1. Descripción de las variables y valores estadísticos (mínimo, máximo, media, desviación, mediana, etc.). Estudia qué valores estadísticos son los convenientes según el tipo de variable y procede en consecuencia.

Los valores no son numéricos, así que no hay como calcular valores estadísticos. Podemos ponerlos en histogramas para verlos, pero no hay una análisis estadística sobre ellos.

1. Describe y realiza modificaciones en la base datos si lo consideras necesario. Por ejemplo, qué harías con valores nominales, si los hubiera.

Hemos puesto los valores en un *one hot encoder* para que sean mejor interpretados. Una vez puesto, tenemos los valores como vectores y podemos hacer análisis sobre ellos para intentar clasificarlos entre los clases existentes: venenosos o no.

1. Estudia si es necesario normalizar los datos y cómo lo harías. Procede a modificar la base de datos (normalizar) si lo consideras necesario.

Como los valores no son numéricos, no hay como normalizarlos, pues una vez que estarán involucrados en un *encoder*, los valores serán 0 o 1.

1. Detección de valores extremos (outliers) y descripción de qué harías en cada caso.

No hay outliers en el conjunto de datos. Hay solamente colecciones de características posibles en cada observación, una vez que los datos son categóricos.

1. Detección de valores perdidos (missing values) y descripción de cómo actuarías para solventar el problema.

En el conjunto de datos, es especificado que los datos tendrán un “?” cuando no estén disponibles, así que el algoritmo procura por valores iguales a “?” en la base de datos. En la especificación, esta dicho que hace falta de 2480 datos, que es exactamente lo que el algoritmo ha encontrado.

Es importante considerar algunas cosas cuanto a los datos:

* Hay valores faltando solamente en una característica llamada “*stalk-root*”;
* De las 8124 observaciones, hace falta este dato en 2480 de ellas, que es 30% de las observaciones;

Hay posibles mecanismos para actuar sobre datos indisponibles, que son: excluirlos/ignorarlos, crear un modelo que prediga valores faltosos o imputación.

De acuerdo con las análisis de (Acock, 2005), (Batista & Monard, 2003) y (Grzymala-Busse & Hu, 2001), todos usan imputación y hablan de los riesgos de sacar un atributo o sustituirlo sin cuidado. Ellos dicen que trabajar con un dato faltoso sin cuidado puede introducir *bias* en la base de datos, así que es algo muy serio.

De acuerdo con (Acock, 2005) y (Grzymala-Busse & Hu, 2001), imputar la media de los resultados es la peor solución, una vez que atenúa la varianza y también resuelta en valores imputados muy malos.

(Batista & Monard, 2003) dicen que imputar dados puede no ser útil y tampoco sano para la base da datos, una vez que hasta el método más avanzado de imputación solo puede aproximar el valor real. En sus experimentos, cuanto más atributos con valores faltosos y cuanto mayor la cantidad de valores faltosos, más simples se torna el problema, así que hay de tomarse cuidado para no simplificar demasiadamente el problema con la cantidad de valores imputados. Ellos también dicen que ignorar y excluir datos es uno de los métodos más principales de trabajar con valores faltosos. Pero para excluirlos, hay que tener en cuenta si el dato falta por ser perdido completamente aleatoriamente (MCAR), perdido aleatoriamente (MAR) o perdido no aleatoriamente (MNAR). De acuerdo con (Allison, 2001), la diferencia básica entre ellos dice si el dato va a inserir *bias* (MNAR) o no (MCAR) caso sea sacado.

(Grzymala-Busse & Hu, 2001) ha testado nueve métodos para intentar resolver valores faltosos. Ellos y (Batista & Monard, 2003) usan en método C4.5, y dicen que este es el mejor método para resolverlos. Todavía, Grzymala et al. concluye también que ignorar ejemplos con valores faltosos es tan bueno como usar el método C4.5 y ambos son mejores que todos los otros métodos.

En conclusión, antes de hacer una decisión, hay que decir cual es el tipo de pierda que se tiene en el atributo “*stalk-root*”: MCAR, MAR o MNAR. Para hacer eso, podemos empezar por hacer el teste de Little (Little, 1988) para decir se es un MCAR, poniendo en 0 los valores existentes y en 1 los valores faltosos. Los resultados son que la significancia del teste es 1, entonces rechazamos la hipótesis nula del teste de Little y la variable es un MCAR, así que podemos removerla.

Caso el teste no rechazase la hipótese nula, tendríamos que imputar valores y podríamos usar el algoritmo C4.5 para eso, como lo que es dicho en (Batista & Monard, 2003; Grzymala-Busse & Hu, 2001).

1. Buscar correlaciones entre:
   1. las variables predictoras, lo que permitirá ver si hay variables redundantes.
   2. variables predictoras y la clase (target).
2. Detecta, si hubiera, falsos predictores.
3. Estudia si fuera conveniente segmentar alguna de las variables.
4. Estudia si fuera conveniente crear nuevas variables sintéticas basada en las variables originales.

Acock, A. C. (2005). Working with missing values. *Journal of Marriage and Family*, *67*(4), 1012–1028. https://doi.org/10.1111/j.1741-3737.2005.00191.x

Allison, P. D. (2001). *Missing data*.

Batista, G. E. A. P. A., & Monard, M. C. (2003). An analysis of four missing data treatment methods for supervised learning. *Applied Artificial Intelligence*, *17*(5–6), 519–533. https://doi.org/10.1080/713827181

Grzymala-Busse, J. W., & Hu, M. (2001). A comparison of several approaches to missing attribute values in data mining. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *2005*, 378–385. https://doi.org/10.1007/3-540-45554-X\_46

Little, R. J. A. (1988). A test of missing completely at random for multivariate data with missing values. *Journal of the American Statistical Association*, *83*(404), 1198–1202. https://doi.org/10.1080/01621459.1988.10478722