# Missing Values in Fuzzy Rule Induction

Thomas R. Gabriel and Michael R. Berthold
ALTANA Chair for Bioinformatics and Information Mining
Department of Computer and Information Science
Konstanz University, Box M 712, 78464 Konstanz, Germany
{gabriel,berthold}@inf.uni-konstanz.de

***Abstract*— In this paper, we show how an existing fuzzy rule induction algorithm can incorporate missing values in the training procedure in a very natural way. The underlying algorithm generates rules which restrict the feature space only along a few, important attributes. This property can be used to limit the algorithm's three major steps to the reduced feature space for each training instance, which allows the features for which no values are known to be ignored. Hence no replacement is necessary and the algorithm simply uses all available knowledge from each training instance. We demonstrate on data sets from the UCI repository that this method works well, generates rule sets that have comparable classification accuracy, and are, at times, even smaller than the rule sets generated by the original algorithm.**

**Keywords:** Fuzzy Rule Induction, Missing Values.

## I. Introduction

Missing values present a big obstacle for many learning algorithms. Especially for rule induction methods, it is often required to use training instances for which all feature values are known. In practice this is often achieved using more or less sophisticated imputation methods, ranging from simple injection of the mean to online prediction of the missing values throughout training.

Dealing with incomplete information is not a new challenge in data mining. In [7] a Bayesian technique for extracting class probabilities given partial data in Gaussian basis function networks is discussed. However to find the optimal solution, integration via all missing dimensions weighted by local probability densities is required. A substitution methodology to tolerate missing values in a fuzzy environment is presented in [1]. Here, missing values are replaced by the so-called *best guess*, that is, the model predicts the most plausible value for a missing attribute value. More often, simply the mean or a constant value are substituted for missing values.

In this paper we concentrate on handling missing values by incorporating them directly into the learning process. This is possible as the underlying fuzzy rule induction algorithm only concentrates on a few, important attributes and hence does not necessarily require complete feature vectors at all times. We can use this property to have the algorithm focus entirely on the known attributes for each individual training example and therefore make use of all available knowledge without introducing artifacts through artificial replacements.

The paper is organized as follows: we first give a brief introduction of the underlying fuzzy rule learning algorithm before explaining the extension that allows to incorporate missing value in detail. We conclude with an evaluation section and discussion.

## II. Fuzzy Rule Induction

The underlying fuzzy rule learning algorithm [3] constructs a set of fuzzy rules from given training data. We briefly summarize the used type of fuzzy rules before explaining the main structure of the training algorithm.

The underlying fuzzy system generates individual fuzzy rules defined by independent membership functions for each dimension in the feature space:

$$\begin{array}{llllll}
\mathcal{R}_1^1 : \texttt{IF}\ x_1\ \texttt{IS}\ \mu_{1,1}^1 & \wedge\cdots\wedge & x_n\ \texttt{IS}\ \mu_{n,1}^1 & \texttt{THEN class}\ 1 \\
\quad\vdots & \quad\vdots & & \quad\vdots \\
\mathcal{R}_{r_1}^1 : \texttt{IF}\ x_1\ \texttt{IS}\ \mu_{1,r_1}^1 & \wedge\cdots\wedge & x_n\ \texttt{IS}\ \mu_{n,r_1}^1 & \texttt{THEN class}\ 1 \\
\quad\vdots & \quad\vdots & & \quad\vdots \\
\mathcal{R}_j^k : \texttt{IF}\ x_1\ \texttt{IS}\ \mu_{1,j}^k & \wedge\cdots\wedge & x_n\ \texttt{IS}\ \mu_{n,j}^k & \texttt{THEN class}\ k \\
\quad\vdots & \quad\vdots & & \quad\vdots \\
\mathcal{R}_{r_c}^c : \texttt{IF}\ x_1\ \texttt{IS}\ \mu_{1,r_c}^c & \wedge\cdots\wedge & x_n\ \texttt{IS}\ \mu_{n,r_c}^c & \texttt{THEN class}\ c
\end{array}$$

where $\mathcal{R}_j^k$ represents rule $j$ for class $k$. The rule base contains rules for $c$ classes and $r_k$ indicates the number of rules for class $k$ ($1 \leq j \leq r_k$ and $1 \leq k \leq c$).

The fuzzy sets $\mu_{i,j}^k : \mathbb{R} \mapsto [0,1]$ are defined for every feature $i$ ($1 \leq i \leq n$) and the overall degree of fulfillment of a specific rule for an input pattern $\vec{x} = (x_1, \ldots, x_n)$ can be computed using the minimum-operator as fuzzy-AND:

$$\mu_j^k(\vec{x}) = \min_{i=1,\cdots,n} \left\{ \mu_{i,j}^k(x_i) \right\}.$$

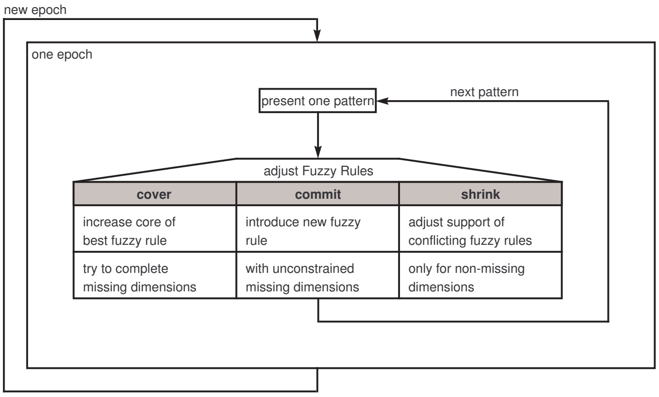
The combined degree of membership for all rules of class $k$ can be calculated using the maximum-operator as fuzzy-OR:

$$\mu^k(\vec{x}) = \max_{j=1,\cdots,r_k} \left\{ \mu_j^k(\vec{x}) \right\}.$$

From these membership values the predicted class $k_{\text{best}}$ for an input pattern $\vec{x}$ is derived then as:

$$k_{\text{best}}(\vec{x}) = \arg\max_{k=1,\ldots,c} \left\{ \mu^k(\vec{x}) \right\}.$$

The algorithm uses trapezoid membership functions which can be described with four parameters $<a_i, b_i, c_i, d_i>$, where $a_i$ and $d_i$ define the fuzzy rule's support-, and $b_i$ and $c_i$ its core-region for each attribute $i$ of the input dimension. The training algorithm usually only constrains few attributes, that is, most support-regions remain infinite, leaving the rules interpretable even in the case of high-dimensional input spaces.



| cover | commit | shrink |
|---|---|---|
| increase core of best fuzzy rule | introduce new fuzzy rule | adjust support of conflicting fuzzy rules |
| try to complete missing dimensions | with unconstrained missing dimensions | only for non-missing dimensions |

Fig. 1. The algorithm to construct a fuzzy rule set based on example data [2] incorporating missing values.

The fuzzy rule induction method is based on an iterative algorithm. During each learning epoch, i. e. presentation of all training patterns, new fuzzy rules are introduced when necessary and existing ones are adjusted whenever a conflict occurs. Figure 1 shows the main flow of the algorithm. For each pattern three main steps are executed:

- **Cover**: If a new training pattern lies inside the support-region of an already existing fuzzy rule of the correct class, its core-region is extended to cover the new pattern. In addition, the weight of this rule is incremented.
- **Commit**: If the new pattern is not yet covered, a new fuzzy rule belonging to the corresponding class is created. The new example is assigned to its core-region, whereas the overall rule's support-region is initialized "infinite", that is, the new fuzzy rule is unconstrained and covers the entire domain.
- **Shrink**: If a new pattern is incorrectly covered by an existing fuzzy rule of conflicting class, this fuzzy rule's core- and/or support-region is reduced, so that the conflict with the new pattern is avoided. The underlying heuristic of this step aims to minimize the loss in volume.

The algorithm usually terminates after only few iterations over the training data. The final set of fuzzy rules can be used to compute a degree of class activation for new input patterns.

As mentioned above, the training procedure relies on a heuristic which affects the strategy to avoid conflicts. One common approach is to shrink the dimension with minimum loss in volume for an existing fuzzy rule:

$$i_{\min} = \arg\min_{i=1,\cdots,n}\left\{V_i\right\}.$$

The loss in volume $V_i$ of a fuzzy rule $\mathcal{R}$ using trapezoid membership functions with parameters $<a, b, c, d>$ where $(a, b)$ and $(c, d)$ bound the support-region, and $[b, c]$ the fuzzy rule's core-region is then:

$$V_i = d_i^*(\vec{x}, \mathcal{R}) \;\cdot \prod_{j=1, j\neq i}^{n} d_j^\times(\vec{x}, \mathcal{R}),$$

where $d_i^*$ $(1 \leq i \leq n)$ is the distance between example pattern $\vec{x}$ and the border (of the core- or support-region) of a fuzzy rule $\mathcal{R}$ in dimension $i$, and $d_j^\times$ $(1 \leq j \leq n)$ indicates the distance of fuzzy rule $\mathcal{R}$ in dimension $j$. Furthermore, the loss in volume is normalized with respect to the overall volume:

$$V_i^{norm} = \frac{d_i^*(\vec{x}, \mathcal{R}) \;\cdot\; \prod\limits_{j=1, j\neq i}^{n} d_j^\times(\vec{x}, \mathcal{R})}{\prod\limits_{j=1}^{n} d_j^\times(\vec{x}, \mathcal{R})} = \frac{d_i^*(\vec{x}, \mathcal{R})}{d_i^\times(\vec{x}, \mathcal{R})}\;.$$

That is, the computation of this loss in volume can be simplified because only the shrunken dimension needs to be considered. In [4] different shrink heuristics and fuzzy norms are evaluated. The most popular choice for these fuzzy norms are introduced by Lotfi A. Zadeh in [8]:

$$\begin{aligned} \top\left(\mu(x), \mu(y)\right) &= \min\left\{\mu(x), \mu(y)\right\}, \\ \bot\left(\mu(x), \mu(y)\right) &= \max\left\{\mu(x), \mu(y)\right\}, \end{aligned}$$

where $\mu$ is the degree of membership of a fuzzy rule, $\top$ (t-norm) the fuzzy operator for the conjunction, and $\bot$ (t-conorm) the operator for the disjunction. This so-called minimum/maximum norm represents the most optimistic resp. most pessimistic choice for these operators.

## III. Incorporate Missing Values

The used fuzzy rule induction algorithm can easily be adapted to handle missing values during the learning process. New fuzzy rules are initialized by an anchor value retrieved from the training example, a core-region with zero spread at the anchor value and the support-region "infinity". In case of missing values, these dimensions are handled as unconstrained, that means, neither core- nor support-region are defined. The membership degree for this dimension is always $1.0$. If a new pattern needs to be covered, the prototype's core-region is extended to cover the new pattern. In case of a missing dimension, the anchor is initialized the first time a valid value appears in the training data. Missing dimensions of a prototype are not shrunken as long as no real attribute value is available from the input data.

The three main steps (which are, cover, commit, and shrink) need to be extended to incorporate missing values as follows:

- **Cover**: If a new training pattern contains missing attribute values, these dimensions are ignored. If the prototype's anchor still contains missing dimensions, the algorithm initializes the anchor with the attribute values given by the training example if existent.
- **Commit**: If a new fuzzy rule needs to be committed, the "missing" dimensions are left missing in the anchor. As before, all dimensions are initialized unconstrained, that is, their support area covers the entire feature space.
- **Shrink**: If a pattern containing missing values needs to be shrunken, both the missing dimension of the training example and the fuzzy rule are ignored. This is due to the fact that no initial values (for the anchor) are yet available to compute the loss in volume for this attribute.

If unknown instances need to be classified, the missing dimensions are treated as unconstrained with a membership

TABLE I

RESULTS ON THE SATIMAGE DATASET.

| Miss. | Incorp. Miss. | | *Best Guess* | | Use Mean | | Use Zero | |
|---|---|---|---|---|---|---|---|---|
| [%] | Error | $\lvert\mathbf{R}\rvert$ | Error | $\lvert\mathbf{R}\rvert$ | Error | $\lvert\mathbf{R}\rvert$ | Error | $\lvert\mathbf{R}\rvert$ |
| 0 | 14.6 | 478 | 14.6 | 478 | 14.6 | 478 | 14.6 | 478 |
| 1 | 14.0 | 470 | 15.4 | 463 | 13.0 | 484 | 13.6 | 530 |
| 5 | 15.5 | 500 | 14.1 | 463 | 14.7 | 514 | 14.3 | 1577 |
| 10 | 15.2 | 461 | 14.4 | 349 | 15.2 | 530 | 15.4 | 2507 |
| 20 | 15.4 | 458 | 15.1 | 294 | 15.9 | 611 | 38.0 | 3366 |
| 40 | 16.4 | 436 | 16.5 | 165 | 15.7 | 679 | 44.8 | 3605 |
| 60 | 16.5 | 463 | 21.2 | 119 | 16.6 | 789 | 48.8 | 3662 |

degree of $1.0$, which is then used to compute the overall fuzzy membership degree for this rule using the minimum-operator. To find the best rule among all, the maximum-operator is applied.

## IV. EVALUATION AND RESULTS

To demonstrate the usefulness of the proposed method, several well-known datasets from the StatLog–project [6] were used. Training and testing were performed according to the instructions available with the data sets, that is, for Shuttle, SatImage, and Letter data a pre-defined split into training and testing data was used. For the remaining data sets $k$-fold cross-validation was performed to estimate the generalization accuracy of the generated fuzzy rule model. Different levels of distortion were randomly generated on the training data, thus simulating different amounts of missing values for each input feature. In addition to the method discussed here, we also implemented the approach presented in [1], the so-called *best guess*, to compare our method. Furthermore, to establish a base line, missing values were also replaced by the overall mean and zero values.

### *A. Satimage Dataset*

The SatImage Dataset contains 4,435 training and 2,000 test cases split into 6 classes in a 36-dimensional feature space. Table I shows the results on this data. The first column shows the percentage of missing values, followed by four evaluation parts, which are our approach to incorporate missing values, the *best guess*, the mean, and zero replacement technique. All of them display the classification error in percentage and the number of generated rules. Our algorithm works well on this data set, always generating almost the same number of rules for all levels of distortion. The classification accuracy is comparable to the other procedures. Only the *best guess* approach is able to generate fewer rules. The substitution with mean leads to a model which performs worse at higher levels of distortion and generates a larger number of rules. Replacing with zero increases this effect even more. Both, the dynamic *best guess* replacement method and our approach are clearly superior to the static replacement with fixed values for this data set.

TABLE II

RESULTS ON THE SHUTTLE DATABASE.

| Miss. | Incorp. Miss. | | *Best Guess* | | Use Mean | | Use Zero | |
|---|---|---|---|---|---|---|---|---|
| [%] | Error | $\lvert\mathbf{R}\rvert$ | Error | $\lvert\mathbf{R}\rvert$ | Error | $\lvert\mathbf{R}\rvert$ | Error | $\lvert\mathbf{R}\rvert$ |
| 0 | 0.0 | 20 | 0.0 | 20 | 0.0 | 20 | 0.0 | 20 |
| 1 | 0.01 | 29 | 0.0 | 16 | 0.01 | 34 | 0.01 | 39 |
| 5 | 0.01 | 89 | 0.0 | 17 | 0.0 | 84 | 0.01 | 176 |
| 10 | 0.02 | 159 | 0.0 | 11 | 0.0 | 143 | 0.0 | 476 |
| 20 | 0.1 | 1038 | 0.0 | 14 | 0.0 | 372 | 0.0 | 1143 |
| 40 | 3.3 | 8679 | 0.0 | 10 | 0.04 | 1375 | 0.06 | 2376 |
| 60 | 3.5 | 7300 | 0.01 | 24 | 0.0 | 2876 | 0.0 | 3433 |

### *B. Shuttle Dataset*

The Shuttle Database consists of 43,314 training and 14,442 test cases along 9 dimensions and 3 classes (all other classes with occurrences below $1\%$ were removed for this experiment). Table II shows results on this benchmark. In general, only a small number of rules are necessary to model the data. Using more and more missing values, the model generated by the proposed approach contains an ever-increasing number of rules. In sharp contrast, the *best guess* method creates models with, essentially, a constant number of rules. This effect is due to an interesting property of this data. One class is separable from the others along an axes-parallel decision line. However, points of different classes lie arbitrarily close to this line, making it crucial that no other features are used for classification except this one. In the case of our method proposed here, every time this particular feature is missing, the algorithm is forced to make a decision using one of the remaining attributes, which essentially renders the resulting rules useless. Not even mean or zero replacement generates such an enormous amount of rules for higher level of distortion. This demonstrates one disadvantage of the approach discussed here—in case all relevant features are missing the algorithm will generate sub-optimal rules.

### *C. Remaining Data Sets*

The remaining experiments were conducted on the Letter, Australian Credit, Pima Indians, Segmentation, and Vehicle Data Sets (see tables III, IV, V, VI, VII). The general trend

TABLE III

RESULTS ON THE LETTER DATASET.

| Miss. | Incorp. Miss. | | *Best Guess* | | Use Mean | | Use Zero | |
|---|---|---|---|---|---|---|---|---|
| [%] | Error | $\lvert\mathbf{R}\rvert$ | Error | $\lvert\mathbf{R}\rvert$ | Error | $\lvert\mathbf{R}\rvert$ | Error | $\lvert\mathbf{R}\rvert$ |
| 0% | 16.2 | 2967 | 16.2 | 2967 | 16.2 | 2967 | 16.2 | 2967 |
| 1% | 17.7 | 3193 | 16.4 | 3069 | 16.2 | 3228 | 17.3 | 3499 |
| 5% | 22.8 | 3749 | 19.0 | 3540 | 19.7 | 4168 | 18.4 | 6022 |
| 10% | 29.5 | 4198 | 20.8 | 3828 | 24.5 | 5071 | 23.8 | 8632 |
| 20% | 21.1 | 5244 | 25.5 | 4030 | 32.3 | 6581 | 32.8 | 10978 |
| 40% | 57.3 | 7554 | 36.8 | 3496 | 51.8 | 8733 | 52.4 | 12328 |
| 60% | 54.3 | 5773 | 52.5 | 2317 | 67.9 | 10350 | 76.0 | 12307 |

TABLE IV

RESULTS ON THE AUSTRALIAN CREDIT APPROVAL.

| Miss. | Incorp. Miss. | | *Best Guess* | | Use Mean | | Use Zero | |
|---|---|---|---|---|---|---|---|---|
| [%] | Error | $\lvert\mathbf{R}\rvert$ | Error | $\lvert\mathbf{R}\rvert$ | Error | $\lvert\mathbf{R}\rvert$ | Error | $\lvert\mathbf{R}\rvert$ |
| 0 | 18.8 | 129 | 18.8 | 129 | 18.8 | 129 | 18.8 | 129 |
| 1 | 18.6 | 132 | 17.2 | 131 | 17.8 | 131 | 17.5 | 129 |
| 5 | 16.2 | 130 | 17.4 | 127 | 19.3 | 129 | 19.4 | 140 |
| 10 | 20.3 | 137 | 18.8 | 127 | 19.7 | 145 | 20.1 | 156 |
| 20 | 24.1 | 142 | 22.2 | 121 | 20.9 | 161 | 19.9 | 204 |
| 40 | 35.1 | 163 | 21.9 | 104 | 28.7 | 187 | 28.1 | 246 |
| 60 | 35.1 | 159 | 18.4 | 66 | 36.5 | 238 | 30.7 | 231 |

TABLE V

RESULTS ON THE PIMA INDIANS DIABETES DATA.

| Miss. | Incorp. Miss. | | *Best Guess* | | Use Mean | | Use Zero | |
|---|---|---|---|---|---|---|---|---|
| [%] | Error | $\lvert\mathbf{R}\rvert$ | Error | $\lvert\mathbf{R}\rvert$ | Error | $\lvert\mathbf{R}\rvert$ | Error | $\lvert\mathbf{R}\rvert$ |
| 0 | 25.9 | 155 | 25.9 | 155 | 25.9 | 155 | 25.9 | 155 |
| 1 | 27.8 | 157 | 25.9 | 151 | 28.2 | 153 | 26.8 | 156 |
| 5 | 27.3 | 162 | 28.1 | 150 | 26.3 | 160 | 27.8 | 154 |
| 10 | 28.3 | 165 | 25.9 | 145 | 29.0 | 171 | 28.9 | 160 |
| 20 | 29.2 | 187 | 29.0 | 135 | 30.0 | 176 | 30.8 | 164 |
| 40 | 30.4 | 231 | 29.4 | 74 | 30.5 | 186 | 30.3 | 186 |
| 60 | 31.3 | 222 | 31.9 | 40 | 35.0 | 198 | 30.0 | 213 |

TABLE VI

RESULTS ON THE SEGMENTATION DATA.

| Miss. | Incorp. Miss. | | *Best Guess* | | Use Mean | | Use Zero | |
|---|---|---|---|---|---|---|---|---|
| [%] | Error | $\lvert\mathbf{R}\rvert$ | Error | $\lvert\mathbf{R}\rvert$ | Error | $\lvert\mathbf{R}\rvert$ | Error | $\lvert\mathbf{R}\rvert$ |
| 0 | 3.8 | 118 | 3.8 | 118 | 3.8 | 118 | 3.8 | 118 |
| 1 | 3.7 | 122 | 4.2 | 120 | 4.3 | 149 | 4.0 | 128 |
| 5 | 4.7 | 147 | 5.2 | 134 | 4.4 | 343 | 5.1 | 177 |
| 10 | 4.8 | 160 | 6.0 | 148 | 5.4 | 634 | 7.0 | 256 |
| 20 | 6.5 | 181 | 8.7 | 161 | 7.0 | 971 | 11.5 | 429 |
| 40 | 15.4 | 266 | 21.6 | 177 | 16.9 | 1336 | 29.3 | 691 |
| 60 | 29.9 | 428 | 30.7 | 200 | 65.2 | 1257 | 58.6 | 912 |

TABLE VII

RESULTS ON THE VEHICLE SILHOUETTES DATA.

| Miss. | Incorp. Miss. | | *Best Guess* | | Use Mean | | Use Zero | |
|---|---|---|---|---|---|---|---|---|
| [%] | Error | $\lvert\mathbf{R}\rvert$ | Error | $\lvert\mathbf{R}\rvert$ | Error | $\lvert\mathbf{R}\rvert$ | Error | $\lvert\mathbf{R}\rvert$ |
| 0 | 32.9 | 195 | 32.9 | 195 | 32.9 | 195 | 32.9 | 195 |
| 1 | 33.9 | 185 | 32.6 | 191 | 36.0 | 192 | 31.9 | 193 |
| 5 | 32.6 | 184 | 32.1 | 194 | 35.5 | 269 | 33.9 | 201 |
| 10 | 34.3 | 191 | 34.5 | 198 | 36.7 | 363 | 35.4 | 218 |
| 20 | 32.7 | 181 | 38.4 | 183 | 46.9 | 461 | 39.5 | 243 |
| 40 | 38.7 | 192 | 45.6 | 134 | 62.2 | 533 | 47.3 | 280 |
| 60 | 42.3 | 200 | 50.1 | 94 | 73.7 | 538 | 54.9 | 303 |

from the experiments discussed above remains unchanged. Both static replacement methods tend to produce dramatically larger rule sets for medium to large amounts of distortion. The proposed method performs equally well in comparison to the *best guess* approach but tends to generate larger rule sets. In one example, the segmentation data, the proposed method did generalize substantially better than the *best guess* method.

## V. CONCLUSION

In this paper an approach was discussed that incorporates missing values during a fuzzy rule learning process. The proposed methodology can handle missing values in a natural way without any need for artificial replacement of the missing values themselves. Results on benchmark datasets show that the algorithm performs well and outperforms standard replacement algorithms in the number of rules and with respect to classification accuracy. Our method generates, at times, better classification results in comparison to an earlier approach that uses the evolving model to compute dynamic estimates for the missing values.

## VI. ACKNOWLEDGMENTS

This work was supported by the DFG Research Training Group GK-1042 “Explorative Analysis and Visualization of Large Information Spaces”.

## REFERENCES

[1] M. R. Berthold and K.-P. Huber. *Tolerating Missing Values In A Fuzzy Environment*. IFSA World Congress, 1:359–361, 1997.

[2] M. R. Berthold and K.-P. Huber. *Constructing fuzzy graphs from examples*. Intelligent Data Analysis, 3(1), 1999.

[3] M. R. Berthold. *Mixed fuzzy rule formation*. International Journal of Approximate Reasoning (IJAR), 32:67-84, 2003.

[4] Thomas R. Gabriel and Michael R. Berthold. *Influence of fuzzy norms and other heuristics on “Mixed Fuzzy Rule Formation”*. International Journal of Approximate Reasoning (IJAR), 35:195–202, Elsevier, 2004.

[5] D. Michie, D. J. Spiegelhalter, and C. C. Taylor, editors. *Machine Learning, Neural and Statistical Classification*. Ellis Horwood Limited, 1994.

[6] P. M. Murphy and D. W. Aha. *UCI repository of machine learning databases*. Machine-readable data repository at ics.uci.edu in pub/machine-learning-databases.

[7] V. Tresp and S. Ahmad. *Some solutions to the missing feature problem in vision*. In Stephen J. Hanson, Jack D. Cowan, and C. Lee Giles, editors. Advances in Neural Information Processing Systems, 5:393–400, California, 1993. Morgan Kaufmann.

[8] L. A. Zadeh. *Fuzzy sets*. Information and Control, 8:338-353, 1965.