Computational Intelligence

AN INTERNATIONAL JOURNAL

Computational Intelligence, Volume 00, Number 0, 2012

AGGREGATE CERTAINTY ESTIMATORS

KRISTINE MONTEITH AND TONY MARTINEZ

Department of Computer Science, Brigham Young University, Utah, USA

Selecting an effective method for combining the votes of base inducers in a multiclassifier system can have a significant impact on the system's overall classification accuracy. Some methods cannot even achieve as high a classification accuracy as the most accurate base classifier. To address this issue, we present the strategy of aggregate certainty estimators, which uses multiple measures to estimate a classifier's certainty in its predictions on an instance-by-instance basis. Use of these certainty estimators for vote-weighting allows the system to achieve a higher overall average in classification accuracy than the most accurate base classifier. Weighting with these aggregate measures also results in higher average classification accuracy than weighting with single certainty estimates. Aggregate certainty estimators outperform three baseline strategies, as well as the methods of modified stacking and arbitration, in terms of average accuracy over 36 data sets.

Received 16 March 2011; Revised 20 August 2011; Accepted 22 January 2010

Key words: combining classifiers, certainty estimators, multiclassifier systems, ensembles, supervised learning.

1. INTRODUCTION

In the late eighteenth century, the Marquis de Condorcet composed the *Essay on the Application of Analysis to the Probability of Majority Decisions*. It outlined the concept now known as the "Condorcet Jury Theorem," the idea that, if each member of a group has a greater than 50% chance of making a correct decision, the probability that a plurality of voters will make the correct decision increases as voters are added. The essay was originally intended to provide a theoretical argument for the benefits of democracy. However, the concept also has application in the field of supervised learning. In theory, a group of classifiers should be better suited to the task of classification. The base classifiers need not even be highly accurate. As long as each classifier can exhibit a better-than-random performance, an ensemble of these classifiers should be able to take advantage of the expertise of each to assign more accurate classifications. Schapire (1990) demonstrated how a collection of "weak" classifiers—ones that perform only slightly better than random guessing—can be combined to produce a classifier with arbitrarily high accuracy. This provided the theoretical basis for his well-known AdaBoost algorithm, (Freund and Schapire 1996; Schapire and Singer 1998). Other researchers have proposed similar ensemble-creation strategies, ranging from the simple strategy of bagging (Breiman 1996) to random forests (Breiman 2001) and Bayesian model averaging (Hoeting et al. 1999).

Rokach (2010) outlines four basic building blocks of an ensemble: a labeled training set, a base inducer, a diversity generator, and a combiner. A number of strategies can be used in selecting the base inducers and generating diversity. For example, with the adaptive mixture of local experts strategy (Jacobs et al. 1991), a gating network determines the probability of selecting the output of one of the base inducers. Delegating (Ferri, Flach, and Hernandez-Orallo 2004) is an approach where a base inducer assigns the final class label to a given instance only if it has highcertainty in that particular class. If it is less confident, the instance is delegated to another base inducer. Diversity can be generated by selecting different subsets of the training set for use in training each of the base inducers.

Address correspondence to K. Monteith and T. Martinez, Department of Computer Science, Brigham Young University, UT 84602, USA; e-mail: kristinemonteith@gmail.com and martinez@cs.byu.edu.

With bagging (Breiman 1996), instances are drawn randomly with replacement from the original training set to create a new set with which to train the base inducers. The AdaBoost algorithm (Freund and Schapire 1996; Schapire and Singer 1998) takes into account which instances were misclassified by previously constructed base inducers when selecting data for training subsequent ones. Multiclassifier systems address the problem of diversity generation by using different algorithms for training their base inducers. For example, Ho, Hull, and Srihari (1994) use four different algorithmic approaches to character recognition and discuss methods of combining their outputs.

This work focuses on the fourth aspect outlined by Rokach: combining the outputs of the base inducers. It investigates the possibility that the same principle of combining weak classifiers to produce a strong one can also be applied to the weighting strategies used when combining the votes of those classifiers. While, as in the case of Naïve Bayes, there is often a standard method for estimating confidence in a classifier's predictions, certainty in classification can be estimated in a variety of ways. If each of these methods can demonstrate even a slight tendency for assigning higher certainly values to correctly classified instances as opposed to incorrectly classified instances, they could theoretically be combined to create a more accurate measure of certainty, much as weak inducers in an ensemble can be combine to form a stronger classifier.

These multiple measures are incorporated into the strategy of aggregate certainty estimators. With this technique, the votes of classifiers are weighted by certainty as determined on an instance-by-instance basis. Each classifier is trained using a different algorithm on the same training data set. Then, each instance in the test set is assigned a class value and an overall certainty rating for that classification by each of n classifiers. Multiple factors are taken into consideration when determining this overall certainty rating. For example, six different certainty estimators are used to calculate certainty in the prediction of a decision tree classifier. A given instance would receive six certainty ratings, reflecting properties such as the purity of the leaf node in which it was classified and the number of instances classified at that leaf. These six numbers are then aggregated to produce an overall certainty rating for the decision tree's classification of this particular instance. A similar method is used to calculate an overall certainty rating for each of the classifiers. The class label assigned to the instance is then calculated by summing the weights for each possible label and selecting the class label with the maximum total.

The technique of aggregate certainty estimators is shown to achieve higher average classification accuracy over 36 data sets than the standard combination strategies employed by Bagging and Boosting as well as the SelectBest strategy of allowing the most accurate classifier in the system to make all the classifications. It also outperforms Arbitration (Ortega, Koppel, and Argamon 2001) and the Stacking algorithm presented by Dzeroski and Zenko (2004) in terms of average classification accuracy.

Section 2 of this work provides an overview of related research. Section 3 outlines the aggregate certainty estimators algorithm. Section 4 presents certainty estimators for five common classification algorithms. Section 5 provides results comparing aggregate certainty estimators with standard voting, voting by accuracy, the SelectBest strategy, Arbitration, and Modified Stacking. Section 6 outlines conclusions and suggests options for further research.

2. RELATED WORK

One common method of combining votes of base classifiers on an instance-by-instance basis is to use posterior class probabilities (Rokach 2010). However, most traditional classifiers are not naturally designed to output these probabilities. Thus, statistical or heuristic estimates of how certain a given classifier should be in its classification of an instance can prove useful.

For example, Provost and Domingos (2003) found that using a simple Laplace correction improves probability-based rankings. Specific pruning strategies and bagging techniques also resulted in ranking improvement. Ferri, Flach, and Hernandez-Orallo (2003) found that the performance of these trees could further be improved by using different splitting criteria and a new smoothing technique that considers all the nodes along the classification path from root to leaf. These types of techniques have also been applied to rule-based classifiers. For example, Dzeroski, Cestnik, and Petrovski (1993) used the m-estimate to smooth probabilities and make predictions more effective.

While Naïve Bayes classifiers naturally produce a probability distribution for all class values, Domingos and Pazzani (1997) found that the power of the Naïve Bayes classifier lies more in the ordering of the classes than the actual probabilities predicted. They found that the classifier performs surprisingly well, even when the prior assumption of feature independence is clearly not met. Other researchers, He and Xiaoqing (2007), introduce a number of smoothing methods that can improve these probability estimates, resulting in more accurate and stable estimates than those that can be achieved with Laplace smoothing.

Unlike Bayesian models, multilayer perceptrons do not calculate class probabilities explicitly, but Ruck et al. (1990) provide a proof that the activation outputs of a multilayer perceptron approximate these probabilities. Richard and Lippman (1991) also found that the accuracy of these probability estimates depended on the network complexity, the amount of training data, and how well the training data represented true a priori class probabilities.

In addition, while combining votes of base inducers according to class probabilities may be an intuitive method, other methods benefit from taking additional factors into consideration. For example, Dolev, Leshem, and Yagel (2010) focus on attributes of the data set when determining how to weight votes. Their algorithm assumes a percentage of corrupted data, and they statistically analyze the data and attempt to remove corrupted data by identifying outliers in the distributions. The estimated distribution parameters are also used to determine the likelihood of feature values in the training set and a corresponding certainty level for leaf nodes in a decision tree where these training set instances are classified. Carney, Cunningham, and Bhagwan (1999) focus on the variances among the distributions of base classifier output when determining how votes should be combined. Ali and Pazzani (1996) compare simple voting to three other evidence combination methods. "Bayesian Combination" approximates the optimal Bayes approach, taking into account both accuracy on training data set and posterior probabilities when weighting rule output. The "Distribution Summation" method takes into account the number of training instances covered by a given rule. "Likelihood Combination" takes both coverage and training set accuracy into account when assigning weights.

In most cases in literature, estimates of certainty are calculated using a single measure, and improvements are aimed at finding ways to smooth and improve this single measure's accuracy. Certainty measures may take into account variables such as the inherent properties of a classifier, how classifiers behave in an overall system, or distributions of attributes in a data set. However, given the variety of things that may be considered when estimating certainty of classification, it stands to reason that an algorithm could greatly benefit from taking multiple variables into account. The certainty estimators incorporated in this work include traditional class probability estimators, but also consider other features discussed by researchers such as data set coverage and distributions of base classifier outputs.

To demonstrate the usefulness of our proposed weighting system, we compare the technique of aggregate certainty estimators to other methods that weight votes of base classifiers

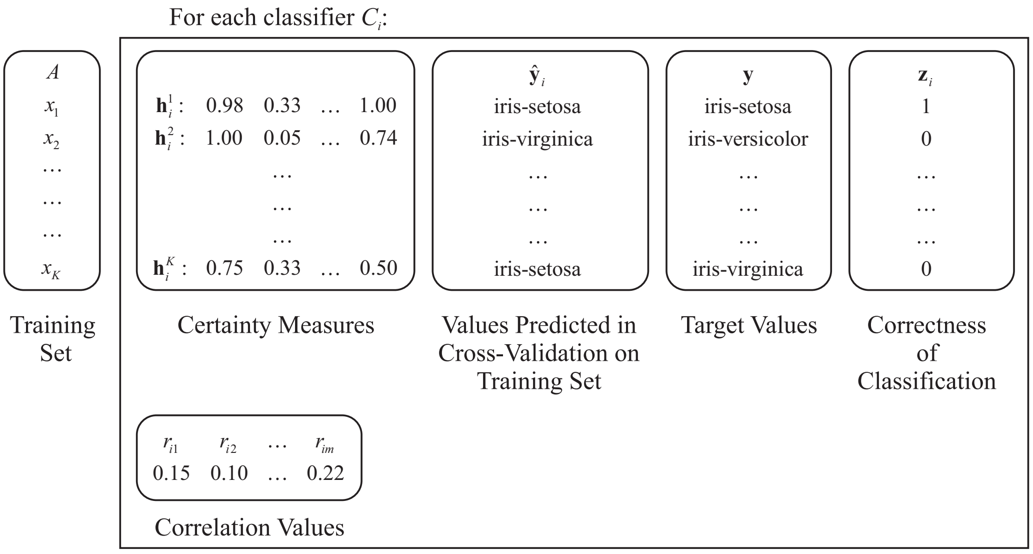

For each classifier C_i:
A
x_1
x_2
x_K
Training Set
$\mathbf{h}_i^1$: 0.98 0.33 … 1.00
$\mathbf{h}_i^2$: 1.00 0.05 … 0.74
$\mathbf{h}_i^K$: 0.75 0.33 … 0.50
Certainty Measures
$\hat{\mathbf{y}}_i$
iris-setosa
iris-virginica
iris-setosa
Values Predicted in Cross-Validation on Training Set
$\mathbf{y}$
iris-setosa
iris-versicolor
iris-virginica
Target Values
$\mathbf{z}_i$
1
0
0
Correctness of Classification
r_{i1} r_{i2} … r_{im}
0.15 0.10 … 0.22
Correlation Values

FIGURE 1. Values calculated for a component classifier during training of aggregate certainty estimators.

on an instance-by-instance basis. Arbitration (Ortega et al. 2001) creates a "referee" to determine the certainty that a learning model has in its classification of the various subdomains of a given problem. Information about both the misclassification of instances and the classifiers themselves is used in the development of the metalearner referees. Stacking (Wolpert 1992) makes use of a metalevel learning algorithm that discovers the best way to combine outputs from the base-level classifiers.

Dzeroski and Zenko (2004) found that the accuracy of an ensemble over a data set is often no better than the accuracy of one of the classifiers contributing to the ensemble. To justify the overhead of creating an ensemble, the ensemble should be able to perform better than a strategy of simply selecting the best classifier by cross-validation. In the algorithms Dzeroski and Zenko explore, only their Modified Stacking strategy was able to consistently achieve this level of performance. We demonstrate that our strategy of aggregate certainty estimators also tends to outperform the single most accurate base classifier in a given ensemble.

3. AGGREGATE CERTAINTY ESTIMATORS

Let $C_1 \ldots C_n$ be classifiers constructed using instances from a training set A. For each classifier C_i, m predefined certainty estimators are used to calculate certainty estimates. For a given instance k in the training set, a vector $\mathbf{h}_i^k$ of m certainty values is calculated, with h_{ij}^k being the value assigned to instance k by the jth certainty estimator of classifier C_i.

For each classifier C_i, let $\hat{\mathbf{y}}_i$ be the predictions of C_i over the training set A, with $\hat{y}_i^k$ being the class label assigned by classifier C_i when instance k appeared in a test fold during cross-validation on the training set. Let $\mathbf{y}$ be a vector of the target values for the training instances, and $\mathbf{z}_i$ be a vector describing $\hat{\mathbf{y}}_i = \mathbf{y}$. In other words, if $\hat{y}_i^k = y^k$, then $z_i^k = 1$; if not, $z_i^k = 0$. For each classifier C_i, let $\mathbf{r}_i$ be a vector of correlation values, where r_{ij} is the correlation between $\mathbf{z}_i$ and $\mathbf{h}_{ij}$. The values in $\mathbf{r}_i$ are then scaled to sum to one. Figure 1 outlines the values calculated for each classifier in the ensemble.

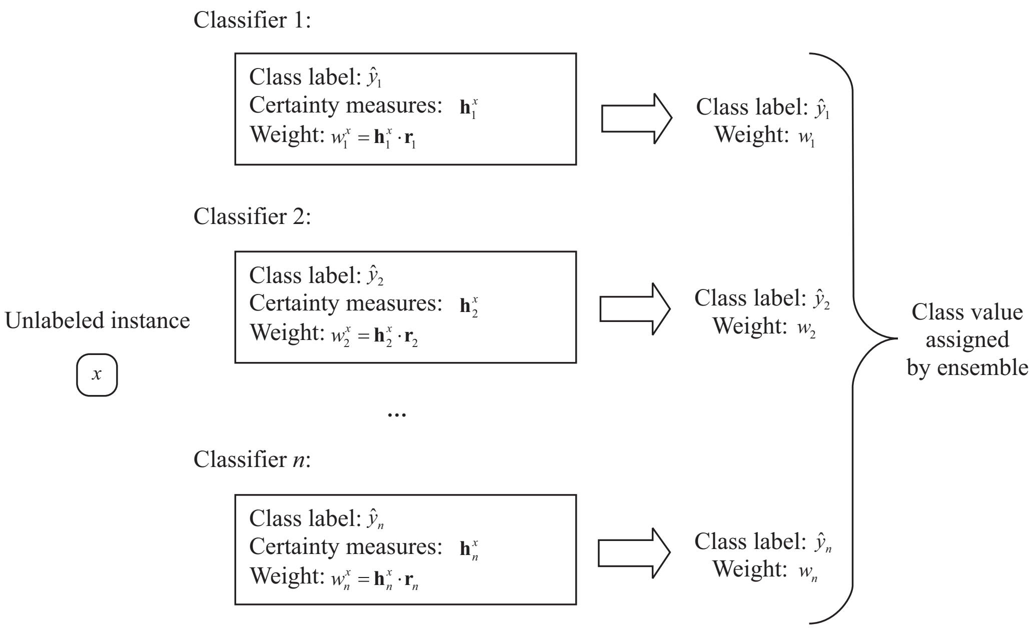

Classifier 1:
Class label: $\hat{y}_1$
Certainty measures: $\mathbf{h}_1^x$
Weight: $w_1^x = \mathbf{h}_1^x \cdot \mathbf{r}_1$
Class label: $\hat{y}_1$
Weight: w_1
Classifier 2:
Class label: $\hat{y}_2$
Certainty measures: $\mathbf{h}_2^x$
Weight: $w_2^x = \mathbf{h}_2^x \cdot \mathbf{r}_2$
Class label: $\hat{y}_2$
Weight: w_2
Unlabeled instance
x
...
Classifier n:
Class label: $\hat{y}_n$
Certainty measures: $\mathbf{h}_n^x$
Weight: $w_n^x = \mathbf{h}_n^x \cdot \mathbf{r}_n$
Class label: $\hat{y}_n$
Weight: w_n
Class value assigned by ensemble

FIGURE 2. Values calculated by aggregate certainty estimators to evaluate an unlabeled instance.

For each unlabeled instance x, let $\hat{y}_i^x$ be the class label assigned to instance x by classifier C_i. Let $\mathbf{h}_i^x$ be a vector of certainty values calculated for the classification of instance x by classifier C_i. These values will be used in determining how much weight the overall ensemble should assign to the classification $\hat{y}_i^x$. To make the values assigned by the various estimators more uniform among the classifiers, h_{ij}^x is normalized using the maximum and minimum values from the vector $\mathbf{h}_{ij}$ of values calculated for training set instances.

Let w_i^x be the dot product of $\mathbf{h}_i^x$ and $\mathbf{r}_i$. This aggregate measure is then used to weight $\hat{y}_i^x$. The class label assigned to $\mathbf{x}$ by the overall ensemble is calculated by summing the weights for each possible label and selecting the class label with the maximum total. Figure 2 outlines the values calculated for unseen instances. The strategy of aggregate certainty estimators is outlined in Figure 3.

As concrete example, six measures were used to describe certainty of classification by a rule-based classifier in our experiments. These measures include such factors as the purity of classification of training set instances covered by the rule and the number of instances covered. These six numbers were then averaged to produce an overall certainty measure. In a similar fashion, overall certainty measures are calculated for each of the five learning algorithms incorporated in the multiclassifier system. These measurements are calculated for both training set and test set instances.

This means that for a training set with 135 instances (e.g., the iris data set using 10-fold cross-validation), a 135×5 matrix would be generated, with one row for every instance in the training set and one column for every classifier incorporated in the overall system. A correlation value would then be calculated between each column of the matrix and a column of “1”s and “0”s that described whether each of the training set instances was correctly classified in cross-validation experiments on the training set (e.g., train a rule-based classifier on

```
1.Train each of $n$ classifiers $C_1...C_n$ using training set $A$.
   A. Determine the following for each classifier $C_i$:
      1. For each instance $k$ in $A$:
         a. Calculate vector $\mathbf{h}_i^k$ of certainty values using $m$ estimators specific to $C_i$
         b. Calculate $\hat{y}_i^k$, the prediction of class label by $C_i$ when $k$ appeared in a test fold
            during cross-validation on $A$
         c. Identify $y^k$, the target value for instance $k$
      2. Define $\mathbf{z}_i$ to be a vector describing $\hat{\mathbf{y}}_i = \mathbf{y}$
      3. Calculate vector $\mathbf{r}_i$ of correlation values where $r_{ij}$ is the correlation between $\mathbf{z}_{ij}$ and $\mathbf{h}_{ij}$
         (Scale each value in $\mathbf{r}_i$: $r_{ij} = r_{ij}/\Sigma_{j=1}^{m} r_{ij}$ so that values sum to one)
2. For an unlabeled instance $x$:
   A. For each classifier $C_i$:
      1. Determine $\hat{y}_i^x$, the class value of $\mathbf{x}$ as predicted by $C_i$
      2. Create vector $\mathbf{h}_i^x$ of certainty values using $m$ estimators specific to $C_i$
         (Scale each value in $\mathbf{h}_i^x$: $h_{ij}^x = (h_{ij}^x - min_{ij})/(max_{ij} - min_{ij})$ where
         $max_{ij}$ and $min_{ij}$ are the maximum and minimum $h_{ij}^k$ values from the training set)
      3. $w_i^x = \mathbf{h}_i^x \cdot \mathbf{r}_i$
   B. Class value for $\mathbf{x} = \text{argmax}_{y \in Y}(\Sigma_{i=1}^{n} \delta(y, \hat{y}_i^x) w_i^x)$
      $\delta(y, \hat{y}_i) = \begin{cases} 1 \text{ when } \hat{y}_i = y \\ 0 \text{ otherwise} \end{cases}$
```

FIGURE 3. Aggregate certainty estimators.

134 test set instances and determine if the resulting classifier could correctly label the 135th instance of the training set[1]).

In our example, five certainty measures would also be calculated for each test set instance. Each measure would then be multiplied by the correlation value for its respective classifier. These values would then be used to weight the predictions from each classifier. For example, assume the following for a given instance:

	Classification	Certainty estimator	Correlation value
Decision tree	iris-setosa	0.93	0.15
Rule-based classifier	iris-setosa	0.84	0.10
Instance-based classifier	iris-virginica	0.23	0.36
Naïve Bayes classifier	iris-virginica	0.87	0.45
Multilayer perceptron	iris-setosa	0.66	0.22

Summing the votes for iris-setosa: $0.93 * 0.15 + 0.84 * 0.10 + 0.66 * 0.22 = 0.3687$. Summing the votes for iris-virginica: $0.23 * 0.36 + 0.87 * 0.45 = 0.4743$. Finding the maximum value, the overall system would assign the label of ”iris-virginica” to this particular instance.

[1] In the experiments described in this paper, hold-one-out cross-validation was used to generate this column, simply to provide a higher degree of accuracy in evaluating the measures. In practice, cross-validation with a low number of folds could generate this column at much less expense in terms of computation time.

4. MULTIPLE CERTAINTY ESTIMATORS

This section contains the information about the certainty estimators used to predict certainty in classifications for each of five different algorithms. The five algorithms used in this work were selected because they are representative of standard classes of algorithms used in machine learning. Many of the certainty estimators presented here could be adapted for use with similar machine learning algorithms. The algorithms used in this work are implemented using Weka open source code (Witten and Frank 2005). Default settings are used for each of the algorithms to allow for easier reproduction of results. These settings allowed for reasonable performance of the base classifiers on the test data sets.

While we have tried to select diverse models to represent the spectrum of machine learning algorithms, the technique of aggregate certainty estimators could be applied to ensembles with any number and type of base-level classifiers. The systems of the five base-level classifiers discussed here are designed simply to present the concept. One classifier of each type is used for parsimony and to avoid skewing the ensemble in favor of any particular classifier.

Efficacy of the various certainty estimators is evaluated using 36 data sets taken from the UCI Repository (Hettich, Blake, and Merz 1998). Table 1 provides information about these data sets. Data sets were selected so as to achieve variety in number of instances, attributes, attribute types, and output classes. The data sets range from 90 to 2,310 instances, 5 to 70 attributes, and 2 to 24 output classes. Roughly, a third of the data sets feature discrete attributes, another third have real-valued attributes, and the remaining data sets have a mixture of discrete and real-valued attributes. Ten of the data sets contain missing values. In the case of discrete attributes, missing values were replaced by the most common value for the given attribute. For data sets with real-valued attributes, unknown values were replaced with the average value for the attribute.

To evaluate the certainty measures for each of the algorithms studied, 10-fold cross-validation experiments were conducted for each of the data sets. Instances were marked as correctly or incorrectly classified based on the classifier's ability to classify the instance when it appeared in the test set. This correctness of classification is compared to the certainty measure assigned to each instance by each of the certainty estimators. Please note that the correct/incorrect labels assigned to test instances and used for the purposes of evaluating the certainty measures are independent of the correct/incorrect labels assigned to instances during training.

Just as base inducers must exhibit at least slightly better-than-random performance to provide benefit to an ensemble, we specify a baseline measure of performance for our certainty estimators.

Graphs such as the one in Figure 4 were constructed for each of the measures studied. This graph shows the number of instances receiving a given certainty value that were correctly and incorrectly classified. While real-valued, unbinned certainty estimates are used in the actual classification experiments, for clarity in graphing, certainties are grouped in discrete bins (e.g., certainty values from 0.5 to 0.59 are all graphed as 0.5). The bar on the left for each bin represents the number of instances receiving this certainty value that were correctly classified. The bar on the right represents the number of instances that were incorrectly classified. For the purity certainty estimator, the far right-hand bin in the graph shows that out of all 36 data sets, 5,128 instances receiving a certainty value of 1.0 were correctly classified, and 863 instances receiving this certainty value were incorrectly classified.

We fit a trend line to the percentage of correctly classified instances in each of the various bins using least-squares regression. Each measure included in our experiments satisfies the minimum requirement that this trend line has a positive slope. Intuitively, this indicates that

TABLE 1. Information for Data Sets.

	Number of instances	Number of attributes	Output classes	Attribute type	Missing attributes?
anneal	898	39	6	Mixed	no
audiology	226	70	24	Discrete	yes
balance-scale	625	5	3	Real	no
bupa	345	7	2	Real	no
car	1,728	7	4	Discrete	no
cmc	1,473	10	3	Mixed	no
colic	368	23	2	Mixed	yes
credit-a	690	16	2	Mixed	yes
credit-g	1,000	21	2	Mixed	no
dermatology	366	35	6	Mixed	yes
diabetes	768	9	2	Real	no
ecoli-c	336	8	8	Real	no
glass	214	10	7	Real	no
haberman	306	4	2	Real	no
heart-disease	294	14	5	Mixed	yes
heart-statlog	270	14	2	Real	no
hepatitis	155	20	2	Mixed	yes
ionosphere	351	35	2	Real	no
iris	150	5	3	Real	no
lymph	148	19	4	Mixed	no
monks	432	7	2	Discrete	no
postop	90	9	3	Discrete	no
primary-tumor	339	18	22	Discrete	yes
segment	2,310	20	7	Real	no
sonar	208	61	2	Real	no
soybean	683	36	19	Discrete	yes
spect	267	23	2	Discrete	no
tic-tac-toe	958	10	2	Discrete	no
vehicle	846	19	4	Real	no
vote	461	17	2	Discrete	no
vowel	990	14	11	Mixed	no
wine	178	14	3	Real	no
wisconsin-cancer	286	10	2	Discrete	yes
yeast	1,484	9	10	Real	no
yugoslavia-cancer	699	10	2	Real	no
zoo	101	17	7	Discrete	no

the measure is more likely to assign a high certainty value to a correctly classified instance as opposed to an incorrectly classified one. Figure 5 shows such a line for the decision tree purity certainty estimator.

Each subsection contains information about the various algorithms and the certainty estimators used for each. A table outlining the correlation between the various certainty estimators and correctness in test set classification is also included.

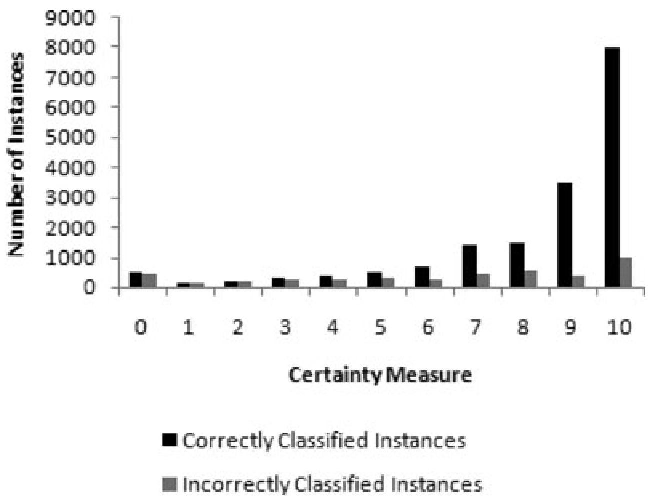


FIGURE 4. Decision tree purity certainty estimator—treatment of correct and incorrect instances.

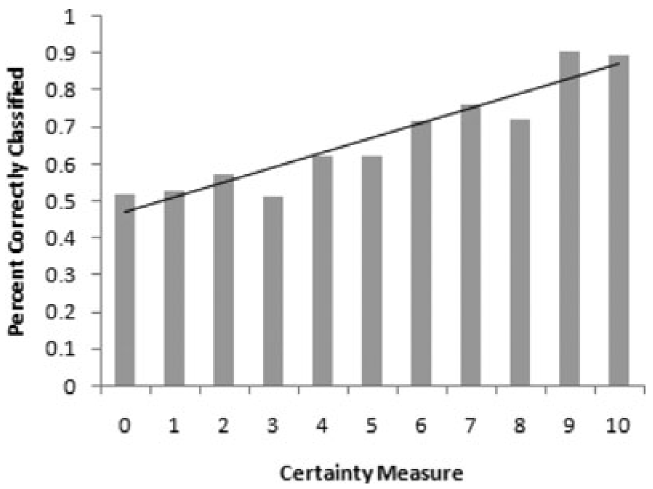


FIGURE 5. Decision tree purity certainty estimator—percentage of correctly classified instances per bin.

4.1. Decision Tree—J48

To demonstrate the effectiveness of multiple certainty measures, we provide a case study with the measures used for decision trees. The J48 algorithm is the Weka implementation of the C4.5 algorithm (Quinlan 1993), an extension of the ID3 decision tree algorithm (Quinlan 1986). Six different certainty estimators are used to predict certainty in this algorithm's classification of a given instance:[2]

(i) The number of instances with the predicted class at the leaf node when the given instance is classified (the purity of classification at that node).

[2] For a more rigorous presentation of these and all other certainty estimators mentioned in this paper, please see the Appendix.

TABLE 2. Decision Tree Certainty Estimators and Correlation with Correctly Classified Instances.

Certainty estimator	Correlation
1. Purity of classification	0.219
2. Instances at leaf node	0.167
3. Level of leaf node	0.199
4. Information gain along path	−0.072
5. Correctly classified instances	0.280
6. Correctly classified voters	0.248
Aggregate certainty estimator	**0.292**

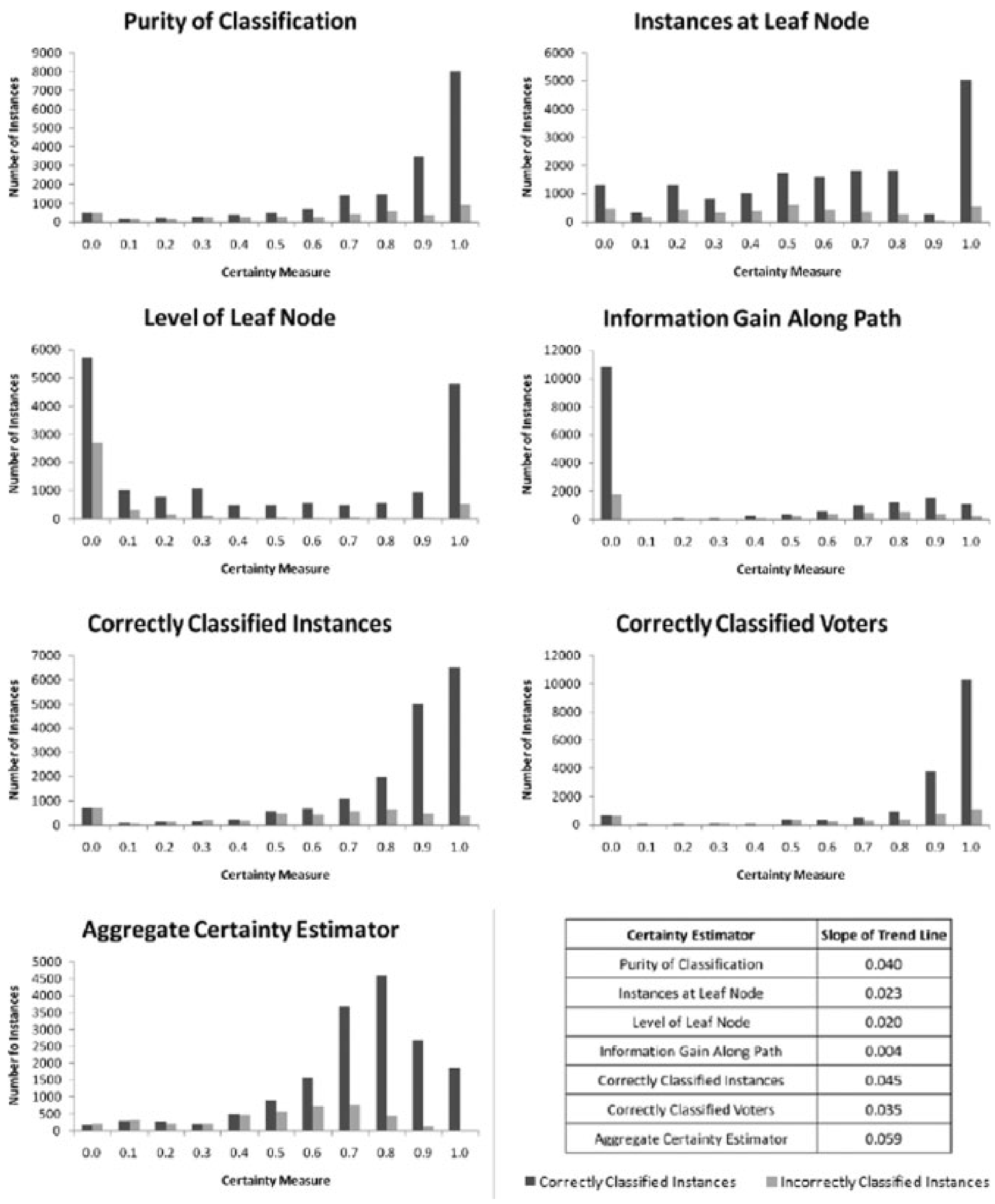
(ii) The number of instances at the leaf node.
(iii) The level of the tree at which the given instance is classified.
(iv) The average of the information gain statistics along the classification path (normalized by maximum possible information gain for a given data set).
(v) The percentage of instances at the leaf node that were correctly classified in hold-one-out cross-validation experiments on the training set.
(vi) The percentage of instances at the leaf nodes with the predicted class for that node that are correctly classified in hold-one-out cross-validation on the training set.

The first certainty estimator is a standard method for predicting certainty in the classification of a decision tree (Witten and Frank 2005). The second and third provide an effective complement to the first by providing information about the amount of overfit, and thus how much the first should be trusted. The fourth certainty estimator provides information about how effectively a given attribute is able to split the data at each level of the decision tree, assuming that strong attributes will lead to more confident classifications. The fifth identifies how effective the classifier is at classifying the instances in this particular section of the data. The sixth certainty estimator provides information about how effectively the classifier was able to classify the instances specifically contributing to the classification of the given instance.

Figure 6 provides information about the behavior of each certainty estimator on the data sets shown in Table 1. For example, the graph in the top left of the figure displays information about the certainty estimator measuring purity of classification at the leaf node of a decision tree.

Table 2 shows the correlation between correctness of classification and values provided by each certainty estimator. One can infer from this and Figure 6 that the purity of classification measure and the two certainty measures concerned with correctly classified instances appear to be better predictors of correctness of classification for test instances. However, the other certainty estimators do provide additional information that may be useful, particularly when taken into consideration with the more accurate certainty-predicting measures.

For example, with the haberman data set, a majority of the correctly classified instances were assigned a certainty rating of 0.82 by the purity certainty estimator. The few instances receiving higher certainty ratings were all misclassified. However, the misclassified instances that received deceptively high certainty ratings from the purity certainty estimator were generally found in leaf nodes that contained only a few instances. Thus, they received lower certainty ratings both from the certainty estimator that measured the percentage of



Certainty Estimator	Slope of Trend Line
Purity of Classification	0.040
Instances at Leaf Node	0.023
Level of Leaf Node	0.020
Information Gain Along Path	0.004
Correctly Classified Instances	0.045
Correctly Classified Voters	0.035
Aggregate Certainty Estimator	0.059

FIGURE 6. Decision tree certainty estimators treatment of correct and incorrect instances.

instances at the leaf node and the one that measured the level of the tree. A combination of these certainty estimators is better at predicting whether or not an instance from this data set will be correctly classified.

Table 2 shows that, on average, the certainty estimator relating to information gain was slightly negatively correlated with correctness of classification. However, this measure proved effective on selected data sets. For example, on the hepatitis data set, correlation between the information gain certainty estimator and correctness of training set classification was 0.153, while correlation with the purity of classification estimator was only 0.091. Overall accuracy of the aggregate certainty estimators strategy on the hepatitis data set as evaluated by 10-fold cross-validation was 85.16% when the information gain estimator was included, and only 84.52% when it was excluded. The inclusion of this certainty estimator resulted in an improvement in accuracy of aggregate certainty estimators on 16 of the 36 data sets studied. It reduced accuracy on only three of the data sets.

Similar patterns can be seen for the certainty estimators presented for all of the algorithms studied. For each of the estimators studied, higher certainty values generally corresponded with a higher percentage of correctly classified instances. More specifically, a trend line fit to a percentage of correct instances for each binned certainty estimator had a positive slope (e.g., Figure 5); instances assigned the highest certainty measure were more likely to be correctly classified than instances assigned the lowest certainty measure for each of the estimators studied. This is true even for certainty estimators that exhibit low average correlation with correctness of classification. However, in each case, the aggregate certainty estimator was significantly more correlated with correctness of classification than each of the individual estimators.

The Friedman test indicates that there are significant differences among the correlations of the various certainty measures. ($92.45 \sim \chi^2$, $DF = 6$, $p <= 0.0001$). The Bonferroni–Dunn post-hoc test indicates that the differences in average ranks between aggregate certainty estimator and five of the six other estimators exceed the critical difference for significance at a certainty level of 95% (adjusted $\alpha = \frac{0.05}{7-1}$, critical difference = 1.319, mean rank differences: 2.750, 2.438, 1.921, 4.781, 0.953, 2.031).

4.2. Multilayer Perceptron Trained with Backpropagation

In contrast, we also present an explanation of the certainty estimators used for the multilayer perceptron. One of the most common methods of training a multilayer perceptron, backpropagation incrementally changes the weights between nodes when these weights are responsible for the misclassification of instances during training (Rumelhart, Hinton, and Williams 1986). These experiments use a multilayer perceptron with a single hidden. The following are considered in trying to predict certainty in classification by the multilayer perceptron:

(i) The activation output for the selected classification.
(ii) The difference between the highest and second highest activation outputs.
(iii) The percentage of the five neighbors nearest in activation output that were correctly classified in hold-one-out cross-validation on the training set.
(iv) The percentage of the five neighbors nearest in activation output of the hidden layer that were correctly classified in hold-one-out cross-validation on the training set.
(v) The average difference in activation output between the selected classification and its five nearest neighbors compared to the average of this statistic computed for all instances.
(vi) The average difference in hidden-layer activation output between the selected classification and its five nearest neighbors compared to the average of this statistic computed for all instances.

The first and second certainty estimators provide information about the certainty of a given classification and certainty relative to other possible classifications. The third and fourth provide information about how confident the classifier is on this region of the input

TABLE 3. Multilayer Perceptron Certainty Estimators and Correlation with Correctly Classified Instances

Certainty estimator	Correlation
1. Activation output	0.053
2. Highest minus second	0.051
3. Correctly classified neighbors	0.295
4. Correctly classified neighbors (hidden layer)	0.266
5. Average distance to neighbors	0.239
6. Average distance to neighbors (hidden layer)	0.157
Aggregate certainty estimator	**0.310**

space. All the instances in the training set are considered, and the five with output vector most similar to the instance in question are then used to calculate the certainty estimator. The third certainty estimator uses the outputs from the standard output nodes to identify the nearest neighbors. The fourth certainty estimator uses the outputs from the hidden nodes. The fifth and sixth certainty estimators provide information about how similar a given instance is to previously seen instances, based on the assumption that the classifier will be more effective at predicting a class value for an instance similar to one that it has seen before. Figure 7 shows the behavior of these certainty measures on data set instances.

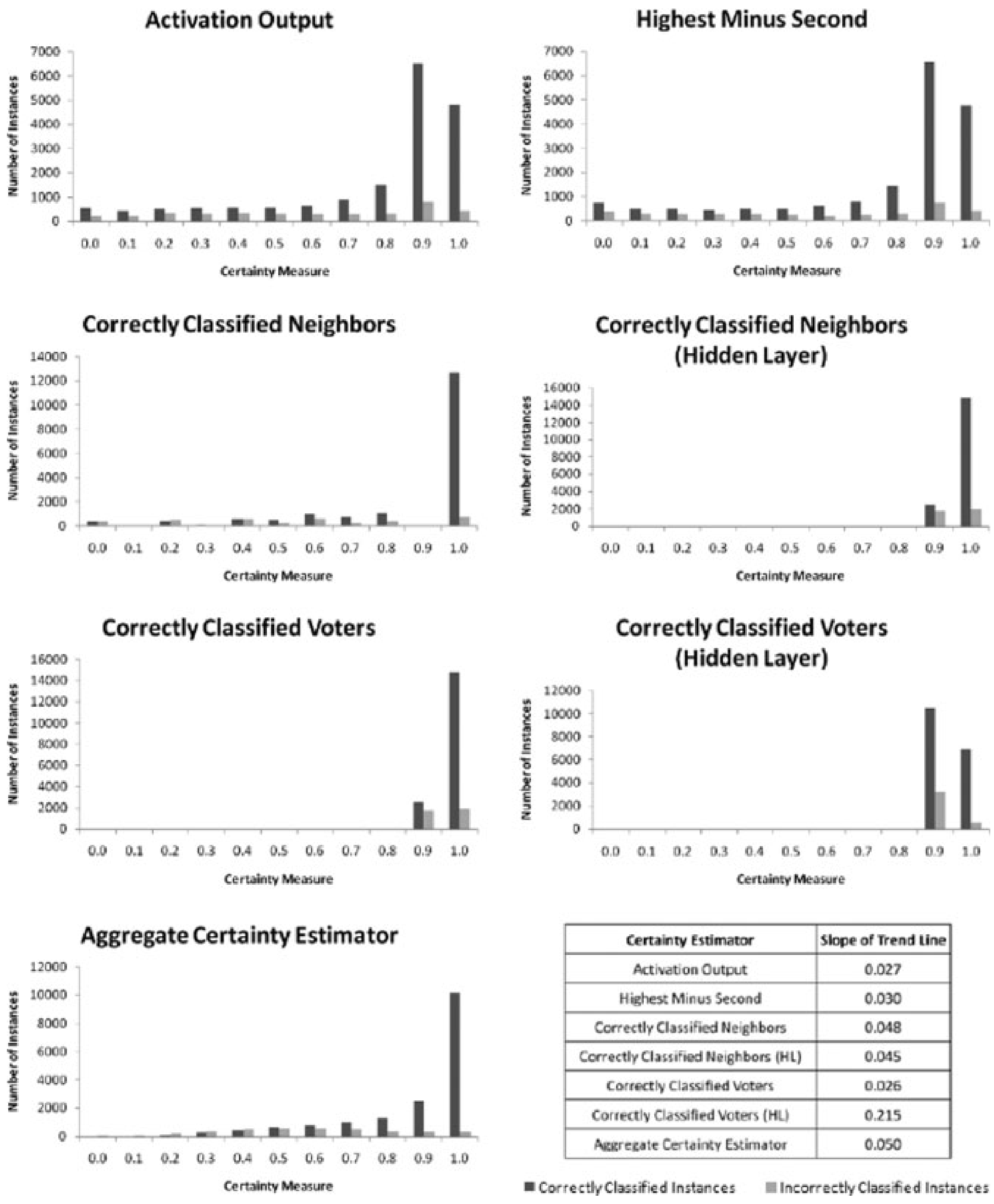
As illustrated by the graphs in Figure 7, all of these heuristics tend to assign a 1.0 certainty rating to a large number of correctly classified instances. The number of correctly classified instances at each certainty rating tends to taper off as the ratings become lower. On average, the heuristics for this classifier were more highly correlated with each other than the heuristics for other classifiers. However, an examination of the certainty ratings assigned to individual instances in the data sets shows that there is enough variation that each heuristic does provide some extra information to a classifier. The biggest jump in correlation with correctness of classification between individual certainty measures and an aggregate certainty estimator was seen with the multilayer perceptron.

Table 3 reports how values assigned by these certainty measures correlate with correctness of classification. The Friedman test indicates that there are significant differences among the correlations of the various certainty measures. ($127.99 \sim \chi^2$, $DF = 6$, $p <= 0.0001$). The Bonferroni–Dunn post-hoc test indicates that the differences in average ranks between aggregate certainty estimator and five of the six other estimators exceed the critical difference for significance at a certainty level of 95% (adjusted $\alpha = \frac{0.05}{7-1}$, critical difference = 1.216, mean rank differences: 4.329, 4.529, 0.943, 1.643, 1.871, 2.786).

4.3. Rule-Based Classifier—Decision Table

These experiments use one of Weka's rule-based classifiers called a Decision Table (Kohavi 1995). This algorithm selects a set of attributes to be used in determining classification, and produces a classification for each combination of observed values for these attributes. The following attributes are taken into consideration when trying to predict certainty in this algorithm's classification of a given instance:

(i) The number of instances with the predicted class covered by the rule.
(ii) The number of antecedents in the rule.
(iii) The number of instances covered by the rule.



Certainty Estimator	Slope of Trend Line
Activation Output	0.027
Highest Minus Second	0.030
Correctly Classified Neighbors	0.048
Correctly Classified Neighbors (HL)	0.045
Correctly Classified Voters	0.026
Correctly Classified Voters (HL)	0.215
Aggregate Certainty Estimator	0.050

FIGURE 7. Multilayer perceptron certainty estimators treatment of correct and incorrect instances.

(iv) The percentage of instances covered by the rule that were correctly classified in hold-one-out cross-validation experiments on the training set.
(v) The percentage of instances covered by the rule with the predicted class for that rule that were correctly classified in hold-one-out cross-validation on the training set.
(vi) Whether or not this instance is covered by a rule.

TABLE 4. Rule-Based Classifier Certainty Estimators and Correlation with Correctly Classified Instances.

Certainty estimator	Correlation
1. Purity of classification	0.147
2. Number of antecedents	−0.004
3. Number of instances covered	0.110
4. Correctly classified instances	0.139
5. Correctly classified voters	0.102
6. Instance is covered by rule	0.217
Aggregate certainty estimator	**0.240**

The rationale for these certainty estimators is similar to the rationale for the decision tree certainty estimators. The first is a standard measure of certainty. The second and third assess the probability of overfit or underfit. The fourth and fifth measure the effectiveness and strength of classification. They indicate how effectively the decision table was able to classify instances that would end up in this region and how effectively the most pertinent instances in this region can be classified. The sixth certainty estimator indicates whether or not a rule was found in the table that covered the given instance to be classified. Table 4 shows how values assigned by these certainty measures correlate with correctness of classification.

The Friedman test indicates that there are significant differences among the correlations of the various certainty measures. ($55.96 \sim \chi^2$, $DF = 6$, $p <= 0.0001$). The Bonferroni–Dunn post-hoc test indicates that the differences in average ranks between aggregate certainty estimator and all six other estimators exceed the critical difference for significance at a certainty level of 95% (adjusted $\alpha = \frac{0.05}{7-1}$, critical difference = 1.523, mean rank differences: 1.917, 4.542, 2.583, 2.208, 2.874, 2.208).

4.4. Instance-Based Classifier

With the instance-based k-nearest-neighbor algorithm, an instance is classified based on the classifications of the k instances nearest that instance (Cover and Hart 1967). These experiments use the five-nearest-neighbor version of the algorithm. Attribute values are normalized, and standard rather than distance weighted voting is used. Six different options are used to predict certainty in this algorithm's classification of a given instance:

(i) The percentage of the first five neighbors that have the same classification as the predicted class for those five neighbors.
(ii) The difference between the distance weighted vote of the predicted class and the distance weighted vote of the next highest class.
(iii) The average distance from this instance to its first five neighbors (normalized and subtracted from one).
(iv) The percentage of the first five neighbors that were correctly classified in hold-one-out cross-validation on the training set.
(v) The percentage of neighbors with the predicted class that were correctly classified in hold-one-out cross-validation on the training set.
(vi) A comparison of 3-NN, 5-NN, and 7-NN classifications of a given instance.

The first and second certainty estimators indicate the general certainty in a classification, and how confident that classification is relative to other possible classifications. The third

TABLE 5. Instance-Based Classifier Certainty Estimators and Correlation with Correctly Classified Instances.

Certainty estimator	Correlation
1. Neighbors in agreement	0.325
2. Highest minus second	0.343
3. Average distance to neighbors	0.114
4. Correctly classified neighbors	0.276
5. Correctly classified voters	0.198
6. 3-NN versus 5-NN versus 7-NN	0.242
Aggregate certainty estimator	**0.358**

measures how close the neighbors are to the individual instance, making the assumption that an instance closer to other instances is more likely to be correctly classified. The fourth and fifth certainty estimators measure the classification accuracy of instances in this region and the accuracy on instances contributing to the classification of the instance in question. The last certainty estimator indicates the effectiveness of using this particular number of neighbors to classify the given instance. Table 5 reports correlation between values assigned by these measures and correctness of classification.

The Friedman test indicates that there are significant differences among the correlations of the various certainty measures. ($99.96 \sim \chi^2$, $DF = 6$, $p <= 0.0001$). The Bonferroni–Dunn post-hoc test indicates that the differences in average ranks between aggregate certainty estimator and four of the six other estimators exceed the critical difference for significance at a certainty level of 95% (adjusted $\alpha = \frac{0.05}{7-1}$, critical difference = 1.261, mean rank differences: 1.257, 0.571, 4.114, 2.143, 3.143, 2.971).

4.5. Naïve Bayes Classifier

The Naïve Bayes classifier uses Bayesian logic to predict class values for each instance based on the probabilities of the attribute values for that instance (Lang 1995; Mitchell 1997). The following are considered when trying to predict certainty in classification of a given instance by the Naïve Bayes classifier:

(i) Probability of the class value predicted by the Naïve Bayes classifier.
(ii) The distance between the predicted probability and the probability of the second most likely class value for the instance.
(iii) The distance between the predicted probability and the sum of the probabilities for the remaining class values.
(iv) The average probability across the data set of each attribute value in the instance.
(v) The percentage of the five neighbors nearest in probability that were correctly classified in hold-one-out cross-validation on the training set.
(vi) The percentage of the nearest five neighbors with the same class value as this instance that were correctly classified in hold-one-out cross-validation on the training set.

The first certainty estimator is used because it is the standard way of predicting the certainty of a Naïve Bayes classifier. The second and third certainty estimators are attempts to gain more information about how confident the classifier is in its ordering. The

TABLE 6. Naïve Bayes Classifier Certainty Estimators and Correlation with Correctly Classified Instances.

Certainty estimator	Correlation
1. Probability of class value	0.303
2. Highest minus second	0.298
3. Highest minus remaining	0.303
4. Value probability averages	0.075
5. Correctly classified neighbors	0.371
6. Correctly classified voters	0.306
Aggregate certainty estimator	**0.394**

fourth certainty estimator addresses the fact that attribute values with lower representation in a data set may be less effective at contributing to a correct classification. The fifth certainty estimator is aimed at determining how confident the classifier is in this region of the input space. With this certainty estimator, the output probabilities of all the instances in the training data are taken into consideration. The five instances with output probabilities closest to those of the instance in question are then located, and the certainty estimate is calculated by observing the percentage of these five instances that were correctly classified in hold-one-out cross-validation on the training set. The sixth certainty estimator focuses specifically on neighbors with the same classification as the given instance. Table 6 shows how values assigned by these certainty measures correlate with correctness of classification.

The Friedman test indicates that there are significant differences among the correlations of the various certainty measures. ($56.59 \sim \chi^2$, $DF = 6$, $p <= 0.0001$). The Bonferroni–Dunn post-hoc test indicates that the differences in average ranks between aggregate certainty estimator and five of the six other estimators exceed the critical difference for significance at a certainty level of 95% (adjusted $\alpha = \frac{0.05}{7-1}$, critical difference = 1.244, mean rank differences: 1.542, 1.917, 1.417, 3.528, 0.777, 2.097).

5. RESULTS AND DISCUSSION

In this section, the technique of aggregate certainty estimators is compared with a number of different ensemble combining strategies. Overall accuracy for each method is calculated by using 10-fold cross-validation and averaging accuracy over each of the 10-folds. Two sets of experiments are conducted. The first set demonstrates advantages of the aggregate certainty estimators. The strategy of aggregate certainty estimators is shown to be more effective than the strategy of weighting by single certainty estimators. The second set of experiments compares aggregate certainty estimators to a number of baseline ensemble creation strategies. Specifically, aggregate certainty estimators are also shown to be competitive with other vote weighting strategies, the SelectBest method, and the methods of Arbitration and Modified Stacking.

5.1. Results

To motivate the need for multiple certainty estimators, the accuracies of different ensembles created with single certainty estimators are tested. Three different options are given for selecting single certainty estimators. The first alternate ensemble is created by using certainty estimators traditionally used in predicting certainty in a classification:

* Decision Tree: Purity of classification.
* Rule-based classifier: Purity of classification.
* Instance-based classifier: Neighbors in agreement.
* Naïve Bayes classifier: Probability of class value.
* Multilayer perceptron: Activation output.

The next ensemble is also constructed using single certainty estimators to predict certainty. However, in this case, an attempt is made to select more effective certainty estimators. For this ensemble, each algorithm uses the measure most highly correlated with whether or not an instance was correctly classified as the method for predicting certainty:

* Decision tree: Correctly classified instances.
* Rule-based classifier: Instance is covered by rule.
* Instance-based classifier: Highest minus second.
* Naïve Bayes classifier: Correctly classified neighbors.
* Multilayer perceptron: Correctly classified neighbors.

In addition, the Weka source code provides a way of calculating a probability distribution over possible classes for each instance classification. In most cases, Weka's method of calculating the probability distribution is similar to the first certainty estimator for each classifier presented in this work. Weka makes a few modifications and refinements to these measures. For a third comparison ensemble, the probability distributions predicted by Weka are used to weight the votes of each of the five classifiers.

All these techniques are then compared to the aggregate certainty estimators' strategy of using a larger set of measures to predict certainty. The resulting predictive accuracies, as shown in Table 7, demonstrate the utility of using more certainty estimators.

An application of the Friedman test reports significant differences in accuracy among the classifiers. ($14.19 \sim \chi^2$, $DF = 3$, $p <= 0.003$). The Bonferroni–Dunn post-hoc test reveals that the differences in average ranks between aggregate certainty estimators and two of the three other methods exceed the critical difference for significance at a certainty level of 95% (adjusted $\alpha = \frac{0.05}{4-1}$, critical difference = 0.89, mean rank differences: 1.01, 0.85, 0.92). An algorithm-by-algorithm comparison between the various strategies using the Wilcoxon signed-rank test shows that aggregate certainty estimators outperform the other three algorithms at a certainty level of 99% (p values<=0.0001, 0.001, 0.001).

In the next set of experiments, aggregate certainty estimators are compared with several different baseline methods. Once again, overall accuracy for each method is calculated by using 10-fold cross-validation and averaging accuracy over each of the 10-folds. The first is a standard voting method where each classifier in an ensemble votes on the classification of an instance and the votes are weighted equally. The second baseline method weights the votes by the overall accuracy of the classifier on the training data for a given fold of the experiments. The third baseline method, identified here as the SelectBest method, chooses the classifier in the ensemble that achieved the highest accuracy on the training data and uses that classifier alone on the test data.

Aggregate certainty estimators is also compared to the method of Stacking found to be most effective by Dzeroski and Zenko (2004). In this method, identified as Modified Stacking in the following analyses, the output probabilities of each of the component classifiers are

TABLE 7. Comparison of Predictive Accuracies of Aggregate Certainty Estimators with Single Certainty Measure Ensembles.

Data set	Traditional certainty estimators	Highest correlation estimators	Weka outputs	Aggregate confidence ensembles
anneal	99.44	99.33	99.22	99.33
audiology	79.20	80.53	80.09	78.32
balance-scale	88.00	89.76	90.08	89.92
bupa	70.44	68.70	69.28	71.01
car	97.28	96.53	96.41	97.74
cmc	54.04	52.21	53.43	53.50
colic	83.97	84.51	83.97	84.24
credit-a	85.07	85.22	85.65	86.23
credit-g	75.60	74.60	75.20	75.40
dermatology	97.81	97.81	97.81	97.27
diabetes	76.43	76.43	76.56	76.56
dcoli-c	85.71	87.20	87.20	87.50
dlass	71.03	71.03	71.50	71.50
haberman	74.51	71.90	74.18	73.86
heart-h	83.33	80.95	82.99	82.99
heart-statlog	84.82	83.70	83.33	84.82
hepatitis	82.58	83.87	81.29	85.16
ionosphere	92.59	92.31	92.02	93.16
iris	94.67	95.33	96.00	96.00
lymph	84.46	83.11	83.11	85.14
monks	99.77	99.54	99.77	99.54
postop	67.78	68.89	70.00	71.11
primary-tumor	45.72	46.31	46.90	46.02
segment	97.32	97.49	97.40	97.49
sonar	82.69	84.62	82.69	83.65
soybean	95.17	95.02	94.58	94.14
spect	84.27	83.52	83.90	85.39
tic-tac-toe	92.80	94.26	93.42	94.68
vehicle	71.63	74.82	74.47	74.94
vote	95.66	95.88	95.88	96.31
vowel	91.41	94.55	94.04	95.05
wine	97.75	98.32	97.19	97.75
wisconsin-cancer	73.43	74.48	73.08	75.52
yeast	59.77	60.31	59.97	60.78
yugoslavia-cancer	96.28	96.42	96.57	96.42
zoo	96.04	96.04	96.04	97.03
Average:	83.57	83.76	83.76	84.32

given as input to a set of model trees. Each tree is designed to make a binary decision about a given possible output class, and the ensemble assigns a value to the instance according to which model tree has the highest positive certainty in its prediction. Table 8 shows the results of these comparisons.

TABLE 8. Comparison of Predictive Accuracies of Aggregate Certainty Estimators with Additional Baseline Strategies.

Data set	Standard voting	Accuracy weighted	Select best	Arbitration	Modified stacking	Aggregate certainty estimators
anneal	99.22	99.33	98.89	99.22	99.11	99.33
audiology	78.76	79.20	78.76	80.09	75.66	78.32
balance-scale	89.28	89.60	89.92	89.44	95.52	89.92
bupa	68.99	69.28	67.25	66.38	57.10	71.01
car	96.30	96.30	98.73	96.18	99.13	97.74
cmc	53.70	52.89	52.61	56.14	50.44	53.50
colic	83.97	83.97	83.70	82.34	82.07	84.24
credit-a	85.65	85.65	84.78	85.51	84.64	86.23
credit-g	75.30	75.30	75.30	75.00	73.30	75.40
dermatology	97.81	97.81	96.18	97.00	96.72	97.27
diabetes	76.56	76.56	74.35	77.34	71.22	76.56
ecoli-c	86.31	87.20	86.61	86.31	84.82	87.50
glass	71.50	73.36	70.09	67.76	71.50	71.50
haberman	74.18	74.18	74.84	71.57	71.90	73.86
heart-h	82.99	82.99	84.35	81.29	80.27	82.99
heart-statlog	83.70	83.70	81.48	84.44	79.26	84.82
hepatitis	81.29	81.29	83.23	83.87	81.94	85.16
ionosphere	92.02	92.02	88.32	92.02	93.16	93.16
iris	95.33	96.00	90.00	96.00	95.33	96.00
lymph	81.76	81.76	77.70	82.43	83.11	85.14
monks	99.77	99.77	100.00	100.00	100.00	99.54
postop	70.00	70.00	70.00	70.00	71.11	71.11
primary-tumor	48.08	47.49	51.03	47.49	37.46	46.02
segment	97.32	97.49	96.32	96.84	97.40	97.49
sonar	82.69	82.69	83.65	81.73	86.06	83.65
soybean	94.44	94.14	92.83	94.88	93.85	94.14
spect	83.90	83.90	83.52	83.15	79.40	85.39
tic-tac-toe	93.32	93.32	98.96	96.56	99.79	94.68
vehicle	75.65	75.65	79.20	74.47	80.73	74.94
vote	95.88	95.88	96.10	96.53	97.18	96.31
vowel	93.64	94.65	96.06	94.95	96.67	95.05
wine	97.19	97.19	97.75	97.19	96.63	97.75
wisconsin-cancer	73.43	73.43	75.18	73.43	74.48	75.52
yeast	59.70	59.70	58.42	60.45	57.62	60.78
yugoslavia-cancer	96.57	96.57	97.00	95.99	97.43	96.42
zoo	95.05	95.05	92.08	95.05	93.07	97.03
Average:	83.65	83.76	83.48	83.58	82.92	84.32

Again, the Friedman test reports significant differences in accuracy among the classifiers. ($15.51 \sim \chi^2$, $DF = 5$, $p <= 0.008$). An application of the Bonferroni–Dunn post-hoc test reveals that the differences in average ranks between aggregate certainty estimators and four of the remaining five methods exceed the critical difference for significance at a certainty level of 95% (adjusted $\alpha = \frac{0.05}{6-1}$, critical difference = 1.14, mean rank differences: 1.21, 0.88, 1.24, 1.31, 1.43). An algorithm-by-algorithm comparison between the various strategies using the Wilcoxon signed-rank test shows that aggregate certainty estimators outperform the other five algorithms at a certainty level of 99% (p values$<=$0.001, 0.003, 0.014, 0.003, 0.008).

5.2. Discussion

Aggregate certainty estimators are able to achieve higher average classification accuracy than any of three standard baseline strategies over the 36 data sets studied. A comparison between Tables 7 and 8 shows that using single certainty estimators in weighting the votes of an ensemble can allow the ensemble to make improvements in average predictive accuracy. Two of the three single certainty estimator ensembles can achieve higher average classification accuracy than a baseline strategy of standard voting. However, the use of these single certainty estimator values is not sufficient to create an ensemble that can produce a higher average predictive accuracy on a level that is statistically significant, so investigation into additional certainty estimators is warranted.

The higher average accuracy of aggregate certainty estimators does come with a higher cost of computation, but for two-thirds of the certainty estimators, the increase in computational complexity is only linear in regards to the size of the data set. The other one-third of the certainty estimators requires a cross-validation strategy in the training set. The computational complexity for these certainty estimators could be reduced substantially by reducing the number of folds used in the calculations.

6. CONCLUSION AND FUTURE WORK

This work presents a viable new method of combining the outputs of base inducers in a multiclassifier system using multiple certainty estimators to predict certainty in the classification of a given instance. A number of certainty estimators designed for this task are proposed for each of five different types of classifiers. Aggregate measures are shown to be more highly correlated with whether an instance is correctly classified than any of the individual measures. The strategy of aggregate certainty estimators, which employs all of the certainty estimators presented, is shown to achieve a higher average classification accuracy over 36 data sets than five alternate ensemble strategies.

The certainty estimators presented in this work explore some of the strengths and weaknesses of a given classifier on a given data set. This information could result in the development of new algorithms. For example, a new instance-based classifier might be developed in which only instances that were correctly classified in hold-one-out cross-validation would be allowed to vote on the classification of an unseen instance. The probabilities output by a Naïve Bayes classifier might be altered slightly based on information gained through certainty estimators like the ones presented here. Insights gained by observing the behavior of the certainty estimators on various data sets may help target areas of improvement to increase classification accuracy of individual classifiers.

REFERENCES

ALI, K. M., and M. J. PAZZANI. 1996. Error reduction through learning multiple descriptions. Machine Learning, **24**(3):173–202.

BREIMAN, L. 1996. Bagging predictors. Machine Learning, **24**(2):123–140.

BREIMAN, L. 2001. Random forests. Machine Learning, **45**(1):5–32.

CARNEY, J., P. CUNNINGHAM, and U. BHAGWAN. 1999. Confidence and prediction intervals for neural network ensembles. *In* Proceedings of the International Joint Conference on Neural Networks, 1999 (IJCNN'99), Vol. **2**, IEEE, pp. 1215–1218.

COVER, T. M., and P. HART. 1967. Nearest neighbor pattern classification. IEEE Transactions on Information Theory, **13**:21–27.

DOLEV, S., G. LESHEM, and R. YAGEL. 2010. Purifying data by machine learning with certainty levels. *In* Proceedings of the Third International Workshop on Reliability, Availability, and Security, ACM, article **5**.

DOMINGOS, P., and M. J. PAZZANI. 1997. On the optimality of the simple bayesian classifier under zero-one loss. Machine Learning, **29**:103–130.

DZEROSKI, S., B. CESTNIK, and I. PETROVSKI. 1993. Using the m-estimate in rule induction. Journal of Computing and Information Technology, **1**:37–46.

DZEROSKI, S., and B. ZENKO. 2004. Is combining classifiers with stacking better than selecting the best one? Machine Learning, **54**:255–273.

FERRI, C., P. FLACH, and J. HERNANDEZ-ORALLO. 2003. Improving the auc of probabilistic estimation trees. *In* Proceedings of the Fourteenth European Conference of Machine Learning, pp. 121–132.

FERRI, C., P. FLACH, and J. HERNANDEZ-ORALLO. 2004. Delegating classifiers. *In* Proceedings of the 21st International Conference on Machine Learning, pp. 289–296.

FREUND, Y., and R. E. SCHAPIRE. 1996. Experiments with a new boosting algorithm. *In* Proceedings of the 13th International Conference on Machine Learning, pp. 148–156.

HE, F., and D. XIAOQING. 2007. Improving naive bayes text classifier using smoothing methods. *In* Advances in Information Retrieval, pp. 703–707.

HETTICH, S., C. L. BLAKE, and C. J. MERZ. 1998. Uci repository of machine learning databases.

HO, T., J. HULL, and S. SRIHARI. 1994. Decision combination in multiple classifier systems. IEEE Transactions on Pattern Analysis and Machine Intelligence, **16**(1):66–75.

HOETING, J., D. MADIGAN, A. RAFTERY, and C. VOLINSKY. 1999. Bayesian model averaging: A tutorial. Statistical Science, **14**(4):382–417.

JACOBS, R. A., M. I. JORDAN, S. J. NOWLAN, and G. E. HINTON. 1991. Adaptive mixtures of local experts. Neural Computation, **3**:79–87.

KOHAVI, R. 1995. The power of decision tables. *In* Proceedings of the 8th European Conference of Machine Learning, pp. 174–189.

LANG, K. 1995. Newsweeder: Learning to filter netnews. *In* Proceedings of the 12th International Conference on Machine Learning, pp. 331–339.

MITCHELL, T. M. 1997. Machine Learning. WCB/McGraw-Hill: New York.

ORTEGA, J., M. KOPPEL, and S. ARGAMON. 2001. Arbitrating among competing classifiers using learned referees. Knowledge and Information Systems Journal, **3**(4):470–490.

PROVOST, F., and P. DOMINGOS. 2003. Tree induction for probability-based ranking. Machine Learning, **52**:199–216.

QUINLAN, J. R. 1986. Induction of decision trees. Machine Learning, **1**(1):81–106.

QUINLAN, J. R. 1993. C4.5: Programs for Machine Learning. Morgan Kaufman: San Mateo, CA.

RICHARD, M., and R. LIPPMAN. 1991. Neural network classifiers estimate bayesian a-posteriori probabilities. Neural Computation, **3**:461–483.

ROKACH, L. 2010. Ensemble-based classifiers. Artificial Intelligence Review, **33**(1):1–39.

RUCK, D., S. ROGERS, M. KABRISKY, M. OXLEY, and B. SUTER. 1990. The multilayer perceptron as an approximation to a bayes optimal discriminant function. IEEE Transactions on Neural Networks, **1**(4):296–298.

RUMELHART, D. E., G. E. HINTON, and R. J. WILLIAMS. 1986. Learning internal representations by error propagation. Parallel Distributed Processing: Explorations in the Microstructure of Cognition, **1**: 318–362.

SCHAPIRE, R. 1990. The strength of weak learnability. Machine Learning, **5**(2):197–227.

SCHAPIRE, R. E., and Y. SINGER. 1998. Improved boosting algorithms using confidence-rated predictions. *In* Proceedings of the 11th Annual Conference on Computational Learning Theory, pp. 80–91.

WITTEN, I. H., and E. FRANK. 2005. Data Mining: Practical Machine Learning Tools and Techniques (2nd ed.). Morgan Kaufmann: San Francisco.

WOLPERT, D. 1992. Stacked generalization. Neural Networks, **5**(2):241–260.

APPENDIX

Tables A1–A5 offer more rigorous presentations of how the estimators are calculated. When calculating certainty measures for training set instances to determine correlation values, "training set" in these formulas refers to the instances used for training in hold-one-out cross-validation and "test instance" refers to the instance being held out (e.g., if 135 instances of the iris data set were being used to evaluate the other 15 instances in 10-fold cross-validation, measures for the 135 training set instances would be calculated using 134 of them as a "training set" subset and the remaining instances as a "test instance"). When calculating certainty measures for the test instances, "training set" in these formulas refers to the standard cross-validation training set (e.g., the entire set of 135 iris instances) and "test instance" refers to the instances in the fold being evaluated (e.g., the remaining 15 iris instances).

TABLE A1. Certainty Estimators for Decision Tree.

1. Purity of Classification: $\frac{p}{m}$

p : number of training set instances with predicted class
at leaf node where test instance is classified
m : number of training set instances at leaf node where test instance is classified

2. Instances at Leaf Node: $\frac{m}{n}$

m : number of training set instances at leaf node where test instance is classified
n : number of instances in the training set

3. Level of Leaf Node : $\frac{(k-v)}{k}$

k : maximum number of levels in the tree
v : level of leaf node where test instance is classified

4. Information Gain along Path: $\sum_1^v g_i * \frac{1}{v} * \frac{1}{h}$

g_i : information gained by splitting on the selected attribute at level i in the tree
v : level of leaf node where test instance is classified
h : maximum possible information gain for given training set

Information gain for an attribute A given data set S is defined as follows:
$InformationGain(A) = Entropy(S) - \Sigma_{i=1}^{|A|} \frac{|S_i|}{|S|} Entropy(S_i)$

TABLE A1. Continued.

5. Correctly Classified Instances: $\frac{q}{m}$

q : number of training set instances at leaf node where test instance is classified
that were correctly classified in cross-validation on the training set

m : number of training set instances at leaf node where instance is classified

6. Correctly Classified Voters: $\frac{r}{p}$

r : number of training set instances with predicted class
at leaf node where test instance is classified
that were correctly classified in cross-validation on the training set

p : number of training set instances with predicted class
at leaf node where test instance is classified

TABLE A2. Certainty Estimators for Rule-Based Classifier.

1. Purity of Classification: $\frac{p}{m}$

p : number of training set instances with predicted class
covered by the rule applying to the test instance

m : number of training set instances covered by the rule applying to the test instance

2. Number of Antecedents: $\frac{(k-v)}{k}$

k : maximum number of antecedents in any rule of the table

v : number of antecedents in the rule applying to the test instance

3. Number of Instances Covered: $\frac{m}{n}$

m : number of training set instances covered by the rule applying to the test instance

n : number of instances in the training set

4. Correctly Classified Instances: $\frac{q}{n}$

q : number of training set instances covered by the rule applying to the test instance
that were correctly classified in cross-validation on the training set

n : number of training set instances covered by the rule applying to the test instance

5. Correctly Classified Voters: $\frac{r}{p}$

r : number of training set instances with predicted class
covered by the rule applying to the test instance
that were correctly classified in cross-validation on the training set

p : number of training set instances with predicted class
covered by the rule applying to the test instance

6. Instance is Covered by Rule: $\begin{cases} 1.0 : \text{A rule in the table applies to the test instance} \\ 0.0 : \text{No rules in the table apply to the test instance} \end{cases}$

TABLE A3. Certainty Estimators for Instance-Based Classifier.

---

1. Neighbors in Agreement: $\frac{p}{m}$

p : number of neighbors with predicted class
m : number of neighbors considered (5)

2. Highest Minus Second: $w_1 - w_2$

w_1 : distance-weighted vote which determines the predicted class
w_2 : distance-weighted vote of the next highest class

3. Average Distance to Neighbors: $1 - \frac{c}{d}$

c : average distance to five neighbors
d : average distance to all instances in training set

4. Correctly Classified Neighbors: $\frac{q}{m}$

q : number of neighbors that were correctly classified in cross-validation on the training set
m : number of neighbors considered (5)

5. Correctly Classified Voters: $\frac{r}{p}$

r : number of neighbors with predicted class that were correctly classified in cross-validation on the training set
p : number of neighbors with predicted class

6. 3-NN vs. 5-NN vs. 7-NN: $\begin{cases} 1.0 : \text{3-NN and 7-NN match 5-NN classification} \\ 0.5 : \text{one matches} \\ 0.0 : \text{neither matches} \end{cases}$

TABLE A4. Certainty Estimators for Naïve Bayes Classifier.

---

1. Probability of Class Value: b_1

b_i : ith ordered value of label assigned to test instance (ranked by probability)

2. Highest Minus Second: $b_1 - b_2$

b_i : ith ordered value of label assigned to test instance (ranked by probability)

3. Highest Minus Remaining $b_1 - \sum_3^n b_i$

b_i : ith ordered value of label assigned to test instance (ranked by probability)

4. Value Probability Averages $(\Sigma_{i=1}^{m}(v_i/n))/m$

v_i : number of training set instances that have the ith attribute value in common with the test set instance
n : number of instances in the training set
m : number of attributes in any instance

5. Correctly Classified Neighbors: $\frac{q}{m}$

q : number of neighbors that were correctly classified in cross-validation on the training set
m : number of neighbors considered (5)
Note: Neighbors are calculated based on similarity of predicted probability

6. Correctly Classified Voters: $\frac{r}{p}$

r : number of neighbors with predicted class that were correctly classified
in cross-validation on the training set
p : number of neighbors with predicted class

Note: Neighbors are calculated based on similarity of predicted probability.

TABLE A5. Certainty Estimators for Multilayer Perceptron.

1. Activation Output: a_1

a_1 : highest activation output

2. Highest Minus Second: $a_1 - a_2$

a_1 : highest activation output
a_2 : second highest activation output

3. Correctly Classified Neighbors: $\frac{p}{m}$

p : number of neighbors that were correctly classified in cross-validation on the training set
m : number of neighbors considered (5)

Note: Neighbors are calculated based on similarity of activation outputs

4. Correctly Classified Neighbors (Hidden Layer): $\frac{p_h}{m}$

p_h : number of neighbors that were correctly classified in cross-validation on the training set
m : number of neighbors considered (5)

Note: Neighbors are calculated based on similarity of hidden layer activation outputs

5. Average Distance to Neighbors $1 - \frac{c}{d}$

c : average distance to five neighbors
d : average distance to all instances in training set

Note: Neighbors are based on similarity of output layer activation

6. Average Distance to Neighbors (Hidden Layer) $1 - \frac{c_h}{d_h}$

c_h : average distance to five neighbors
d_h : average distance to all instances in training set

Note: Neighbors are based on similarity of hidden layer activation outputs.