UNIVERSIDAD DE GRANADA

Práctica 2. Preparación de datos.

Tratamiento Inteligente de Datos.

Máster Profesional en Ingeniería Informática

Curso académico 2022/2023

Autor

José Alberto Gómez García

Índice

1. Introducción.

En esta práctica de la asignatura abordaremos la preparación de la base de datos proporcionada en el fichero "*accidentes.xls".* Dicha preparación consistirá en la limpieza, tratamiento y manipulación de los datos, de diversas formas, para obtener el mejor árbol de decisión C4.5 posible, consiguiendo un modelo que permita predecir la gravedad de un accidente de tráfico de la mejor manera posible.

Como parte de la entrega, se adjunta un fichero "KNIME Workflow" (con los nodos reseteados), el cual contiene la gran mayoría de los experimentos realizados a lo largo de esta práctica. La entrega de Prado no permite adjuntar el fichero que se produce al no resetar los nodos, dado su elevado peso, por lo que se proporciona este enlace a Google Drive, desde el que se podrá descargar.

2. Primeros pasos.

2.1. Limpiar la base de datos.

Antes de importar nuestra base de datos en KNIME, la abriremos con Microsoft Excel y realizaremos un primer análisis, para saber con qué información contamos y sus características.

Lo primero que podemos observar es que tenemos una gran cantidad de datos distintos, cuyo significado se encuentra en la segunda hoja del documento. Si examinamos dicha hoja, veremos que muchas columnas presentan valores perdidos, pero que no han sido dejados en blanco, si no que cuentan con algún valor en específico.

Los valores que indican datos perdidos suelen ser 9, 99, 9998 o 9999, por lo que iremos a las columnas correspondientes y vaciaremos las celdas en las que se encuentran dichos valores numéricos. Esto se puede realizar fácilmente en Excel mediante la opción "Buscar y reemplazar".

La excepción a la regla que acabamos de mencionar es la columna *MAN_COL*. En este caso, la documentación indica que el valor perdido es 99, pero dicho valor no se encuentra en ninguna celda de la columna. En su lugar, tenemos valores 0 que no tienen explicación asociada en la documentación, por lo que consideraremos que el fichero contiene una errata, y los valores perdidos han sido marcados con un 0 en su lugar, por lo que los borraremos.

Aquellas variables que tienen valores perdidos disponen de una segunda columna donde dichas pérdidas han sido rellenadas (imputadas) siguiendo algún método desconocido. Debemos tener esto en cuenta pues no podremos usar ambas columnas simultáneamente. Además, resulta que la variable *WEEKDAY* no tiene ningún valor perdido, pero aun así posee

una segunda columna, *WKDY_I*. Dado que tenemos dos columnas con exactamente los mismos datos, borraremos la supuesta columna correspondiente a la imputación de datos. Además, en las columnas *HOUR* y *HOUR_I* se observa la aparición tanto del valor 0 como del 24, el cual técnicamente corresponde al mismo momento temporal. Sólo sucede en 25 instancias, pero se ha corregido para que todos los valores 24 pasen a ser 0.

También se observan algunas columnas, como *PED_ACC, TRAF_CON* o *MAN_COL,* en las que puede existir una gran cantidad de valores, pero en la práctica sólo se observan unos pocos.

Se ha consultado la matriz de correlación para ver si había alguna pareja de variables que estuvieran muy relacionadas, de forma que pudiéramos obviar una de ellas. Las únicas que presenta algún grado apreciable de correlación son *PED_ACC* y *NON_INVL,* lo cual tiene sentido. Dado que el valor es cercano a 0.65, decidimos mantener ambas columnas.

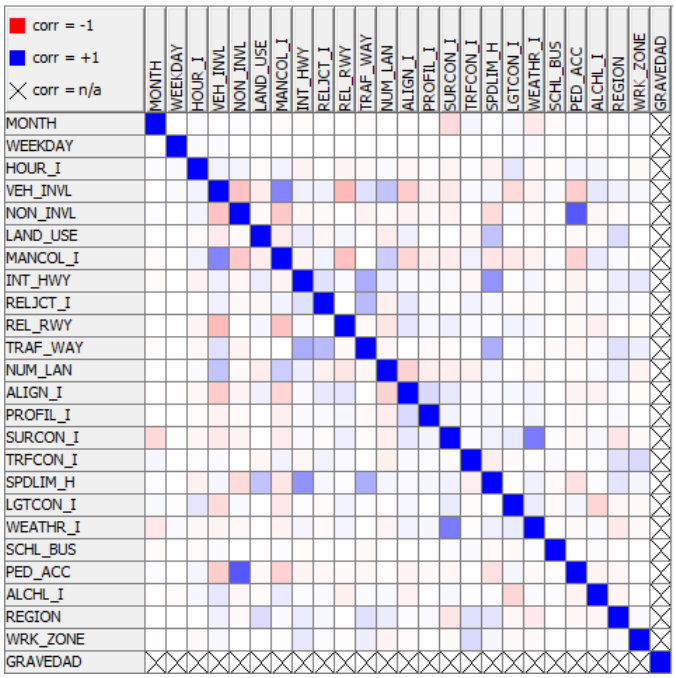


Figura 1. Matriz de correlación.

2.2. *Variable para la clasificación.*

De acuerdo con el guion de la práctica, debemos crear una variable que permitirá la clasificación de la gravedad del accidente de acuerdo con los valores de las columnas *FATALITIES, INJURY_CRASH* y *PRPTYDMG_CRASH.* He llamado a dicha variable *GRAVEDAD*.

Dado que tenemos libertad de elección, propongo que dicha variable *GRAVEDAD* tenga 2 posibles valores.

- Consideraremos un accidente como grave si en él ha fallecido alguien o han tenido lugar daños personales. Se representará mediante el valor numérico 0.
- Consideraremos un accidente como leve si únicamente han tenido lugar daños materiales. Se representará mediante el valor 1.

Esta división viene dada por la manera en que se asignan los valores de las 3 columnas mencionadas anteriormente. Por ejemplo, si pudieran darse combinaciones en que *INJURY_CRASH* y *PRPTYDMG_CRASH* fueran ambas verdaderas, podríamos clasificar la gravedad del accidente como media si hubiera daños personales y materiales, pero no fallecidos. Otra posible opción sería clasificar los accidentes en alta gravedad si hay fallecidos, media si hay daños personales y baja si solo hay daños personales.

Personalmente, prefiero utilizar el modelo binario propuesto, ya que representa más fielmente la idea que tengo de un accidente grave y uno leve.

Una vez hemos realizado este pre-procesamiento en Microsoft Excel guardamos el fichero, cuyo nombre será "*accidentes_with_missing_values.xls"* y lo importamos en el software KNIME. Nada más hacerlo, eliminaremos las columnas *FATALITIES, INJURY_CRASH* y *PRPTYDMG_CRASH,* pues son las que hemos utilizado para generar la variable de clasificación, y de mantenerlas el árbol de decisión las usaría y obtendríamos un 100% de precisión en las predicciones.

3. Evaluación del modelo.

Para comprobar la eficacia del árbol de decisión C4.5 dividiremos la totalidad de los datos en un conjunto de entrenamiento y otro de prueba. Dichos conjuntos, que contienen al 70% y 30% de las muestras respectivamente, han sido creados haciendo uso del nodo "Partitioning". Posteriormente, se le pasa el conjunto de entrenamiento al "Decision Tree Learner" para que genere el árbol de decisión. Al "Decision Tree Predictor" se le pasará el conjunto de prueba y el modelo generado por el nodo anteriormente mencionado. Finalmente, las predicciones realizadas serán evaluadas haciendo uso del nodo "Scorer", que nos proporciona métricas como la matriz de confusión, la precisión total y el coeficiente kappa de Cohen.

Este proceso de división de datos, generación del árbol de decisión y evaluación del mismo se encuentra contenido en metanodos de nombre "Evaluación", con el objetivo de proporcionar un entorno de trabajo más limpio y ordenado.

El único ajuste significativo sobre los nodos anteriores es marcar la opción de poda en "Decision Tree Learner"". Este proceso reduce el tamaño del árbol de decisión, reduciendo la complejidad del clasificador y mejorando la precisión al reducir el sobre-aprendizaje.

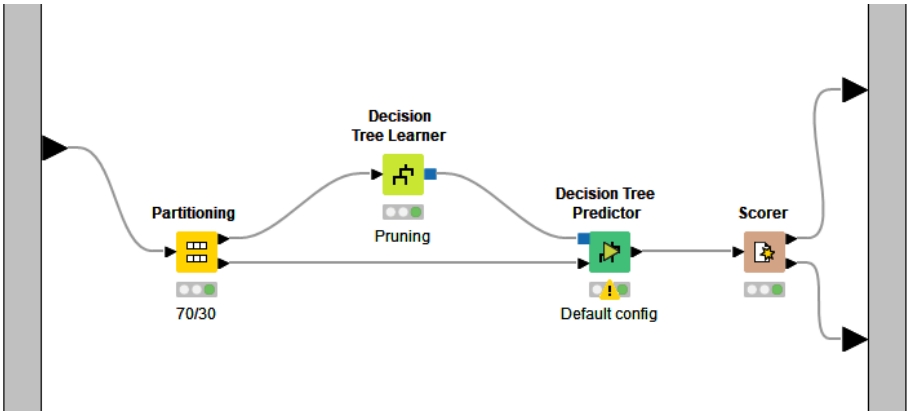


Figura 2. Metanodo de evaluación.

4. Discretización.

4.1. Algoritmo de clasificación C4.5 sobre la base de datos original.

En primer lugar, probamos a entrenar el modelo de clasificación sobre la base de datos sin tratar ningún valor perdido ni realizar ningún otro procesamiento más que obviar las columnas con valores imputados.

El modelo generado nos arroja una precisión del **59.9%** sobre el conjunto de prueba. El árbol de decisión generado es sumamente ramificado, hasta el punto de no poder mostrarlo entero sin que el ordenador se ralentice demasiado.

En la figura 3 se muestra parte del árbol de decisión. Cabe destacar que como primer criterio de decisión utiliza una de las variables categóricas cuyos valores se encuentran más desbalanceados.

Si en lugar de hacer uso de las columnas con valores perdidos, hacemos uso de las columnas que tienen valores imputados, la precisión del clasificador aumenta ligeramente hasta el **60.6%.** En este caso, hemos necesitado imputar nosotros las variables *TRAF_WAY* y *NUM_LAN,* las cuales tienen valores perdidos, pero no columna de valores imputados por defecto.

Los dos primeros niveles de este nuevo árbol de decisión son iguales al anterior, siendo a partir del tercer nivel cuando encontramos diferencias. En la rama derecha pasa a usarse la columna *SPDLIM_H,* dividiéndose en función de si este valor es mayor o menor igual a 62.5. También puede observarse como se usan varias veces las mismas características; *VEH_INVL* se usa tanto en el segundo como en el cuarto nivel del árbol.

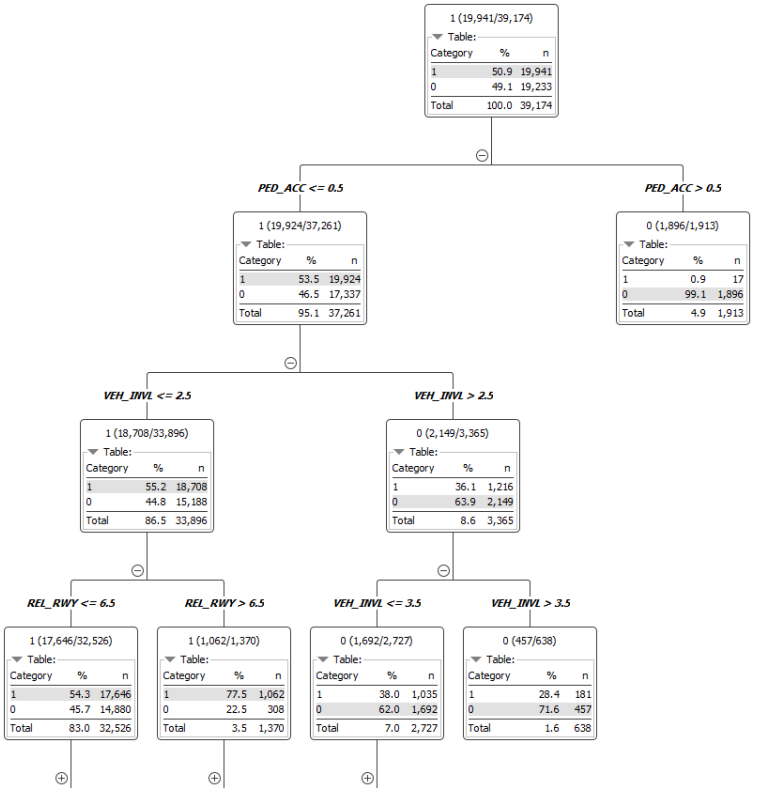


Figura 3. Árbol de decisión sin tratar los valores perdidos.

4.2. Aplicación del algoritmo de discretización CAIM sobre características numéricas.

Aunque la base de datos contiene multitud de características, sólo 7 (*MONTH, WEEKDAY, HOUR, VEH_INVL, NON_INVL, NUM_LAN* y *SPD_LIM* y sus contrapartes imputadas de haberlas*)* son numéricas. Por tanto, aplicaremos el algoritmo de discretización "automático" CAIM sólo sobre las versiones imputadas de ellas.

Los intervalos generados, dos por variable, por el algoritmo son los siguientes:

Variable	Intervalo 0	Intervalo 1
MONTH	[1.0; 10.5]	(10.5; 12.0]
WEEKDAY	[1.0; 1.5]	(1.5; 7.0]
HOUR_I	[0.0; 18.5]	(18.5; 24.0]
VEH_INVL	[1.0; 2.5]	(2.5; 23.0]
NON_INVL	[0.0; 0.5]	(0.5; 19.0]
SPDLIM_H	[0.0; 72.5]	(72.5; 75.0]

Tabla 1. Intervalos generados por CAIM para variables numéricas.

Nótese que no hemos tratado posibles outliers, como que haya 7 o más vehículos implicados en un accidente (lo cual sucede en 25 de los casi 56.000 ejemplos), o que haya 3 o más no-motoristas implicados (lo cual sucede en 31 casos). Dado el pequeñísimo número de ejemplos que suponen, hemos decidido ignorarlos.

La precisión del modelo resultante mejora muy ligeramente hasta el **61.1%**. El árbol de decisión generado es el mismo que el de la figura 2 en los primeros niveles, donde se ha sustituido los valores numéricos de *VEH_INVL* por los intervalos, pero dado que la frontera de dichos intervalos sigue siendo la misma que el valor numérico, no hay diferencia.

Si decidimos utilizar la base de datos, pero sin realizar la imputación de las columnas *TRAF_WAY* y *NUM_LAN,* realizada en el apartado anterior, la precisión es de un **60.7%.** Si decidimos usar las características numéricas de la base de datos sin imputar ningún dato, la precisión es del **60.8%.** Observamos que la diferencia entre imputar y no imputar datos no es demasiada, pues el algoritmo CAIM está preparado para lidiar con valores perdidos.

4.3. Discretización según criterio propio.

En este apartado vamos a realizar nosotros manualmente la discretización, teniendo en cuenta el significado de cada característica y los posibles valores que puede presentar. Mientras no se indique lo contrario, hemos utilizado los valores imputados que venían rellenos por defecto en la base de datos proporcionada.

Comencemos analizando las variables numéricas.

- Podemos agrupar los **meses** (variable *MONTH)* en función de la estación del año. Puede resultar lógico pensar que en verano debido al gran volumen de desplazamientos que hay en vacaciones, o en invierno debido a las reuniones familiares, pueda haber más accidentes.
- Un factor que puede parecer importante es la **hora del accidente** (variable *HOUR_I)*, pues puede que en horas punta como la entrada a salida del trabajo los accidentes revistan mayor gravedad dada la mayor densidad de tráfico. Si observamos el histograma de la figura 4, parece ser que las horas entre las 2pm y las 5pm son las que registran el mayor número de accidentes, mientras que suceden muchos menos entre las 0h y las 6am. Por tanto, proponemos tres posibles valores discretizados:
    - Frecuencia alta: [14:00 – 17:00]
    - Frecuencia media: [6:00 – 14:00) y (17:00 – 00:00)
    - Frecuencia baja: [0:00 – 6:00)

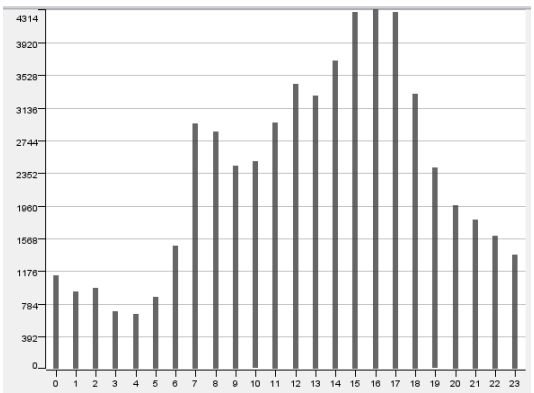


Figura 4. Número de accidentes en función de la hora.

- También puede resultar interesante tener en cuenta si el accidente se produjo en un día laborable (lunes a viernes) o en fin de semana (sábado y domingo), a pesar de que los datos nos indican que no existen grandes diferencias en el número de accidentes en función del día de la semana.
- El **límite de velocidad** (variable *SPD_LIM)* parece un factor determinante a la hora de predecir la gravedad de un accidente, por lo que discretizaremos esta variable en varios intervalos. Debemos tener en cuenta que en la documentación se nos indica que un límite de velocidad 0 indica que no hay límite, por lo que podemos considerar que son accidentes que se realizan a velocidad muy bajas, como maniobrando en un aparcamiento o en situaciones similares. Se proponen los siguientes intervalos, basándonos en los límites de velocidad en España:
    - Maniobrando: [0 – 5 mph]
    - Límite urbano: (5 – 20 mph]
    - Velocidad media: (20 – 40 mph]
    - Límite carretera secundaria: (40 – 60 mph]
    - Límite autovía/autopista: (60 – 75 mph]
- Respecto del **número de carriles** (*NUM_LAN)*, la documentación no indica si se refiere al número en un sentido o en ambos; asumiremos lo segundo. Podríamos clasificar en "carretera secundaria" si tiene 2 o menos carriles, y en "autovía/autopista" si tiene 3 o más carriles.
- Respecto del **número de vehículos** y **no-motoristas** implicados, realizaremos la misma distinción para ambos casos. Consideraremos como "poco" 2 o menos implicados, y como "mucho" 3 o más implicados.

Si analizamos los resultados de aplicar únicamente esta discretización de valores numéricos obtenemos que el modelo generado proporciona una precisión del **60.5%.** Este valor es ligeramente inferior al que obteníamos con la discretización automática del algoritmo CAIM sobre las mismas variables.

Parte del árbol de decisión generado se muestra en la figura 5. Se sigue manteniendo la misma estructura que en ocasiones anteriores, de manera que se utiliza en el segundo nivel el número de vehículos implicados en el accidente. Si nos fijamos en el cuarto nivel, vemos que no hay ningún accidente que cumpla las restricciones de los niveles anteriores y a su vez se haya producido a velocidad "de maniobra", por lo que podríamos plantearnos eliminar dicho intervalo y hacer que lo absorba el de "límite urbano".

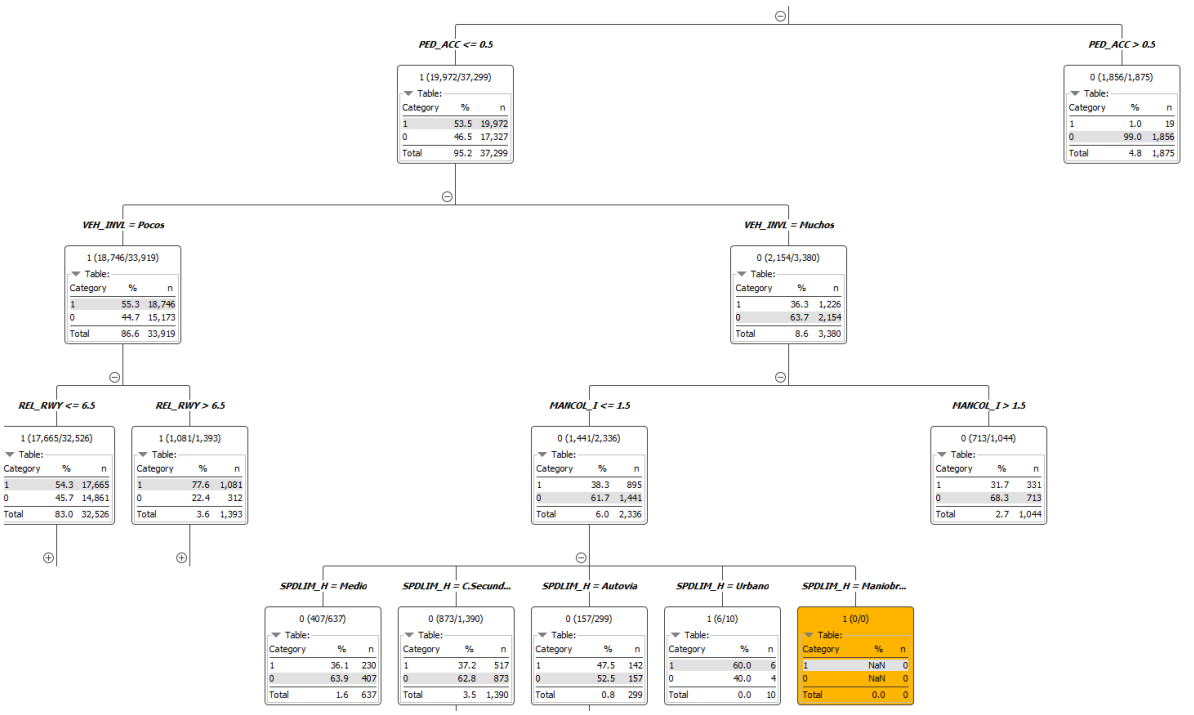


Figura 5. Parte del árbol de decisión tras discretización manual de variables numéricas.

La mayoría de las variables categóricas suelen tener un único valor asociado a "bueno/sí" o "malo/no", y el resto de los valores son gradaciones o diferentes casuísticas. Las excepciones son las variables *MANCOL_I, RELJCT_I, TRAFCON_I* y *PED_ACC,* que pueden tener multitud de valores; y *LAND_USE,* que ya corresponde a una discretización de la población que hay en la zona.

Se propone lo siguiente:

- No discretizar *MANCOL_I.* A pesar del posible número de valores, en la base de datos sólo se dan 6 de ellos, y no guardan demasiada relación entre sí.
- Discretizar *RELJCT_I* en dos grupos. Tendremos "no intersección" si el valor es 0, 8, 9 o 10 (0 y 10 corresponden al mismo caso) e "intersección" en el resto.
- No discretizar *TRAFCON_I,* dado que de hacerlo tendríamos un grupo con significado "sin señal" correspondiente a los valores 0 y "con señales", correspondiente al resto de valores, los cuales son muy numerosos, y considero que no ayudaría en nada.
- No discretizar *PED_ACC* ya que la mayoría de los valores de la base de datos son 0, y hay muchas situaciones que involucran tanto a ciclistas como a motoristas, a motoristas y peatones y a ciclistas y peatones, lo cual dificulta un valor no ambiguo a esta variable.
- Discretizar *LGTCON_I* y *WEATHR_I* en dos grupos. Un grupo de "bueno" correspondiente al valor 1, y un grupo de "malo" correspondiente al resto de valores (del 2 al 8).
- Discretizar *ALCOHOL_I* en dos grupos. "Alcohol" contendrá los valores 1, mientras que "No Alcohol" tendrá los valores 2 y 8.

Si solo aplicamos esta discretización de algunas variables categóricas, obtenemos una precisión del **61.3%**, algo superior a la que obteníamos si sólo utilizábamos discretización de variables numéricas.

Si combinamos la discretización de variables numéricas y categóricas, obtenemos una precisión es del **61.4%.** En la figura 6 se muestra parte del árbol de decisión generado.

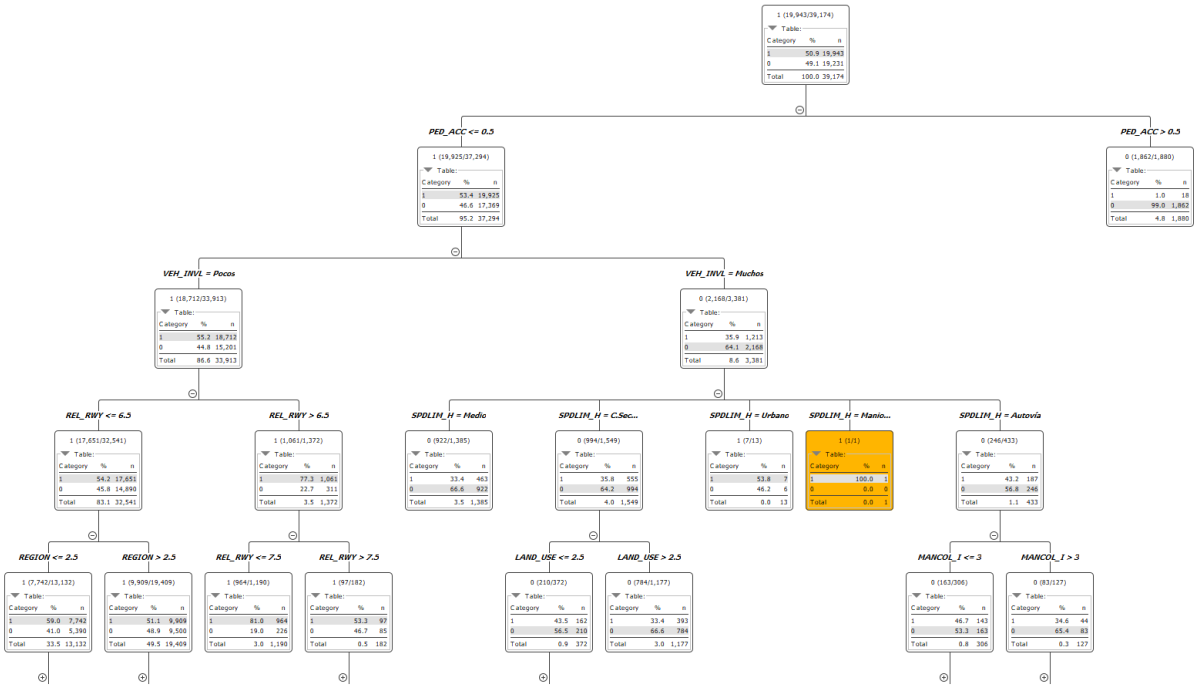


Figura 6. Parte del árbol de decisión tras discretización manual de las variables seleccionadas.

En la tabla 2 se muestran los resultados obtenidos al usar la discretización que proponemos, en función de si se utilizan los datos imputados (como hasta ahora) o si decidimos utilizar los datos con valores perdidos sin realizar imputación alguna.

Discretización	Valores imputados por defecto	Valores perdidos sin imputar
Numérica	60.5%	59.7%
Categórica	61.3%	59.4%
Ambas	**61.4%**	59.6%

Tabla 2. Precisión al discretizar en función de los valores usados.

Como podemos ver, no existe demasiada diferencia en los resultados obtenidos, aunque bien es cierto que conseguimos una precisión ligeramente superior si decidimos imputar los valores perdidos siguiendo algún criterio.

5. Valores perdidos.

5.1. Algoritmo de clasificación sobre datos sin imputar e imputados por defecto.

Esta casuística ya se trató en la sección 4.1. En esta, se comenta que la precisión para el conjunto de datos sin imputar es del **59.9%,** mientras que si usamos los datos imputados conseguimos mejorar las predicciones hasta un **60.6%** de aciertos. Recordemos que ha sido necesario imputar las variables *TRAF_WAY* y *NUM_LAN,* las cuales tienen valores perdidos, pero no columna de valores imputados por defecto. Si decidimos obviarlas, de manera que usamos valores imputados salvo en sendas columnas, la precisión es del **60.7%.**

En los tres casos, los dos primeros niveles del árbol son los mismos. Es a partir de la ramificación por debajo de *VEH_INVL >= 2.5* en el tercer nivel (y en general, a partir del quinto nivel) que los árboles comienzan a diferenciarse entre sí.

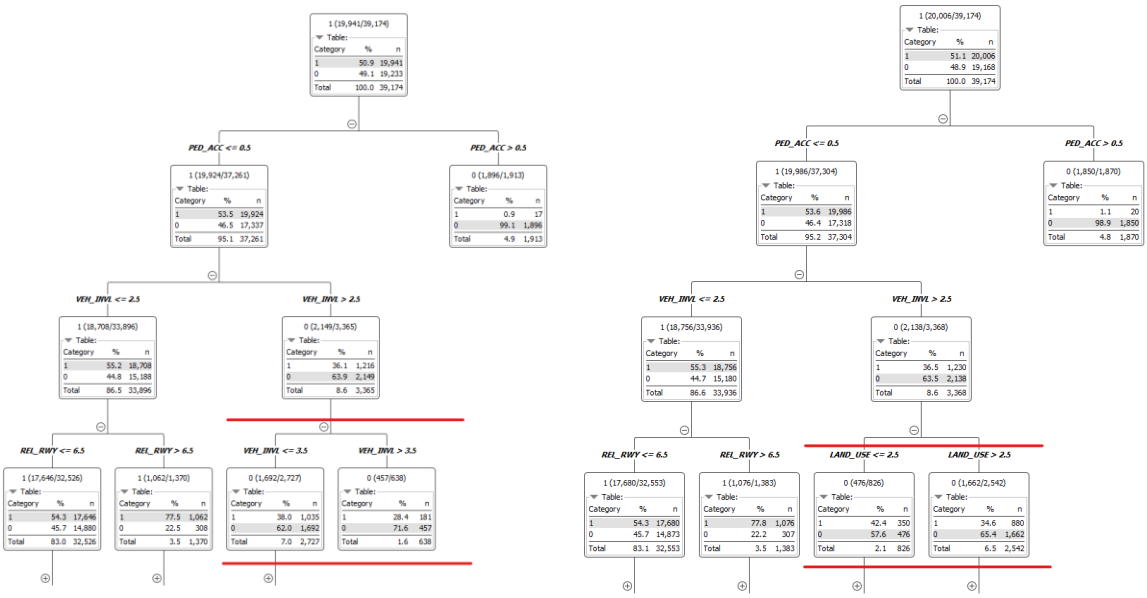


Figura 7. Primeros niveles de los árboles generados al (izq), usar datos sin imputar; (der) usar datos imputados por defecto.

5.2. Resultados al imputar según diferentes criterios.

En este apartado, decidimos rellenar los valores perdidos haciendo uso de diferentes medidas estadísticas, como son la media, la mediana y la moda. Posteriormente, discretizaremos los datos haciendo uso del algoritmo CAIM y del criterio propio propuesto en secciones anteriores, y evaluaremos la precisión obtenida en cada uno de los 3 casos para cada medida estadística utilizada. A modo de resumen, los resultados se muestran en la tabla 3.

Método	Precisión
Imputar con media	56.8%
Imputar con media + CAIM	58.1%
Imputar con media + discretización propia	60.9%
Imputar con mediana	56.6%
Imputar con mediana + CAIM	57.9%
Imputar con mediana + discretización propia	61.0%
Imputar con moda	57.6%
Imputar con moda + CAIM	58.6%
Imputar con moda + discretización propia	**61.3%**

Tabla 3. Resultados obtenidos al imputar y discretizar valores.

Si comparamos con los resultados que hemos ido obteniendo a lo largo del documento, podremos ver que la imputación de valores, y la posterior aplicación de CAIM, nos aporta peores resultados que si usábamos la imputación por defecto que traía la base de datos (cuya precisión era del 59.9%).

Sin embargo, se aprecia una mejora cuando discretizamos los datos imputados haciendo uso del discretizador propuesto en la sección 4.3. Particularmente, los mejores resultados se obtienen imputando los valores haciendo uso de la moda y después aplicando este proceso de discretización, con una precisión del **61.3%.**

Analizando la salida de los nodos durante este apartado me encuentro con un comportamiento curioso. Si aplicamos el algoritmo CAIM y consultamos los datos que salen de dicho nodo, podemos ver cómo cada celda tiene asignada uno u otro intervalo (ver imagen 8). Sin embargo, si aplicamos nuestro algoritmo propio, los datos no parecen asignados a ningún intervalo, independientemente de si los datos de entrada eran enteros o flotantes, es como si no los hubiéramos transformado (ver imagen 9); pero los resultados sí que varían significativamente ya que podemos llegar a obtener hasta un **4% más** de precisión. Desconozco si se trata de un fallo en el visualizador del nodo “Numeric Binner” ya que no he encontrado documentación ni reportes que mencionen este hecho.

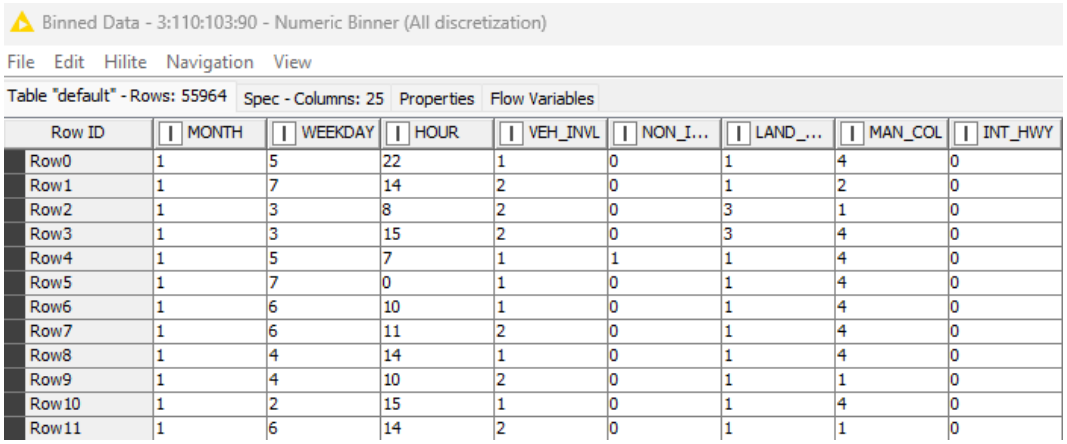
Binned Data - 3:110:103:50 - CAIM Binner

File Edit Hilite Navigation View

Table "default" - Rows: 55964 Spec - Columns: 25 Properties Flow Variables

Row ID	MONTH	WEEKDAY	HOUR	VEH_INVL	NON_I...	LAND_...	MAN_COL	INT_HWY
Row0	Interval_0	Interval_1	Interval_1	Interval_0	Interval_0	Interval_0	Interval_0	Interval_0
Row1	Interval_0	Interval_1	Interval_0	Interval_0	Interval_0	Interval_0	Interval_0	Interval_0
Row2	Interval_0	Interval_1	Interval_0	Interval_0	Interval_0	Interval_1	Interval_0	Interval_0
Row3	Interval_0	Interval_1	Interval_0	Interval_0	Interval_0	Interval_1	Interval_0	Interval_0
Row4	Interval_0	Interval_1	Interval_0	Interval_0	Interval_1	Interval_0	Interval_0	Interval_0
Row5	Interval_0	Interval_1	Interval_0	Interval_0	Interval_0	Interval_0	Interval_0	Interval_0
Row6	Interval_0	Interval_1	Interval_0	Interval_0	Interval_0	Interval_0	Interval_0	Interval_0
Row7	Interval_0	Interval_1	Interval_0	Interval_0	Interval_0	Interval_0	Interval_0	Interval_0
Row8	Interval_0	Interval_1	Interval_0	Interval_0	Interval_0	Interval_0	Interval_0	Interval_0
Row9	Interval_0	Interval_1	Interval_0	Interval_0	Interval_0	Interval_0	Interval_0	Interval_0
Row10	Interval_0	Interval_1	Interval_0	Interval_0	Interval_0	Interval_0	Interval_0	Interval_0
Row11	Interval_0	Interval_1	Interval_0	Interval_0	Interval_0	Interval_0	Interval_0	Interval_0

Figura 8. Usar CAIM tras imputar valores. Los datos pasan a corresponder a intervalos.



Binned Data - 3:110:103:90 - Numeric Binner (All discretization)

File Edit Hilite Navigation View

Table "default" - Rows: 55964 Spec - Columns: 25 Properties Flow Variables

Row ID	MONTH	WEEKDAY	HOUR	VEH_INVL	NON_I...	LAND_...	MAN_COL	INT_HWY
Row0	1	5	22	1	0	1	4	0
Row1	1	7	14	2	0	1	2	0
Row2	1	3	8	2	0	3	1	0
Row3	1	3	15	2	0	3	4	0
Row4	1	5	7	1	1	1	4	0
Row5	1	7	0	1	0	1	4	0
Row6	1	6	10	1	0	1	4	0
Row7	1	6	11	2	0	1	4	0
Row8	1	4	14	1	0	1	4	0
Row9	1	4	10	2	0	1	1	0
Row10	1	2	15	1	0	1	4	0
Row11	1	6	14	2	0	1	1	0

Figura 9. Discretizar según nuestro criterio tras imputar valores. Los datos no parecen sufrir cambios.

5.3. Eliminación de instancias con valores perdidos.

Por recomendación del guion de la práctica hemos probado a eliminar aquellas filas (o instancias) de la base de datos que contengan algún valor perdido. Para ello, hacemos uso una vez más del nodo “Missing Values” y especificamos la opción “Remove row”.

Tras aplicar este proceso, la base de datos pasa a tener 16.362 filas (de las casi 56.000 originales). Si evaluamos el árbol generado haciendo uso de estos datos, obtenemos una precisión del **55.1%,** notablemente inferior a la obtenida en apartados anteriores. Si utilizamos el algoritmo CAIM obtendremos una precisión ligeramente menor, del **54.9%,** mientras que si usamos nuestra propuesta de discretización obtenemos una precisión del **58.2%.**

Parece razonable afirmar que no conviene eliminar las filas, por mucho que tengan algún valor desconocido, pues perdemos una cantidad significativa de información que nos hace generar un modelo peor que los vistos en apartados anteriores, especialmente si usamos el algoritmo CAIM para discretizar.

En la figura 10 se muestran lo 3 primeros niveles del árbol de decisión generado en este apartado si utilizamos nuestro discretizador. Cabe resaltar que dejamos de usar la variable *PED_ACC* en el primer nivel, y *REL_RWY* en los 5 primeros. En su lugar, aparecen variables como *LIGHT_CON* o *ALCOHOL,* las cuales pueden tener más sentido desde el punto de vista intuitivo, pero que e la práctica parecen ofrecer peores resultados.

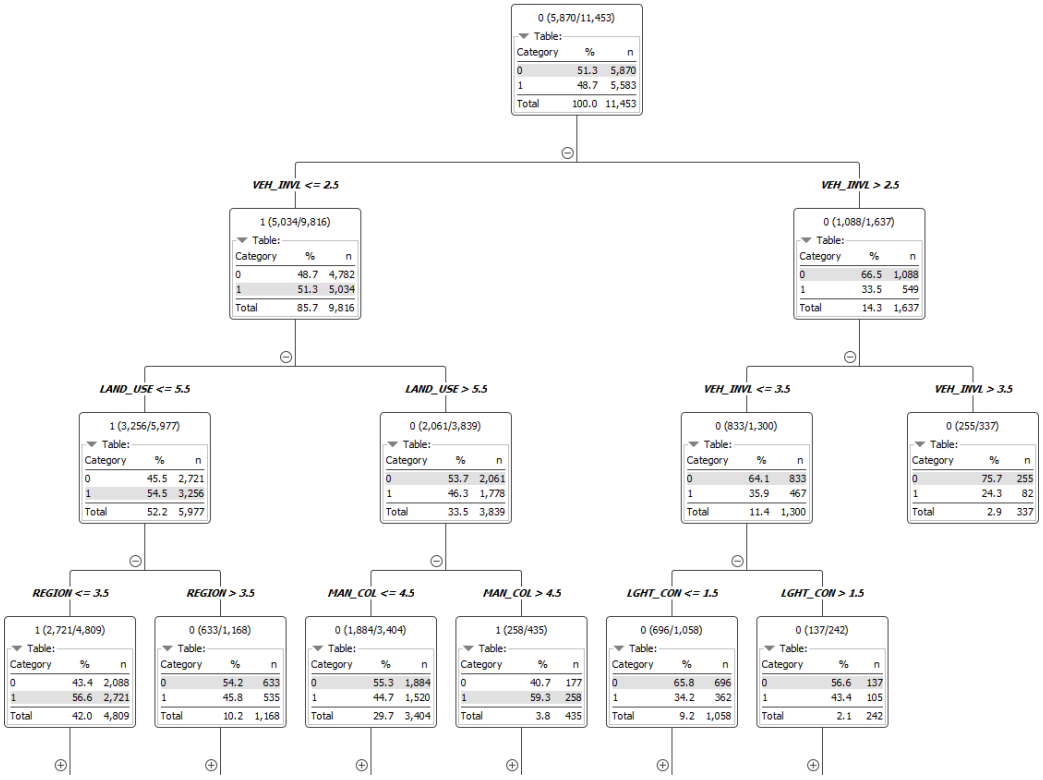


Figura 10. Árbol de clasificación al eliminar instancias con valores perdidos.

5.4. Eliminar características con valores perdidos.

Tras haber probado a eliminar filas con valores nulos, probemos a eliminar columnas con valores nulos. Para llevar a cabo este proceso, hacemos uso del nodo “Missing Value Column Filter”.

Realizaremos dos experimentos. En el primero de ellos probaremos a eliminar todas las columnas que contengan algún valor nulo, aunque solo sea uno. Dado que esta medida parece un poco radical, y podría privarnos de la información de características que tengan una cantidad “prácticamente nula” de valores perdidos, decidimos realizar un segundo experimento en el que solo se eliminen las características que tengan más de un 15% de valores perdidos.

Este umbral lo hemos decidido tras observar la cantidad de valores perdidos de cada columna. Este proceso eliminará las columnas *MAN_COL, TRAF_WAY, SPD_LIM, NUM_LAN* y *PROFILE.*

Si eliminamos todas las características con valores perdidos, obtenemos una precisión del **56.7%, 57.2% y 59.3%** en función de si no discretizamos, si usamos CAIM, o si usamos el criterio para discretizar propuesto. Por otra parte, si eliminamos sólo las características con más de un 15% de valores perdidos obtenemos precisiones del **56.1%, 59.5% y 59.2%.**

En la figura 11, se encuentran los primeros niveles del árbol correspondiente al caso en que se eliminan variables con más de un 15% de valores perdidos y aplicamos nuestro discretizador.

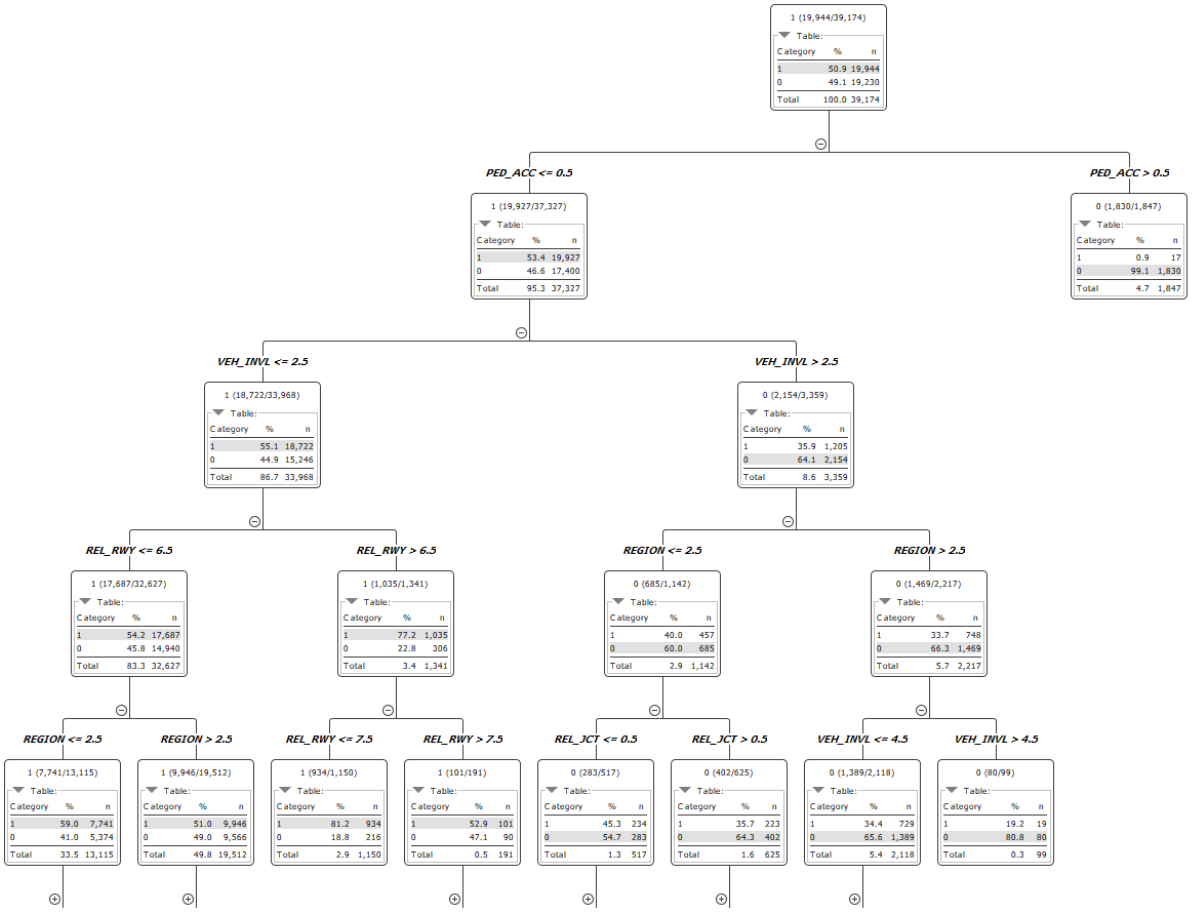


Figura 11. Árbol de clasificación al eliminar características con más de un 15% de valores perdidos.

Si prestamos algo de atención veremos cómo el número de vehículos implicados y *PED_ACC* siguen siendo las variables con mayor importancia, pero en niveles inferiores aparecen características que no habíamos tenido en consideración, como *REL_JCT.*

6. Selección de características.

6.1. Selección de características envolvente hacia atrás.

En esta sección probamos a hacer uso del metanodo “Backward Feature Elimination” con el objetivo de averiguar qué características nos son más útiles en nuestro intento de construir el mejor clasificador posible.

Este metanodo permite definir internamente qué modelo de aprendizaje y predictor usar. Probaremos a hacer uso de Naive-Bayes, el cual es el método que viene especificado por defecto, y árboles de decisión. En el filtro dentro del metanodo, se ha marcado que no se incluyan las columnas estáticas, lo cual reduce el número de atributos a tener en cuenta.

Entrenaremos ambos modelos para los datos imputados por defecto que incluía la base de datos, y datos imputados haciendo uso de la moda, ya que en el apartado 5.2 se vio que parecía ser el método que mejores precisiones proporcionaba.

Los resultados obtenidos se detallan en tabla 4. Téngase en cuenta que la base de datos contiene 24 características únicas.

Método	Precisión	N.º Características
Imputación por defecto + Naive Bayes	56.8%	22
Imputar por defecto + Árbol de decisión	61.9%	12
Imputar con moda + Naive Bayes	58.2%	9
Imputar con moda + Árbol de decisión	**62.7%**	14

Tabla 4. Resultados obtenidos al seleccionar características automáticamente.

Los atributos que utiliza cada método son diferentes, y se invita al lector a que los consulte en el nodo "Feature Selection Filter", pero hay algunos que sí que se mantienen en común. Estos son *VEH_INVL, NON_INVL, LAND_USE, REL_RWY, TRAF_WAY* y *NUM_LAN.*

Los algoritmos consideran que estos son los atributos que más información útil proporcionan para predecir la gravedad del accidente. Analicemos la importancia de cada uno desde nuestro punto de vista.

- *VEH_INVL:* parece razonable pensar que cuantos más vehículos se vean implicados en un accidente más probabilidad hay de que alguien resuelte herido o fallezca.
- *NON_INVL:* el número de no-motoristas implicados es una estadística similar a la anterior. Consideramos que aquí también se incluyen peatones, por lo que tiene sentido su inclusión. Podría atropellarse a alguien y que este salga herido o incluso fallezca sin que tengan lugar daños materiales.
- *LAND_USE:* es una variable que a priori no hubiese considerado, pues no parece haber demasiada relación entre el número de habitantes de una zona y la gravedad de un accidente producido en dicha zona.
- *REL_RWY:* indica el tipo de carretera en que se produce el accidente, por lo que resulta lógico incluirla. Accidentes aparcando no resultaran de la misma gravedad que aquellos que tengan lugar en una autovía, por ejemplo.
- *TRAF_WAY:* indica si la carretera esta divida físicamente o es de un único sentido. Personalmente, tampoco hubiese considerado relevante esta variable.
- *NUM_LAN:* indica el número de carriles que tiene la carretera. Es un apartado que si hubiese considerado pues puede conllevar el impacto con un mayor número de vehículos, lo que aumentaría la gravedad del accidente. Nótese que esta variable y *VEH_INVL* no está correlacionadas (valor del índice 0.25), a pesar de que la intuición podría decirnos que sí.

También cabe destacar que, sólo el modelo Naive-Bayes que hace uso de datos imputados por defecto y el árbol de decisión que hace uso de valores imputados con la moda, usan variables como *SPD_LIM* o *ALCOHOL.* Si hubiéramos tenido que seleccionar características manualmente, estas se encontraría entre las seleccionadas muy seguramente, pues todos asociamos velocidad y alcohol a la gravedad de un accidente.

También va un tanto en contra de la intuición que el modelo que mejor funciona ignore el mes, día de la semana u hora del accidente a la hora de predecir la gravedad de un accidente. Como se mencionó en apartados anteriores, puede resultar intuitivo pensar que accidentes más graves tengan lugar en fines de semana, horas punta, o estaciones del año con gran número de desplazamientos.

Si discretizamos todas las columnas que nos indica el mejor modelo mediante el algoritmo CAIM, o aquellas que podamos haciendo uso del criterio propio expuesto anteriormente, no se observa mejoría en la precisión de las predicciones. Tenemos precisiones del 60% y 61.8% respectivamente.

6.2. Selección de características propia.

Tras comprobar los resultados proporcionados por el modelo generado gracias al metanodo "Backward Feature Elimination" que hace uso del árbol de predicción e imputación por moda, tomé las características de dicho modelo y añadí aquellas que intuitivamente pensé que pueden afectar a la gravedad de un accidente, como *HOUR, WEEKDAY* o *MONTH.*

La precisión obtenida fue del **60.5%**, y si discretizábamos los datos esta precisión se reducía en un 1% aproximadamente. Por tanto, para parte de los experimentos de la siguiente sección se utilizarán las 14 características que nos proponía el mejor modelo de la sección anterior.

7. Selección de instancias.

7.1. Técnicas de submuestreo ("downsampling").

En esta sección comprobaremos si reducir el número de instancias a comprobar mediante una técnica de "downsampling" permite mejorar la legibilidad e incluso precisión del modelo.

Haciendo uso del nodo "Row Sampling", reducimos el número de muestras al 40%, de manera que a partir de ahora tendremos unos 22.400 ejemplos, a los cuales imputaremos los valores perdidos haciendo uso de la moda. Siguiendo la tendencia anterior, el número de casos con diferente valor de la variable de clasificación está más o menos equilibrado.

Si mantenemos únicamente las 14 columnas que nos devolvía el mejor selector de características del apartado 6.2, el predictor generado tiene una predicción en torno al **60.3%**, por lo que no se aprecian cambios significativos con respecto a los resultados que hemos ido obteniendo a lo largo del conjunto de experimentos.

Al reducir el número de instancias, puede que algunas de las características propuestas por el mejor selector del apartado 6.2 dejen de ser tan relevantes como esperábamos, por lo que volvemos a aplicar el metanodo "Backward Feature Elimination" que hace uso de un árbol de decisión. En esta ocasión, se consideran 18 de las 24 características (incluyendo algunas como *WEEKDAY, SUR_COND* o *TRAF_COND,* ignoradas anteriormente) y se obtiene una precisión del **60.8%.** La mejora obtenida no es demasiado significativa.

Sin embargo, el árbol de decisión generado es algo más reducido que los utilizados en secciones anteriores. En esta ocasión, si hemos podido visualizarlo completo sin que el programa KNIME dejara de responder.

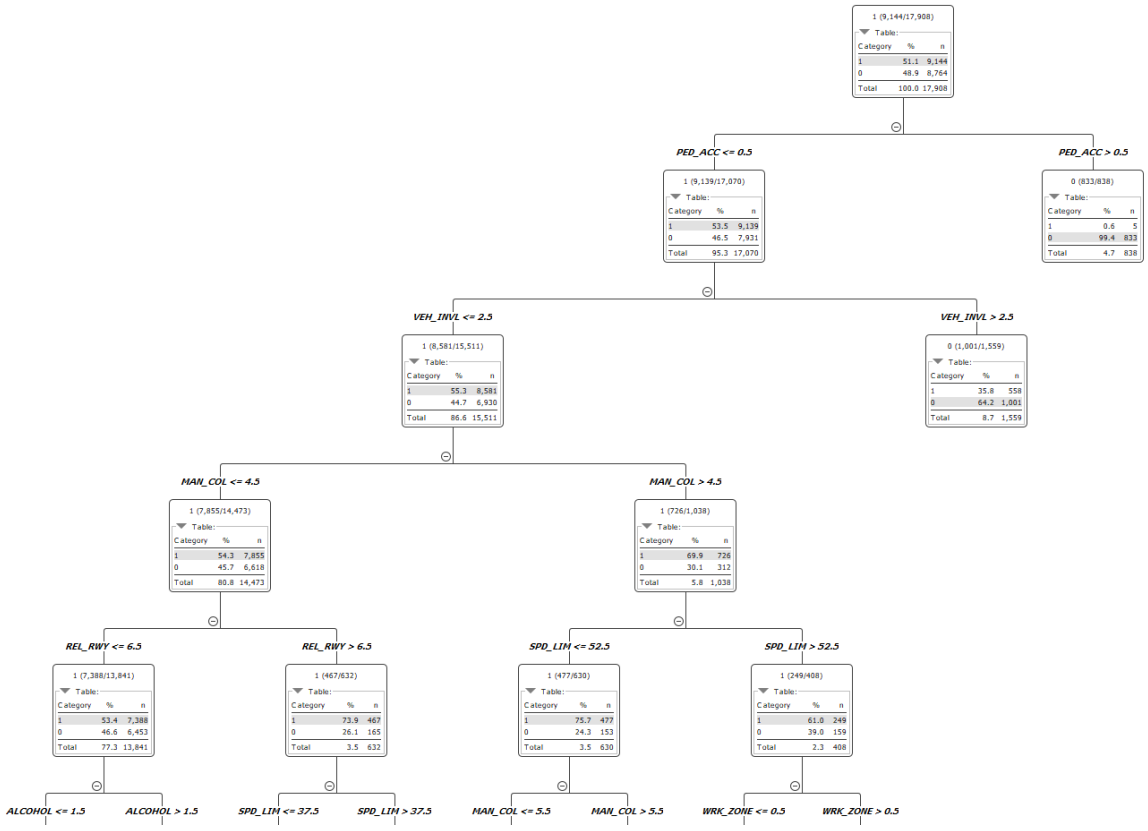


Figura 12. Árbol de clasificación al aplicar submuestreo de filas y "Backward Feature Elimination".

Si observamos los primeros niveles del árbol de la figura 12, vemos como *PED_ACC* y *VEH_INVL* siguen siendo los primeros criterios de división, pero a partir del tercer nivel encontramos otras características que pueden resultar intuitivas, como el tipo de colisión, la velocidad o la presencia o no de alcohol

7.2. Sobremuestreo con SMOTE.

Sobre la reducción de instancias del apartado anterior, probaremos el resultado de realizar un sobremuestreo de la clase minoritaria con el nodo "SMOTE", a pesar de que la variable *GRAVEDAD* está bastante balanceada. Buscamos generar nuevos datos que nos permitan construir un modelo que ofrezca mejores resultados cuando se aplique sobre datos que no se encuentren en la base de datos proporcionada. Téngase en cuenta que, al aumentar el número de datos, algunos métodos como "Backward Feature Elimination" toman un tiempo de ejecución bastante considerable.

Antes de aplicar este último algoritmo, normalizaremos los datos, pues SMOTE hace uso de la técnica de los vecinos más cercano, la cual ofrece un mejor rendimiento con datos normalizados.

Sobremuestreamos los datos para que haya el triple de cada clase, de manera que tendremos unas 90.000 instancias en total. Si probamos a generar el clasificador C4.5 sin realizar más procesamientos, la precisión mejora sustancialmente hasta alcanzar el **78.7%.** Si probamos a usar únicamente las 14 características que consideramos en el apartado 6.2, la precisión se reduce al **74.5%.** Si volvemos a aplicar el metanodo "Backward Feature Elimination" y calculamos que características son realmente relevantes sobre este conjunto de datos sobremuestreado, las cuales pueden haber cambiado con respecto de la base de datos "original", el clasificador resultante nos permite alcanzar una precisión del **78.9%.**

Se ha probado también a aplicar los métodos de discretización propuestos a lo largo del documento, obteniendo resultados entre el 74 y 76% de precisión, por lo que parece que en esta ocasión el proceso de discretización no ayuda a mejorar los resultados.

En el marco de la última batería de experimentos, probamos a aplicar SMOTE sobre una porción mucho menor de la base de datos (6.716 de las casi 56.000 filas). La precisión que nos devuelve el clasificador generado sobre el conjunto de datos sin filtrar características es del **80.3%,** mientras que si utilizamos las características del apartado 6.2 esta decae al **76.2%,** si volvemos a calcular las características relevantes obtendremos una precisión del **79.9%.**

Los primeros niveles del árbol de decisión generado para el caso en que obtenemos un **80.3%** de precisión se muestra en la figura 13. Nótese que los datos están normalizados en el intervalo [0,1]. Al igual que en ocasiones anteriores, en el primer nivel se utiliza *PED_ACC* para decidir; pero en el segundo y tercer nivel aparecen las variables *PROFILE* y *LAND_USE,* las cuales no se tenían en cuenta hasta que no descendíamos bastante en los árboles generados en secciones anteriores. Resulta curioso que este árbol que proporciona mejores resultados obvie el tipo de colisión, el límite de velocidad o la presencia de alcohol.

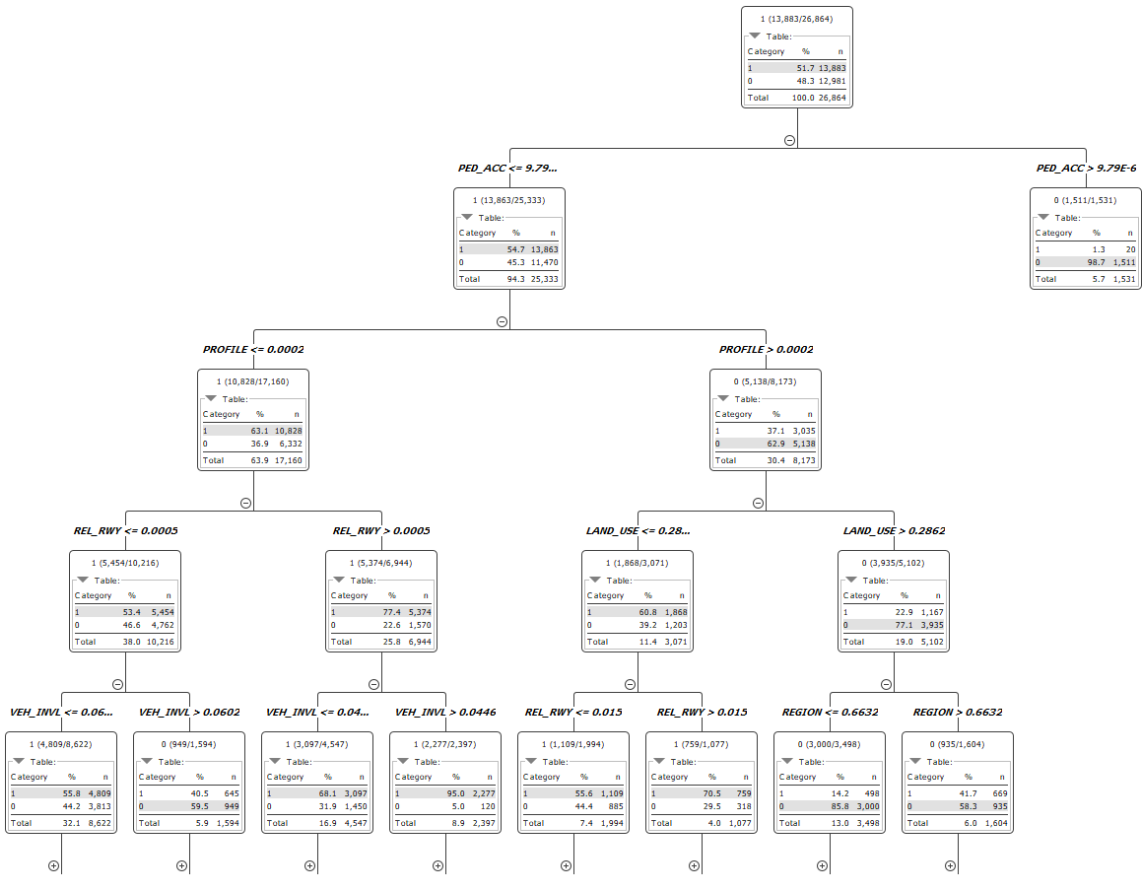


Figura 13. Primeros niveles del mejor árbol de decisión obtenido.

Aunque todo parece indicar que la aplicación de SMOTE sobre una muestra de la base de datos permite generar un predictor más preciso, me muestro escéptico ante los resultados. El proceso de generación de datos seguido en este apartado genera potencialmente "datos sintéticos falsos", que bien podrían no corresponderse del todo con la realidad, de manera que el clasificador generado no generalice bien ante nuevas situaciones del mundo real, pero sí que se ajuste bien al conjunto de datos con los que estamos trabajando, es decir, haya incurrido en sobre-aprendizaje. Sería conveniente tratar con otra base de datos relacionada, pero no identifica, para ver si realmente el modelo generado permite clasificar la gravedad de sus accidentes con un 80% de precisión, o si deberemos conformarnos con otros modelos que nos proporcionen valores de la precisión cercanos al 62%, como ha sucedido a lo largo del documento.

8. Conclusiones.

- Esta práctica nos ha permitido obtener una visión más completa de las técnicas y algoritmos que se utilizan para procesar datos antes de su análisis como tal.
- También, esta práctica ha servido como un ejemplo muy claro de que ciertas bases de datos pueden ser realmente difíciles de limpiar, y muchas veces este proceso

tiene más de arte que de ciencia, en tanto que todos los métodos ofrecen resultados similares y no siempre hay una elección clara.

- Durante la realización de este trabajo, me he dado cuenta de que las variables que uno cree importantes no siempre tienen porqué serlo, y resulta vital dejar de un lado la intuición y prestar atención a aquello que los datos nos quieren decir.
- Resulta importante ser crítico con los resultados obtenidos y considerar si realmente son fieles a la realidad o producto de una "mala manipulación". Especialmente en problemas como el abordado en esta práctica, que podría tener impacto en la gestión de recursos ante emergencias.