Case-Based Explanation of Non–Case-Based Learning Methods

Rich Caruana PhD, Hooshang Kangarloo MD, John David N. Dionisio PhD, Usha Sinha PhD, David Johnson MS

Department of Radiological Sciences
University of California, Los Angeles

We show how to generate case-based explanations for non–case-based learning methods such as artificial neural nets or decision trees. The method uses the trained model (e.g., the neural net or the decision tree) as a distance metric to determine which cases in the training set are most similar to the case that needs to be explained. This approach is well suited to medical domains, where it is important to understand predictions made by complex machine learning models, and where training and clinical practice makes users adept at case interpretation.

1. Introduction

When machine learning is used in medical domains, usually it is important that the models that are learned are able to explain their reasoning. Models that cannot explain their reasoning make it harder for users to understand and verify the model, determine when the model's reasoning is in error, and learn *from* the model. Models that cannot explain their reasoning are less likely to be accepted by users.

One of the advantages of case-based methods such as k-nearest neighbor [1] is that case-based methods can explain their reasoning by presenting to users the cases in the case-base (the training set) that are most similar to a new case that needs to be explained. This allows users to assess the quality of the model's prediction by applying their own expertise to the new case while also seeing relevant empirical data from the training set.

Learning methods such as artificial neural nets or decision trees sometimes yield better predictive accuracy than case-based methods. Unfortunately, neural nets and large decision trees are relatively opaque; their complexity makes their reasoning difficult to understand and evaluate. This poses a serious obstacle to using these methods in medicine.

We present a method that allows case-based explanation to be applied to models that are not themselves case-based. The method uses the trained model (e.g., the neural net or the decision tree) as a *distance metric* for k-nearest neighbor. The cases in the training set that the trained model considers most similar to the case needing explanation are returned as the model's explanation for that case.

In Section 2 we briefly review case-based learning methods such as k-nearest neighbor. In Section 3 we review how to provide explanations for case-based methods. In Section 4 we highlight the difficulty of providing explanations for non-case-based methods such as artificial neural nets or large decision trees. In Section 5 we show how to sidestep this difficulty by using non-case-based models such as neural nets or decision trees as a distance metric for case-based explanation. The strengths and weaknesses of this approach are discussed in Section 6.

2. Case-Based Learning

Consider a training set consisting of N training cases, $X^1 \ldots X^N$. Each of the training cases X^i can be treated as a vector consisting of M input features X^i_1-X^i_M, and a label (outcome) Y^i. The input features are the input measurements for each patient such as age, gender, blood pressure, and white blood cell count. The label (outcome) Y^i can be discrete or continuous.

The principal goal in case-based reasoning is to use the cases in the training set to predict an outcome Y^{test} for a test case from the M input features for that test case, X^{test}_1-X^{test}_M. The basic assumption made by case-based methods is that similarity in input features correlates with similarity in outcomes.

Nearest neighbor (a.k.a. 1-nearest neighbor–1NN) is one of the simplest case-based methods. 1NN finds the case in the training set whose input features are most similar to the input features of the test case, and returns as a prediction for that test case the outcome from that most similar case. Similarity in the input features is measured using a distance metric defined

1091—8280/99/$5.00

on those input features. One of the simplest distance metrics is unweighted Euclidean distance:

$$\text{Euclidean Distance (A,B)} = \text{Sqrt}\left(\sum_1^M (X_i^A - X_i^B)^2\right)$$

In unweighted Euclidean distance, the distance between two cases A and B, is the sum of the squared differences between each of the M input features taken one at a time. More complex distance metrics such as weighted Euclidean distance, which weights each input dimension with a weight W_i, often yield somewhat better performance, though learning what weights to use for each input dimension can be difficult [2].

1NN searches through the training set to find the one training case closest to the test case, and returns its label as the prediction for the test case. 1NN is effective if there is little noise in the input features and output labels, and if the training sample is large. It is less effective when there is noise in the input features or labels, or when the training set is small.

A generalization of 1NN that often yields improved performance is k-nearest neighbor (KNN). KNN uses the same distance metrics as 1NN to find the K training cases closest to the test case. The prediction KNN returns for the test case is the majority label of these K nearest neighbors (or a weighted average of the K labels if it is a continuous prediction problem).

3. Case-Based Explanation

Case-based learning methods do not learn a model that can be used independent of the training set. Instead, they use the training set and distance metric to make predictions. This is quite different from non-case-based methods such as artificial neural nets or decision trees which train a model (the neural net or decision tree) on the training set, and then use only that model to make predictions for new test cases. *The training set is ignored once the model is trained.*

Because case-based methods do not discard the training set, and determine which cases in the training set are similar to the test case, a natural way to explain predictions made by case-based methods is to present the similar cases in the training set to the user. This approach to explanation accomplishes two things. It explains the distance metric used by the case-based method by showing the user what cases the method considers to be most similar to the test case. And it explains the prediction the case-based method makes by presenting to the user the outcomes of cases similar to the test case.

Because medical training and practice emphasizes case evaluation, most medical practitioners are adept at understanding explanations provided in the form of cases. The user can apply their own expertise to judge how similar the selected training cases are to the test case, and how the outcomes in the training set might influence the decision for that test case.

4. Explaining Non-Case-Based Methods is Harder

Providing explanations for predictions made by non-case-based machine learning methods such as artificial neural nets (ANNs) or decision trees (DTs) is more difficult. The problem is that most machine learning methods discard the training set after the model has been trained, using only the learned model to make predictions for new test cases. In order to understand the model's predictions, one has to understand that model.

Understanding a neural net, or a complex decision tree, can be difficult. A typical ANN contains thousands of real-valued parameters that interact in a complex, and intentionally non-linear, way. Trying to understand an ANN model is exceedingly difficult and is the subject of much active research [3].

Decision trees usually are easier to understand than ANNs because decision trees contains fewer real-valued parameters, use input features one at a time, and because decision tree splits are more intelligible than the weighted sums of inputs passing through non-linear squashing functions as in ANNs. Large decision trees, however, can be nearly as unintelligible as ANNs. For example, decision trees trained to recognize chess end positions can grow so large that no human can understand them or verify their correctness [4]. Similarly, decision trees we trained to predict Cesarean section sometimes contain more than 1000 leaf nodes. These trees are too complex to easily be understood or verified by human experts.

One advantage decision trees have over ANNs is that each case is classified into a single leaf node in the tree. The sequence of feature tests on the path to the leaf node can be presented to the user as a rule. Many rules generated this way, however, are complex enough, and appear unnatural enough, that to understand the rule the expert needs to understand the alternate paths in the decision tree. The user may not need to understand the entire tree in order to understand the prediction made for one case, but in practice they often need to understand a significant fraction of the decision tree, and this can be difficult.

5. Using the Learned Model as a Distance Metric

If the artificial neural nets and decision trees trained with machine learning are unintelligible to users, how can we safely employ them in critical applications such as health care? One approach is to focus not on understanding the models themselves, but on understanding the predictions the models make. With case-based methods, practitioners often are satisfied with explanations that consist of cases that the model judges to be most similar to the test case. Practitioners might be satisfied with similar explanations for ANNs and DTs.

Consider the artificial neural net shown in Figure 1. This net is trained to predict pneumonia mortality from 65 demographic, history, physical, and hospital measurements [5]. The ANN has 65 inputs, 64 hidden units, and one output, for a total of 4,289 weights. The complexity of the ANN makes it difficult to understand the model and the predictions it makes.

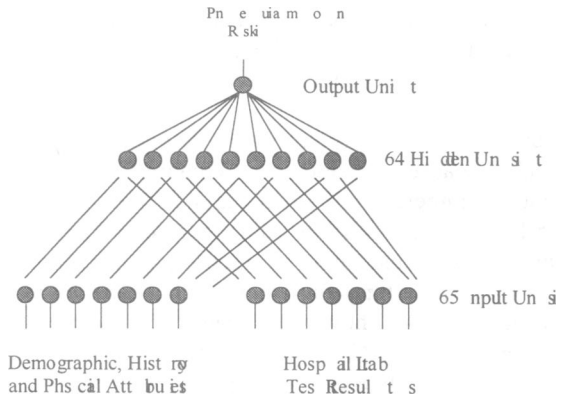

Figure 1. Artificial Neural Net to Predict Pneumonia Mortality Using the Available Measurements

The hidden units in an ANN are internal features that the net learns as it is trained. All of the information from the inputs that the net uses to make predictions at its output passes through the hidden units. The pattern of activation on the hidden layer indicates how the ANN models a case applied to its inputs.

If we record the pattern of activation at the hidden units for each case in the training set, we can compare the patterns of activation for the training cases with the hidden unit activation for a new case that needs to be explained. The cases in the training set that have patterns of hidden unit activation most similar to a test case that needs explaining are the cases in the training set that the ANN model considers most similar to the test case.

We can judge similarity in the pattern of hidden unit activation by applying Euclidean distance to the hidden unit activations. Small Euclidean distance between hidden unit activations suggests that the cases are modeled similarly by the ANN. By keeping the training set after the ANN is trained, and presenting those training cases to the user that have hidden unit activation similar to the test case, we can explain the ANN's reasoning for that test case with the same method used for case-based methods.

A similar method can be applied to decision trees. Decision trees do not have hidden units, but they do provide a strong indication of which training cases they consider similar to a test case. In a decision tree, all cases that end up in the same leaf node are treated the same by the tree. By keeping the training set after the decision tree is trained, and returning to the user the training cases in the leaf node where the test case that needs explaining ends up, we can explain the decision tree's reasoning for that test case. The decision tree acts as a distance metric that identifies cases in the training set with zero "decision tree" distance to the test case.

6. Discussion

Using the learned model as a distance metric for case-based explanation shows the user the cases in the training set most similar to the case being explained *from the model's point-of-view*. The user does not learn how the model reasons internally, but the user does see what the consequences of the model's internal reasoning are.

One might be tempted to use Euclidean distance on the input features, instead of on the learned model, to provide explanations. (In other words, one might be tempted to use regular case-based explanations for predictions made by a non-case-based method.) While this might be effective at explaining the training set to the user, it does not help the user understand the predictions of the model trained on that training set that is making the predictions. For example, the trained model might ignore (or provide little weight to) some input features, while giving all or most of the weight to other input features. Using Euclidean distance on the input features as the distance metric for explanation, instead of using the machine learning model, fails to capture how the trained model uses the features. To explain what the model has learned, we must use the model in the distance metric.

When generating case-based explanations for ANNs, weighted Euclidean distance can be used to insure that the hidden units that are most important to the net's prediction at its output receive more weight in the distance metric. The weights for the weighted Euclidean distance metric can be estimated from the output layer of the ANN. A large positive or negative connection from the output to a hidden unit suggests that the hidden unit is important to the net's prediction, whereas a connection strength near zero indicates that the hidden unit is not important to the net's predictions. Somewhat more accurate weights can be estimated by also taking into account the average activation of the hidden units.

One consideration when applying case-based explanation to non-case-based methods is how many cases to return to the user. If the model being explained is a decision tree, returning more cases than available at the leaf node may be misleading: cases from other leaf nodes are different to the tree. It is safest not to return more cases than available at the leaf node unless the user requests them.

More often, the problem when explaining decision trees is that there are too many cases available at the leaf node to give to the user. Which of the available cases do we return when there are too many? One approach is to return two sets of cases from the leaf node. One set is the cases at the leaf node that are most similar to the test case as measured by Euclidean distance on the inputs. The other set is the cases at the leaf node that are least similar to the test case, also as measured by Euclidean distance on the inputs. This allows the user to see the outcomes of the cases most similar to the test case. But it also allows the user to see the cases most different from the test case that the model treats the same way, so the user can judge the model's quality.

One disadvantage of applying case-based explanation to non-case-based methods is that supplying cases to the user does little to help the user understand the model as a whole because the user gets explanations for one prediction at a time. In the absence of an effective way to help users understand the entire model, this may be the best that can be achieved.

One might be tempted to dismiss machine learning methods that yield unintelligible models. This would be unfortunate, as some of the methods that learn the least intelligible models yield the best performance [5]. Complex models often are more accurate. Applying case-based explanation to opaque non-case-based machine learning methods is one approach for using opaque models while still requiring that these opaque models provide explanations.

We are evaluating the use of case-based explanations with non-case-based learning methods in three medical domains. The first is to provide explanations for artificial neural nets trained to predict pneumonia risk (such as the one shown in Figure 1) [5]. The second is to provide explanations for decision trees trained to predict Cesarean section. The third is to provide explanations for a system that learns to predict imaging protocols for stable patients before they see their primary care physician.

7. Summary

Because medical training and practice emphasizes case evaluation, most medical practitioners are adept at understanding explanations provided in the form of cases. We present a method that allows one to provide case-based explanations for complex learned models that are not themselves case-based. The method uses the trained model (e.g., the neural net or the decision tree) as a distance metric for k-nearest neighbor. The cases in the training set that the trained model considers most similar to the case needing explanation are returned as the explanation for that case. We are evaluating this method of applying case-based explanation to non-case-based learning methods in three different medical domains.

References

[1] Kahn CE Jr. Artificial Intelligence in Radiology: Decision Support Systems. RadioGraphics 1994; 14:849–861.

[2] Atkeson CG, Schaal SA, Moore AW. Locally Weighted Learning. AI Review 1997; 11:11-73.

[3] Craven M, Shavlik, J. Using Sampling Queries to Extract Rules From Trained Neural Nets. 11th International Conference on Machine Learning 1994; 37–45.

[4] Stiller L, Kasif A. Some chess end-games require more than 220 moves. Johns Hopkins 1995.

[5] Cooper G, Aliferis C, Ambrosino CF, Aronis J, Buchanan, B, Caruana R, Fine M, Glymour C, Gordon G, Hanusa BH, Janosky J, Meek C, Mitchell T, Richardson T, Spirtes P. An Evaluation of Machine Learning Methods for Predicting Pneumonia Mortality. AI in Medicine 1997; 9: 107–138.