Estudio de la influencia de los métodos de imputación en Redes Neuronales de Base Radial para clasificación

Julián Luengo
Dept. de Inteligencia Artificial y Ciencias de la Computación
ETS Ingeniería Informática
Univ. de Granada
18071 Granada
julianlm@decsai.ugr.es

Salvador García
Dept. de Inteligencia Artificial y Ciencias de la Computación
ETS Ingeniería Informática
Univ. de Granada
18071 Granada
salvagl@decsai.ugr.es

Francisco Herrera
Dept. de Inteligencia Artificial y Ciencias de la Computación
ETS Ingeniería Informática
Univ. de Granada
18071 Granada
herrera@decsai.ugr.es

Resumen

En la presente contribución, estudiaremos la presencia de valores perdidos en los datos usados por Redes Neuronales basadas en funciones de Base Radial, y como podemos superar su impacto negativo. La presencia de los valores perdidos no es extraña en las bases de datos con los que tratamos usualmente, aunque es habitual eliminarlos directamente o reemplazarlos por otros valores al uso. Queremos mostrar cómo el correcto uso de métodos de imputación puede mejorar el rendimiento del modelo de Red Neuronal basado en funciones de Base Radial, tanto en precisión como en generalización, en comparación con la ausencia de tratamiento o bien la eliminación de los valores perdidos.

1. Introducción

En el trabajo realizado, se ha aplicado un enfoque basado en las Redes Neuronales Artificiales junto a la presencia de información imprecisa, y más concretamente los valores perdidos (MV, utilizaremos el acrónimo de acepción en inglés por el que son conocidos, Missing Values). En particular, consideraremos un tipo de Redes Neuronales particulares como son las basadas en Funciones de Base Radial (usaremos el acrónimo de la acepción en inglés Radial Basis Function Network, RBFN) .Una de las principales herramientas del Data Mining es la inducción de reglas desde datos en bruto que se encuentran organizados en una base de datos. Normalmente, los datos de la vida real son imperfectos: erróneos, incompletos, inciertos e incluso vagos. Nos centraremos en una de las formas de datos incompletos más común: los valores perdidos.

Según se propone en [8, 2], la aleatoriedad de los MV se puede dividir en tres clases:

- Missing Completely at Random (**MCAR**). Es el nivel de aleatoriedad más alto. Ocurre cuando la probabilidad de que una instancia con un valor perdido para un atributo no depende ni de los valores conocidos ni del valor del atributo. En este nivel de aleatoriedad es posible usar cualquier método de tratamiento de los datos sin riesgo de introducir sesgo en los datos.

- Missing at random (**MAR**). Cuando la probabilidad de que una instancia contenga un valor perdido depende de los valores conocidos, pero no del valor perdido en si mismo.

- Not missing at random (**NMAR**). Cuando la probabilidad de que una instancia contenga un valor perdido puede depender del propio valor del atributo.

Se han propuesto muchos métodos en la literatura para tratar con los MV. Muchos de estos métodos, como la sustitución de casos, se desarrollaron para tratar con MV en encuestas, y presentan serios problemas cuando se aplican en Data Mining. Otros métodos pueden introducir sesgos a los datos si no se aplican con cuidado. La presencia del sesgo también depende de la aleatoriedad de los propios MV como hemos visto. Existen en la literatura métodos que asumen una u otra hipótesis de aleatoriedad en su estudio.En estudios en los que se trata de imputar (es decir, estimar un valor apropiado para dicho valor faltante) por medio de relaciones entre instancias, como mínimo un nivel de aleatoriedad MAR es necesario. Aquellas formas de estimación que tratan de inducir modelos de distribución de los datos usualmente usan un nivel NMAR (también denominado a veces "informativo").

Nosotros nos centraremos en el uso de métodos de imputación para tratar con los MV en nuestro estudio, ya que permiten adaptar los conjuntos de datos con valores faltantes de forma transparente al método que se empleará para extraer conocimiento. Los conjuntos de datos que se hayan preprocesado de esta manera se emplearán con un modelo de Red Neuronal Artificial para estudiar y comparar su comportamiento. El uso de la red neuronal en el estudio será el de clasificar, por lo que todos las bases de datos tienen un atributo de clase, y el resto se consideran como atributos de entrada.

El resto del documento se ha organizado como sigue. En la Sección 2 se describen los métodos de imputación que se han empleado en el estudio. En la Sección 3 se realiza una descripción breve del modelo RBFN de red neuronal empleado junto a un resumen de los Bases de datos sobre los que se ha realizado el estudio. En la Sección 4 se muestra el estudio experimental y los resultados obtenidos, así como un análisis de los mismos. Finalmente, en la Sección 5 exponemos nuestras conclusiones.

2. Métodos de imputación empleados

En esta sección describiremos brevemente los métodos de imputación que conforman la comparativa, así como los parámetros usados en la experimentación con cada uno.

2.1. Descripción de los métodos de imputación

Los métodos de imputación que se han empleado en el estudio han sido:

- No hacer nada (**none**). Como indica su nombre, no aplicamos ningún tipo de imputación a la Base de Datos. De esta forma, podemos verificar si los métodos de imputación permiten que la red neuronal mejore en su aprendizaje y posterior comportamiento (en el test). En [6] podemos encontrar un estudio previo de métodos de imputación que puede ser útil como guía, pero a diferencia de nosotros, no emplean un método de Machine Learning después.
- Filtrado de instancias (**ignore_missing**). Por medio de este método, todas aquellas instancias que tengan al menos un valor perdido se eliminan del conjunto.
- Valor más común/media de los valores Global (**MostCommonValue**)[7]. Este método es muy sencillo: para aquellos atributos nominales, el valor perdido se reemplaza con el valor más común para todo el conjunto de datos. Para aquellos atributos numéricos, se reemplaza por la media global.
- Valor más común/media de los valores de la clase (**CMostCommonValue**)[7]. De forma similar al método anterior, la diferencia radica en que sólo aquellas instancias con la misma clase que la considerada se usan para imputar el valor perdido. La estrategia es idéntica: usar el valor más repetido para los atributos nominales, y la media para los numéricos.

- Imputación mediante Máquinas de Vectores de Soporte (**svmImpute**)[5]. Se trata de un modelo de regresión basado en Máquinas de Vectores de Soporte (SVM del término en inglés Support Vector Machines) empleado para sustituir los MV, es decir, se sitúan los atributos de decisión (el atributo de clase) como los atributos de entrada para el algoritmo SVM, y los atributos perdidos como el atributo sobre el que se desea aplicar la regresión y de esta forma predecir el valor. Para ello, primero seleccionamos los ejemplos que no contienen ningún valor perdido. En segundo lugar, aquellos atributos que contengan MV se toman como atributos de salida para la SVM, y los atributos de clase como entrada. Entonces es posible usar la regresión para predecir cada uno de los valores faltantes que encontremos.

- Imputación usando K-Vecinos Más Cercanos (KNN del inglés K-Nearest Neighbors) (**knnImpute**)[2]. Empleando este modelo basado en instancias, cada vez que se encuentra una instancia con un valor perdido, se calculan los k vecinos más cercanos y se imputa el valor a partir de ellos. Para aquellos atributos que sean nominales, se emplea el valor más común entre todos los vecinos como estimación, y para los atributos numéricos, la media aritmética. Como medida de distancia hemos usado la distancia Euclídea.

- Imputación mediante K-NN ponderado (**wknnImpute**)[13]. El algoritmo K-NN ponderado selecciona los ejemplos con valores similares (en términos de distancia) a la que se está considerando para imputar de forma idéntica a la del método K-NN simple. A la hora de estimar el valor que se imputará, se tiene en cuenta la distancia a los diferentes ejemplos vecinos para calcular el valor: una media ponderada, o considerando el valor nominal repetido un mayor número de veces en base a la distancia.

- Imputation K-medias Clustering (**kmeansImpute**)[3]. Dado un conjunto de instancias, el objetivo global del clustering es dividir el conjunto de datos en grupos, basados en la similitud de los objetos de forma que se minimice la distancia intra-cluster. En el clustering de K-medias, la forma de medir la diferencia intra-cluster es la suma de la distancias de los objetos que lo conforman al centroide del cluster. El centroide representa el valor medio de los objetos del cluster. Una vez que los clusters han convergido, se rellenan todos aquellos atributos que estén incompletos a partir de la información del cluster. En concreto, se consideran todos los objetos del cluster como vecinos al considerado, y se usa una estrategia de vecinos más cercanos para sustituir los valores como se ha descrito anteriormente.

- Imputación con Fuzzy K-means Clustering (**fkmeans**)[1, 3]. En clustering difuso, cada ejemplo x_i tiene una función de pertenencia que describe el grado en el que dicho objeto pertenece a cierto cluster v_k. En el proceso de actualizar las funciones de pertenencia y los centroides, se consideran solamente las instancias completas. En el desarrollo del algoritmo no podemos asignar el ejemplo a un cluster en concreto (representado por el centroide, como hicimos en K-medias), dado que cada instancia pertenece a los K cluster con diferentes grados de pertenencia. Los MV se reemplazan basándose en la información acerca de los grados de pertenencia y los centroides.

- Event Covering (**EventCovering**). Basado en el trabajo de Wong et al. [14], se aproxima un modelo mixto (numérico y nominal) de probabilidad mediante uno discreto. En primer lugar, se requiere que los atributos numéricos se discreticen, mediante un criterio de mínima pérdida de información. Manejando una n-tupla de atributos mixtos como una discreta, los autores proponen un nuevo acercamiento estadístico para la síntesis de conocimiento basado en un modelo de cluster. Como

principal ventaja, este método no requiere un escalado ni una normalización previas. Por sintetizar los datos en conocimiento estadístico, se refieren a los siguientes procesos: 1) sintetizar y detectar patrones que indiquen interdependencia estadística; 2) agrupar los datos dados en clusters basados en dichas interdependencias; 3) interpretar los patrones subyacentes para cada cluster. Con el método desarrollado, es posible hallar los MV gracias a los clusters.

Existen métodos más específicos, incluso derivados del campo de la Bioinformática. En las revisiones [11, 4, 6] podemos encontrar un buen compendio de métodos de imputación, algunos de los cuales han sido tratados aquí.

2.2. Parámetros de los métodos de imputación

En la Tabla 1 recogemos los parámetros de cada uno de los métodos de imputación que hemos usado, si los tiene. Los valores de los parámetros se han tomado como los recomendados por los diferentes autores de las referencias correspondientes.

Tabla 1: Parámetros de los métodos

Método	Parámetro
svmImpute	Kernel: RBF C: 1.0 Epsilon: 0.001 shrinking: No
knnImpute, wknnImpute	K: 10
kmeansImpute	K: 10
fkmeans	K: 3
EventCovering	T: 0.05

3. Descripción del modelo de Red Neuronal

Para nuestro estudio, vamos a usar un modelo de Red Neuronal bien conocido como es RBFN [10]. Estas redes neuronales tienen un buen comportamiento para la aproximación de funciones y reconocimiento de patrones debido a su estructura topológica simple y su capacidad de revelar como se desarrolla el aprendizaje de una forma explícita. Una función de base radial es una función que se construye con un criterio de distancia respecto a un centro (es decir, de forma parecida a una gaussiana). Las funciones de base radial se han aplicado en el área de las redes neuronales como sustituto de la función de activación sigmoide en perceptrones multicapa. Tienen 2 capas de procesamiento: en la primera, la entrada se asigna a cada RBF en la capa ocula. Las redes neuronales RBF tienen la ventaja de no caer de la misma forma en mínimos locales de la misma forma que los perceptrones multicapa. El número de neuronas es fijo a priori. Para nuestra red hemos usado 50 neuronas, valor que será fijo para toda la ejecución.
Es importante señalar como los MV son tratados por las redes neuronales. Cuando encuentran un valor perdido en una instancia que queremos clasificar o que usamos para entrenar, el valor se convierte a 0. En [9] se menciona esto, que trata de eliminar la influencia de los MV en la salida y en la actualización de los pesos. También es importante indicar que usar este criterio junto a un método de imputación (es decir, imputar el conjunto de entrenamiento pero no el de test o viceversa), tiene malos resultados tal y como indica Popov. Así pues, es importante imputar todo el conjunto de datos.

4. Resultados experimentales

En esta sección se detallan los resultados de los experimentos, y se analizan dichos resultados.

4.1. Bases de datos utilizadas

Hemos escogido un conjunto de Bases de Datos del repositorio de la UCI. Se trata de un repositorio de datos conocido, y sus Bases de Datos son frecuentemente usados en el área. En total, se han considerado 8 Bases de Datos para hacer el estudio. En la Tabla 2 resumimos las propiedades de dichas Bases de Datos. Indicamos el porcentaje de MV respecto al to-

Tabla 2: Bases de Datos usadas para la experimentación

Base de Datos	#Instancias	#Atributos	#Clases	%MV	%Inst. con MV
Crx	689	16	2	0.61	5.37
Breast	286	10	2	0.31	3.15
Wisconsin	699	10	2	0.23	2.29
Cleveland	303	13	5	0.14	1.98
Monks	432	7	2	15.00	61.34
Thyroid	7200	22	3	15.00	96.58
Wine	178	14	3	15.00	87.08
Iris	150	4	3	15.00	48.00

tal de valores de la Base de Datos (columna %MV), y el porcentaje de instancias que contienen al menos un valor perdido (columna %Inst. con MV). Dado que Iris originalmente no tiene MV, hemos introducido artificialmente cierta cantidad de ellos, de forma aleatoria (es decir, un esquema *MCAR*). El número de MV es 90, que constituye el 15 % de 600 valores presentes en el conjunto. De igual forma, para las Bases de Datos Monks, Thyroid y Wine hemos añadido un porcentaje de MV del 15 % usando el mismo esquema *MCAR*. Las Bases de Datos Crx, Breast, Wisconsin y Cleveland tenían originalmente MV, por lo que no podemos suponer ningún nivel de aleatoriedad a priori.

4.2. Marco experimental

Para hacer la experimentación, hemos usado un esquema muy conocido: 10-fold cross validation. En este método, cada Base de Datos se divide en 10 partes de aproximadamente el mismo tamaño. Iterativamente se escogen 9 de dichas partes para entrenar la red neuronal, y la parte restante para realizar el test. Como valor significativo del éxito del entrenamiento de la red, se toma el valor medio de los 10 test realizados. Como inicialmente las Bases de Datos tenían MV, encontraremos que hay MV tanto en la partición de train como de test. La imputación se aplicará de forma previa a la clasificación.

4.3. Resultados

Una vez completada la experimentación, los resultados obtenidos en porcentaje de acierto son los que se detallan en la Tabla 3. En la Figura 1 se muestra de forma gráfica los mismos resultados agrupados por Base de Datos. En la Tabla 3 se han resaltado aquellos resultados que constituyen la mejor opción, es decir, producen el mejor porcentaje de acierto.

Queremos resaltar dos hechos importantes principalmente. En primer lugar el método *none* nunca es el que mejores resultados nos ofrece, con lo que podemos justificar el uso de los métodos de imputación para mejorar el rendimiento de nuestro modelo. Pero más interesante aún es comprobar como el método *EventCovering* es el mejor en 6 de las 8 Bases de Datos empleados. Este método se ajusta muy bien al modelo RBFN, y podemos hallar su razón en la naturaleza del propio método, basada en el agrupamiento de instancias mediante elementos probabilísticos, similar al fundamento de las RBF.

El método *svmImpute*, a pesar de emplear un kernel basado también en funciones de base radial RBF, no consigue obtener un buen funcionamiento para nuestro modelo. Los algoritmos de imputación basados en KNN o bien en clustering clásico tampoco consiguen funcionar bien, y son superados en alguna ocasión incluso por métodos sencillos como sustituir por la moda o la media global o de la clase.

Eliminar aquellas instancias con MV tampoco es el mejor método en ninguna de las ocasiones, aunque tampoco es el peor. Dado que

Tabla 3: Porcentaje de Acierto en Test

Método Imputación	Crx	Breast	Wisconsin	Cleveland	Monks	Thyroid	Wine	Iris
ignore_missing	45.41	67.26	87.86	34.69	86.31	62.92	10.00	83.57
EventCovering	**71.43**	68.92	**93.86**	**52.77**	78.81	**84.40**	**75.16**	**88.67**
knnImpute	44.56	66.47	84.13	32.56	87.37	67.78	62.45	70.67
wknnImpute	44.56	**69.98**	84.70	33.70	87.15	61.54	56.18	72.67
kmeansImpute	44.12	67.21	85.28	33.29	85.17	65.33	59.48	83.33
fkmeans	46.01	68.60	85.71	27.74	87.36	61.44	56.76	67.33
svmImpute	44.56	64.70	85.41	38.53	85.26	62.60	57.32	85.33
MostCommonValue	44.56	64.29	85.41	39.69	85.73	78.33	66.83	72.00
CMostCommonValue	45.57	69.97	86.57	33.42	**88.71**	74.58	64.64	80.00
none	47.31	67.25	84.14	32.91	81.15	65.92	61.86	71.33

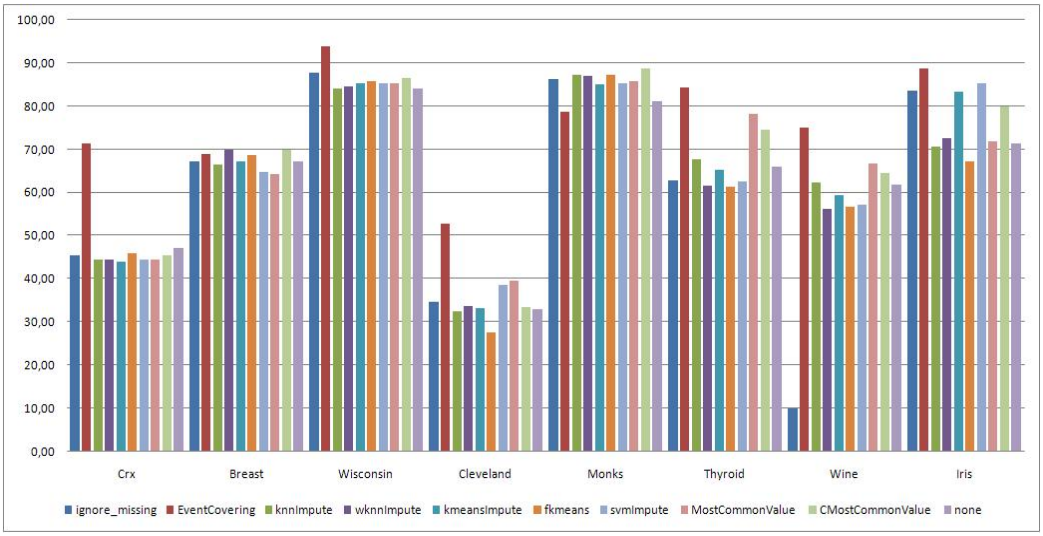

Figura 1: Porcentaje de Acierto en Test

estamos tratando con conjuntos sin un porcentaje de MV muy alto respecto al total de valores del conjunto, descartar este bajo porcentaje no incurre en una penalización muy grave. En el caso de que existiese un porcentaje mucho mayor de valores faltantes, podríamos encontrar un descenso importante en el rendimiento, o incluso que no se pudiese aplicar, al existir siempre un valor perdido en cada ejemplo del conjunto de datos.

Los métodos de imputación usualmente ofrecen mejores resultados si imputan los valores de atributos menos importantes a partir de la información de los atributos importantes (y completos, claro está). Es incluso razonable pensar que si un atributo está muy dañado por los MV, pueda ser filtrado por completo, pero para ello sería necesario conocer la importancia de dicho atributo para la clasificación.

Para comprobar que efectivamente el método *EventCovering* es el mejor de forma estadística, hemos aplicado un Test de Wilcoxon [12] de parejas. En la Tabla 4 se observan los rankings y valores p obtenidos en los emparejamientos de todos los métodos de imputación con el método *EventCovering*. El método *EventCovering* siempre es el ganador en los rankings, por lo que reflejamos el valor p de los emparejamientos.

Para un valor $\alpha = 0{,}05$ hemos encontrado

Tabla 4: Resultados del Test de Wilcoxon

CMostCommonValue	valor p
ignore_missing	0.05
knnImpute	0.025
wknnImpute	0.036
kmeansImpute	0.036
fkmeans	0.036
svmImpute	0.036
MostCommonValue	0.036
CMostCommonValue	0.93
none	0.025

que *EventCovering* es mejor que todos para este nivel de confianza, salvo con *CMostCommonValue* con un 0,93 e *ignore_missing* con un valor p de 0,05 justo. No obstante método *CMostCommonValue* puede afirmarse que es peor que EventCovering con un valor de confianza de 0,1, que es aceptable. También destacamos que el método *none* tiene un valor p muy bajo.
En definitiva, los resultados nos permiten afirmar que:

- El uso de las técnicas de imputación nos han permitido mejorar los resultados para los diferentes modelos de red en todas las ocasiones.
- Las mejoras de las técnicas de clustering como *knnImpute*, *kmeansImpute* y *fkmeans* son similares, y pocas veces competitivas. *EventCovering* es diferente al estar basada en entropía y funciona bien con RBFN.
- Es usual ver que las técnicas basadas en clustering y/o vecino más cercano tienen un comportamiento complementario con *EventCovering*: cuando ésta última funciona bien, las técnicas de cluster no, y viceversa.
- Cuando los MV se encuentran en un mayor número de instancias, el método de eliminar ejemplos con valores perdidos es más competitivo. No obstante, si el valor de instancias afectadas es muy alto, podemos encontrarnos con que estamos realizando un entrenamiento con pocas instancias, y que el modelo usado no generalice bien.

A partir de los datos obtenidos, y de los resultados observados, hemos podido ver que los métodos de imputación son capaces de ofrecer mejoras en el rendimiento. No obstante, los incrementos en dicho rendimiento ofrecido por las técnicas depende de estos factores en orden:

1. La Base de Datos usada. Cada conjunto de datos tiene sus propias características, por lo que las capacidades del método estarán mejor o peor adaptadas al mismo. Cuanta más información de las relaciones entre atributos sea capaz de modelar, mejor estimará el valor y por tanto el entrenamiento y test mejorarán.
2. Los atributos con MV. De acuerdo al punto anterior, las relaciones entre atributos existen, y tienen diferente importancia para el entrenamiento de la red neuronal (es decir, hay atributos que tienen un "poder predictivo"mayor). Por tanto la presencia de MV en estos atributos tendrá un mayor impacto. Si además hay una alta correlación entre los atributos, es posible que la presencia de los MV no tenga un impacto tan decisivo. Esto se observa en las diferencias de porcentaje de acierto entre no aplicar imputación (*none*) y los métodos usados, puesto que varían con cada base de datos. Aquellas Bases de Datos con MV en atributos poco importantes, no mejorarán mucho su rendimiento, y aquellos que tengan sus atributos importantes afectados tendrán incrementos de precisión más importantes.

5. Conclusiones

En la presente contribución se ha mostrado la influencia de los métodos de imputación en los resultados obtenidos mediante una red RBFN, siempre de forma positiva. En definitiva, podemos concluir que el uso de métodos de imputación nos permiten mejorar la precisión del

modelo RBFN en clasificación. Incluso cuando aparecen MV en todos los atributos de la Base de Datos, se puede evitar tener que eliminar las instancias afectadas mejorando los resultados obtenidos por este medio. Se ha podido observar como el método *EventCovering* es capaz de mejorar sustancialmente el rendimiento de la red RBFN por encima del resto de esquemas de imputación.

Agradecimientos

Este trabajo de investigación ha sido posible gracias a la subvención del proyecto TIN2005-08386-C05-01.

Referencias

[1] Acuna, E. and Rodriguez, C.: The treatment of missing values and its effect in the classifier accuracy. In CCDMA, Springer-Verlag Berlin-Heidelberg (2004) 639–648.

[2] G.E.A.P.A. Batista, M.C. Monard: An analysis of four missing data treatment methods for supervised learning. Applied Artificial Intelligence **17** (2003) 519–533.

[3] Deogun J., Spaulding W. Shuart B. and Li D.: Towards Missing Data Imputation: A Study of Fuzzy K-means Clustering Method. Lecture Notes in Computer Science **3066** (2004) 573–579.

[4] Farhangfar, A., Kurgan, L., Pedrycz, W.: Experimental analysis of methods for imputation of missing values in databases. Intelligent Computing: Theory and Applications II **5421** (2004) 172–182.

[5] Feng, H.A B., Chen, G.C., Yin, C.D., Yang, B.B., Chen, Y.E.: A SVM regression based approach to filling in missing values Lecture Notes in Computer Science **3683** LNAI (2005) 581–587.

[6] Grzymala-Busse, J. W. and Hu, M.: A Comparison of Several Approaches to Missing Attribute Values in Data Mining. Rough Sets and Current Trends in Computing : Second International Conference, RSCTC 2000 Banff, Canada, October 16-19, 2000. Revised Papers Lecture Notes in Computer Science **2005** (2001) 378–385.

[7] Grzymala-Busse, J. W., Goodwin, L. K., Grzymala-Busse, W. J. and Zheng, X.: Handling Missing Attribute Values in Preterm Birth Data Sets. Lecture Notes in Computer Science **3642** (2005) 342–351.

[8] Little, R. J. and Rubin, D. B.: *Statistical Analysis with Missing Data* (John Wiley and Sons, New York, 1987).

[9] Popov, S.: Nonlinear Visualization of Incomplete Data Sets. International Computer Science Symposium 2006, LNCS **3967** (2006) 524–533.

[10] Rojas, R. and Feldman, J.: *Neural Networks: A Systematic Introduction* (Springer, 1996).

[11] Schafer, J. L. and Graham, J. W.: Missing data: our view of the state of the art. Psychol Methods **7** (2002) 147-77.

[12] Sheskin, D. J.: *Handbook of Parametric and Nonparametric Statistical Procedures* (CRC Press, 2000).

[13] Troyanskaya, O., Cantor, M., Sherlock, G., Brown, P., Hastie, T., Tibshirani, R., Botstein, D., Altman, R.B.: Missing value estimation methods for DNA microarrays. Bioinformatics **17** (2001) 520–525.

[14] Wong, A.K.C. and Chiu, D.K.Y.: Synthesizing statistical knowledge from incomplete mixed-mode data. IEEE Transactions on Pattern Analysis and Machine Intelligence **9** (1987) 796–805.