	●	●	●	●	●	●				●	●			
ShapeFactor2	●	●	●	●					●	●				●
ShapeFactor3	●										●	●		
ShapeFactor4	●		●	●	●	●	●	●	●	●	●		●	●
N attributes	16	8	11	11	5	8	6	7	10	9	9	4	6	9

Table 5: Attributes selected with each FSS algorithm.

Regarding the selection of features of this dataset, it is clear that not all variables are needed to obtain models that perform as well as the model generated from all attributes.

In the case of the k-Nearest Neighbours, the reduced datasets obtained slightly better scores than the original dataset. This is probably due to some moderate overfitting. With the k parameter set to 10 the model could not fit all de dataset information, because with 10 clusters it is not possible to separate all the classes. This model obtained the best results when the training time is considered.

The Rule Induction classifier did not perform as well as the k-Nearest Neighbours. It obtained slightly poorer results than the kNN implementation, while spending more computational resources.

Both the Support Vector Machine and the Multilayer Perceptron obtained the best results. This is because both models can fit more complex data distributions than the other methods. The SVM calculated more than 1,200 support vectors for each dataset, while the MLP used between 20 and 29 neurons (**??**). The complexity of this models is reflected in the training times, which are significantly higher than the other algorithms.

Lastly the tree algorithm performed the same as the RIPPER algorithm, while spending approximately 10 times less computational time to train the models. A representation of the tree is shown in **??**. However, it is hardly understandable since Weka does

Dataset	Logistic	Naive Bayes	TAN	LDA	Fusion	Stacking	Bagging	Random Forest	Boosting	NBTree	LMT
Original	92.60	89.71	91.47	90.18	91.26	91.28	89.72	92.52	89.71	89.57	92.49
Uni. Filter	92.14	84.09	89.90	89.22	90.00	90.29	84.03	91.04	84.09	87.67	91.94
Mult. Filter	92.57	90.20	91.24	90.05	91.58	91.74	90.31	92.47	90.20	90.63	92.41
Wr. (Logistic)	92.70	89.01	91.47	90.03	91.47	91.53	89.08	92.53	89.01	89.66	89.01
Wr. (N. Bayes)	92.09	91.23	91.54	89.84	89.01	91.77	91.21	92.16	91.23	91.55	92.16
Wr. (TAN)	92.36	90.76	91.60	89.83	91.72	91.62	90.80	92.33	90.76	90.69	92.27
Wr. (LDA)	92.30	88.23	90.42	91.17	91.35	91.56	88.34	91.74	88.23	89.57	92.35
Wr. (Fusion)	92.39	91.05	91.29	90.58	91.91	91.24	91.05	92.68	91.05	90.88	92.44
Wr. (Stacking)	92.55	89.42	91.66	89.86	91.38	92.20	89.45	92.46	89.42	89.97	92.41
Wr. (Bagging)	92.53	90.77	91.44	89.45	91.58	91.78	90.75	92.70	90.77	90.90	92.54
Wr. (R. Forest)	92.38	90.66	91.27	89.72	91.58	91.53	90.65	92.84	90.66	90.33	92.56
Wr. (Boosting)	91.12	80.83	89.60	88.85	89.39	89.69	80.89	91.11	80.83	89.48	91.42
Wr. (NBTree)	92.21	91.22	91.24	90.56	91.89	91.17	91.22	92.46	91.22	91.27	92.27
Wr. (LMT)	92.45	84.75	91.02	90.32	91.22	91.30	84.80	92.28	84.75	89.82	92.52

Table 6: Scores of all classifiers with all obtained datasets (percentage of correctly classified instances).

not allow to rescale the size of the graph representation.

Inspecting Table 6 we can deduce that in most cases the reduced datasets obtain similar scores to the original dataset. This is because there are variables that provide no additional information, in other words, there are dependent or irrelevant variables in the dataset. As stated in the Problem Description section, there are some variables that are calculated from others. For instance, the *Roundness* is obtained from the *Area* and the *Perimeter*, and the *Compactness* from the *Equivalent diameter* and the *Major axis length.* Additionally, there are some attributes that are fundamentally related, such as the *Area* and the *Perimeter.* The most selected variables are the *Perimeter*, the *Roundness*, and the *Shape Factor 1.*

Looking at Table 5, both the Univariant Filter and the Wrapper approach with the RIPPER classifier selected 8 features. However, as Table 6 shows, the scores are better in the latter. This means that there are some features selected by the Wrapper that have not been selected by the Univariant Filter that are *significant.* For example, the Wrapper selected *Shape Factor 3 and 4* and *Solidity, Compactness and Roundness*, which were not selected by the Filter. This means that these attributes provide relevant information.

The rest of the reduced datasets obtained similar scores compared to the original dataset. The cause of this is the redundancy of some variables. However, the scores are not close to 100%, probably because the dataset contains multiple irrelevant attributes, and the isolated *important* features need more complex models to fit the distribution.

Table 8 shows the confusion matrix obtained by the Multilayer Perceptron with the original dataset (the best precision achieved). In this scenario, all instances of the class *BOMBAY* are classified correctly, this is probably due to the fact that the class has a well-defined boundary that could be fitted perfectly by this classification model. In

Dataset	Logistic	Naive Bayes	TAN	LDA	Fusion	Stacking	Bagging	Random Forest	Boosting	NBTree	LMT
Original	57.5	0.02	0.14	0.02	0.17	1.67	0.19	3.99	1.17	11.51	14.38
Uni. Filter	2.39	0.01	0.06	0.01	0.07	0.7	0.09	3.1	0.61	5.83	10.58
Mult. Filter	3.89	0.01	0.09	0.01	0.11	1.06	0.15	3.18	0.84	7.17	11.41
Wr. (Logistic)	5.81	0.01	0.09	0.01	0.11	1.06	0.13	3.21	1.14	10.54	12.27
Wr. (N. Bayes)	1.37	0.01	0.03	0.01	0.04	0.45	0.08	2.38	0.46	1.92	8.39
Wr. (TAN)	2.18	0.01	0.06	0.01	0.07	0.74	0.1	3.14	0.51	6.52	9.58
Wr. (LDA)	1.99	0.01	0.04	0.01	0.07	0.53	0.08	2.48	0.39	2.9	13
Wr. (Fusion)	1.97	0.01	0.05	0.01	0.06	0.63	0.09	2.46	0.58	5.23	9.86
Wr. (Stacking)	2.69	0.01	0.07	0.01	0.09	0.92	0.12	3.21	0.99	8.71	10.74
Wr. (Bagging)	2.44	0.01	0.07	0.01	0.09	0.82	0.12	3.17	0.67	5.63	10.57
Wr. (R. Forest)	2.51	0.01	0.07	0.01	0.09	0.82	0.11	3.21	0.87	8.61	10.33
Wr. (Boosting)	0.97	0.01	0.03	0.01	0.03	0.36	0.06	2.18	0.46	1.43	8.26
Wr. (NBTree)	0.96	0.01	0.04	0.01	0.06	0.53	0.09	2.44	0.45	3.96	9.05
Wr. (LMT)	7.95	0.01	0.06	0.01	0.08	0.83	0.11	3.28	0.56	8.26	16.41

Table 7: Training time in seconds.

contrast, the other classes had some instances wrongly classified because the algorithm could not fit to the boundaries between each other.

Classified as →	a	b	c	d	e	f	g
a = SEKER	1929	8	0	0	0	55	35
b = BARBUNYA	10	1207	0	74	5	25	1
c = BOMBAY	0	0	522	0	0	0	0
d = CALI	3	47	0	1535	29	16	0
e = HOROZ	0	3	0	25	1843	44	13
f = SIRA	31	12	0	7	34	2291	261
g = DERMASON	44	1	0	0	8	191	3302

Table 8: Confusion matrix of the MLP trained with the original dataset.

6 Conclusion

In this practical application, a dataset of dry beans with 13,611 instances and 17 features (including the class) was analysed with different attribute selection methods and different classifiers with a 10-fold cross-validation test approach. The dataset contains instances of seven different classes. The only pre-processing needed was the normalization of the dataset, since there are no missing values or nominal variables (except the class column).

Different datasets were obtained from the original by extracting features with a Univariate Filter, Multivariate Filter, and different Wrapper approaches. All of which were tested with the following classifiers: k-Nearest Neighbour, RIPPER, Support Vector Machine, Multilayer Perceptron and C4.5.

Overall, the precision obtained is similar in most cases, the best ones obtained by the MLP, SVM, and kNN with the original dataset. Generally, both SVM and MLP obtained better scores than RIPPER and C4.5 due to the ability to fit more complex data distributions. However, the impact of the complexity of the models is reflected in the training time.

The k-Nearest Neighbour classifier performed the best when the execution time is taken into consideration. It obtained better scores than the RIPPER and C4.5 algorithms, while consuming less computational time. Note that the optimization of each model (finding the best combination of hyperparameters) is not part of this assignment, but it could be an interesting subsequent work.

On the other side, in most cases the datasets obtained by selecting features did not obtain better precision than the original dataset. Nevertheless, the reduction of attributes did reduce the training time approximately by half in the Multilayer Perceptron and C4.5 classifiers. Some attributes, for instance the *Perimeter* and the *Roundness* were selected by most feature selection methods, which denotes the importance of these.

In summary, the dataset used in this practical application contains variables that are not needed to achieve similar scores to the original dataset. The best precision was obtained by the Multilayer Perceptron, which can fit more complex data compared to other classification algorithms. However, the k-Nearest Neighbour algorithm achieved similar scores while spending significantly less computational time to train the model.