INDIANA UNIVERSITY
BLOOMINGTON

EAR4 Manual
for Version 1.0

Vahid Jalali
David Leake

October 14, 2014

Indiana University, Bloomington, Indiana

1 What is EAR4?

EAR4 is a lazy learner introduced by Jalali and Leake [1]. It applies the Case-Based Reasoning [2] paradigm to regression (i.e. numerical prediction) tasks. The main idea of EAR4 is very similar to that of IBk [3]. However, instead of directly using the solutions of the top k nearest neighbors to predict the value of the input query, EAR4 adjusts the nearest neighbors' values before using them in building the solution. Fig. 1 depicts the generic process of Case-Based Regression.

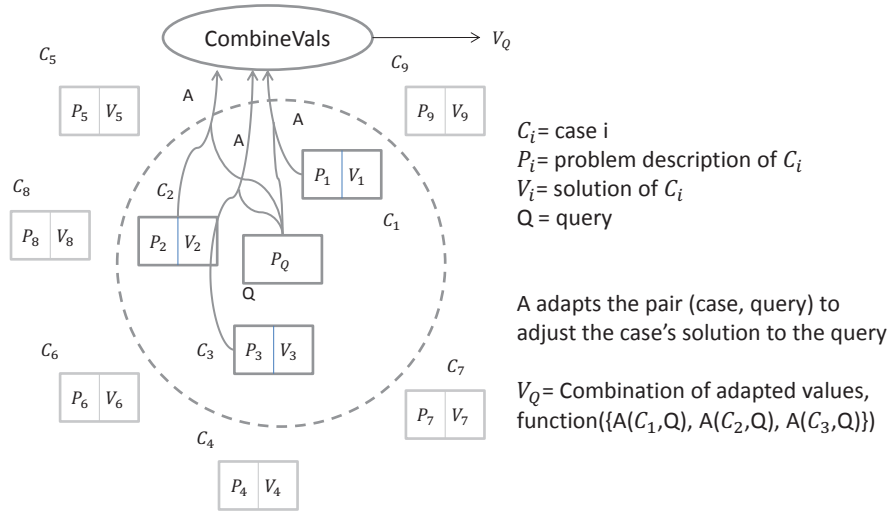

Figure 1: Illustration of the generic case-based regression process

As it can be seen, Case-Based Regression first adjusts the values of the retrieved cases and then combines the adjusted values to build the final estimation.

EAR4 utilizes an Ensemble of adaptation rules for adjusting the values of the nearest neibhbors (also referred to as base cases). These adaptation rules are built using *Case Difference Heuristic* [4] which is a method for deriving adaptation rules by comparing pairs of cases in the case base. The differences in the input features of a pair of cases form the antecedent part of the adaptation rule and the difference in their solutions form the consequent. Fig. 2 depicts a simple rule generation and application scenario for the automobile MPG (Mile Per Gallon) estimation task.

Part a of Fig. 2 shows how an adaptation rule is generated based on a pair of cases and part b shows the application of the generated adaptation rule in part a, for adjusting the value of a nearest neighbor. Adaptation rules can be generated from different parts of the domain. EAR4 generates adaptations by comparing the top nearest neighbors of the input query enabling lazy learning of adaptation and postponing processing until a query is submitted to the system.

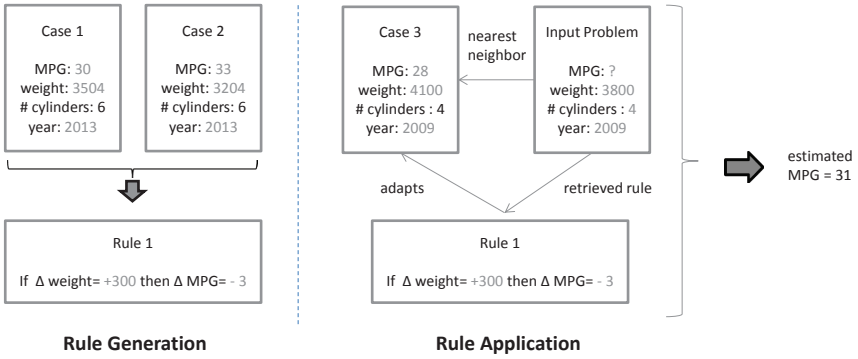

Figure 2: Illustration of using the case difference heuristic to generate an adaptation rule and to generate an MPG estimate.

2 How to Use EAR4 in Weka

EAR4's Weka plugin currently works for domains with numeric input features and target values. The source code of EAR4 can be downloaded from EAR4 source code. You can also download an executable weka jar (version 3.6.11) file with EAR4 bundled in it at Weka + EAR4 executable jar. If you want to modify the source code or compile it, you can follow the instructions provided at How to complie Weka.

Once you have the Weka jar file with EAR4 learner, you can launch the jar file and you will see Weka GUI as depicted in Fig. 3.

Figure 3: Weka GUI.

Next you should select the Explorer button and Weka Explorer will be launched as depicted in Fig. 4. Select the open file button in Weka Explorer and choose the the data set to be tested.

Next choose the Classify menu in Weka Explorer and you will see the available classifiers as depicted in Fig. 5. You will find EAR4 under lazy classifiers category.

Choose EAR4 as the classifier. Now you can tune the parameters used by EAR4 by clicking on the text box next to the choose button as depicted in Fig

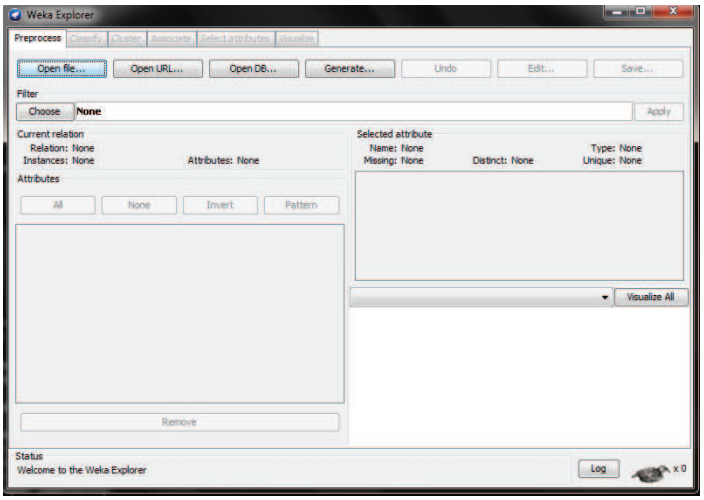

Figure 4: Weka Explorer.

6.

As you can see in Fig 6 you can tune different parameters for EAR4. kNN specifies the number of nearest neighbors that should be used in estimating the solution. l denotes the number of adaptation rules to be applied for adjusting the value of each neareset neighbor (base case). o is a coefficient used for determining the neighborhood used for adaptation generation. EAR4 generates adaptations based on the local neighborhood of the input query. o is a coefficient used for determining the number of nearest neighbors of the input query from which the adaptations should be generated. The number of nearest neighbors of the input query from which adaptations are generated is derived by multiplying kNN and o. The default value of o is one, which means by defualt the adaptation rules are built based on the the top kNN nearest neighbors (i.e. base cases) of the input query. However, it makes sense to use values greater than one for o in practice to make EAR4 more flexibile. $nearestNeighbourSearchAlgorithm$ and $ruleNearestNeighbourSearchAlgorithm$ also show the similarity measures used for retrieving the base cases and the adaptations respectively. Figure 6 shows a sample valuations of EAR4 parameters.

In this case, EAR4 estimates the input query's value by combining the adjusted values of the five nearest neighbors of the input query. While the value of each of those nearest neighbors is adjusted by applying five adaptation rules. The adaptation rules are built from the top 50 (i.e. 5×10) cases of the input query.

Optimal values of EAR4's parameters can be tuned by hill climbing on the training data using cross validation. However, this feature is not currently implemented and it is up to the user to find and set the optimal tuining for the learner in practice.

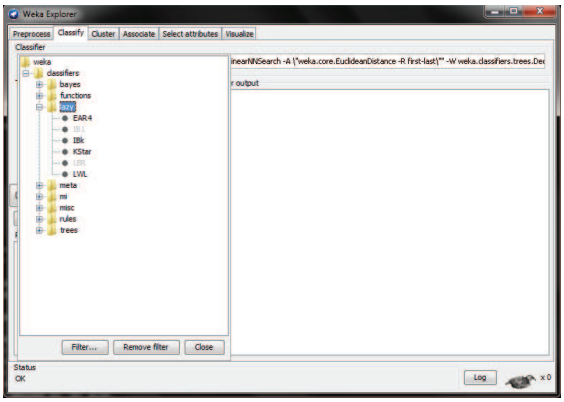

Figure 5: Weka Classifiers.

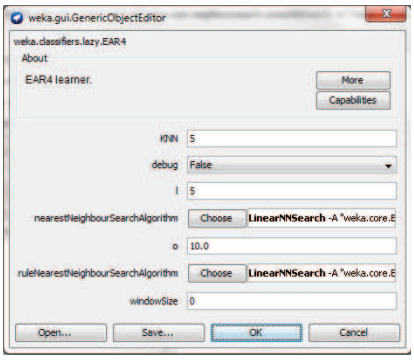

Figure 6: Setting EAR4's parameters.

References

[1] Jalali, V., Leake, D.: Extending case adaptation with automatically-generated ensembles of adaptation rules. In: Case-Based Reasoning Research and Development, ICCBR 2013, Berlin, Springer (2013) 188–202

[2] Lopez de Mantaras, R.; McSherry, D.; Bridge, D.; Leake, D.; Smyth, B.; Craw, S.; Faltings, B.; Maher, M.; Cox, M.; Forbus, K.; Keane, M.; Aamodt, A.; and Watson, I. 2005. Retrieval, reuse, revision, and retention in CBR. *Knowledge Engineering Review* 20(3).

[3] Aha, D., Kibler, D.: Instance-based learning algorithms Machine Learning (1991) 37–66

[4] Hanney, K., Keane, M.: The adaptation knowledge bottleneck: How to ease it by learning from cases. In: Proceedings of the Second International Conference on Case-Based Reasoning, Berlin, Springer Verlag (1997) 359–370