Trabajo Fin de Máster

Técnicas de aprendizaje supervisado en la enfermedad de Alzheimer. Aplicaciones al estudio ADNI.

Jorge Pérez Heredia

Supervisores:

José Tomás Alcalá Nalvaiz
Salvador Olmos Gasso

8 de septiembre de 2014

Universidad Zaragoza

Prólogo

En este trabajo hemos empleando técnicas de Minería de Datos sobre la base de datos del proyecto *Alzheimer's Disease Neuroimaging Initiative* (*ADNI*) con el objetivo de encontrar patrones en el seguimiento de los pacientes con la enfermedad de Alzheimer.

El objetivo es realizar un diagnóstico del estado de la enfermedad y una predicción futura sobre si habrá conversión a una etapa más avanzada de la enfermedad. En el primer caso obtuvimos resultados excelentes con precisiones de hasta 90 % y valores del área bajo la curva *ROC (AUC)* superiores a 0,94, en el segundo problema obtuvimos precisiones del 78 % y *AUC* de 0,8 en predicciones a 6-12 meses vista, mientras que para previsiones a más largo plazo la precisión se redujo a unos valores entre el 55 y el 75 % y el *AUC* entre 0,63 y 0,76.

Prologue

In this project we have applied Data Mining techniques over the *Alzheimer's Disease Neuroimaging Initiative* (*ADNI*) database searching for patterns in the evolution of patients with Alzheimer's disease.

The aim of this project was to diagnose the stage of the disease and to predict if there will a conversion to a worse stage. In the first case we reached accuracy levels up to 90 % with areas under the *ROC* curve (*AUC)* of *0.94*, in the second problem the accuracy was 78 % with an *AUC* of *0.8* in predictions within a 6-12 months term, however in a long term prediction the accuracy was reduced to levels between $55 - 75$ % and an *AUC* between *0.63-0.76*.

Índice general

1. Introducción

Uno de los principales problemas en la investigación de la enfermedad de *Alzheimer* [1] es su tardía detección, por lo que conseguir diagnosticarla antes de que empiecen a aparecer los síntomas es uno de los principales objetivos de los científicos.

En este trabajo hemos tenido acceso a la base de datos recopilada por el proyecto *Alzheimer's Disease Neuroimaging Initiative* (*ADNI*) [2], donde se recoge el seguimiento de hasta 1650 pacientes en diferentes etapas de la enfermedad.

Hemos conseguido distinguir entre ancianos normales y dos estadios de la enfermedad con una precisión diagnóstica muy buena, y aunque no hemos logrado obtener un test de diagnóstico competente para las fases tempranas de la enfermedad, sí hemos podido establecer unas nuevas reglas de selección que dependen únicamente de pruebas sin apenas coste económico y no invasivas. Esas reglas pueden servir de ayuda en la toma de decisiones de la práctica clínica rutinaria.

1.1. Enfermedad de Alzheimer

La enfermedad de *Alzheimer* [1] o *Alzheimer Disease (AD)* fue descrita en 1906 por *Alois Alzheimer,* es la forma de demencia más común, en el 2006 afectó al 0.4 % de la población y se prevee que esta cifra se triplique para el año 2050.

Actualmente no tiene cura y únicamente se tratan sus síntomas (con escasa eficacia), es una enfermedad degenerativa que empeora con el tiempo llegando a provocar incluso la muerte. La esperanza de vida después del diagnóstico es de 7 años. Se han propuesto numerosas medidas preventivas (dieta, ejercicio físico, entrenamiento cognitivo,...) pero ninguna ha conseguido demostrar su eficacia.

Aunque algunos síntomas pueden variar según la persona, la mayoría son comunes y normalmente se atribuyen erróneamente a la edad o al estrés. Generalmente afecta a la población mayor de 65 años, pero pueden darse casos en personas más jóvenes.

1.1.1. Fisiopatología y posibles causas

La enfermedad de *Alzheimer* se caracteriza por la pérdida de neuronas y sinapsis en la corteza cerebral y en algunas regiones subcorticales, provocando atrofia en varias partes del cerebro. Estos daños son visibles en las imágenes de resonancia magnética *(MRI)*, al ver una clara disminución en el volumen de algunas estructuras anatómicas del cerebro.

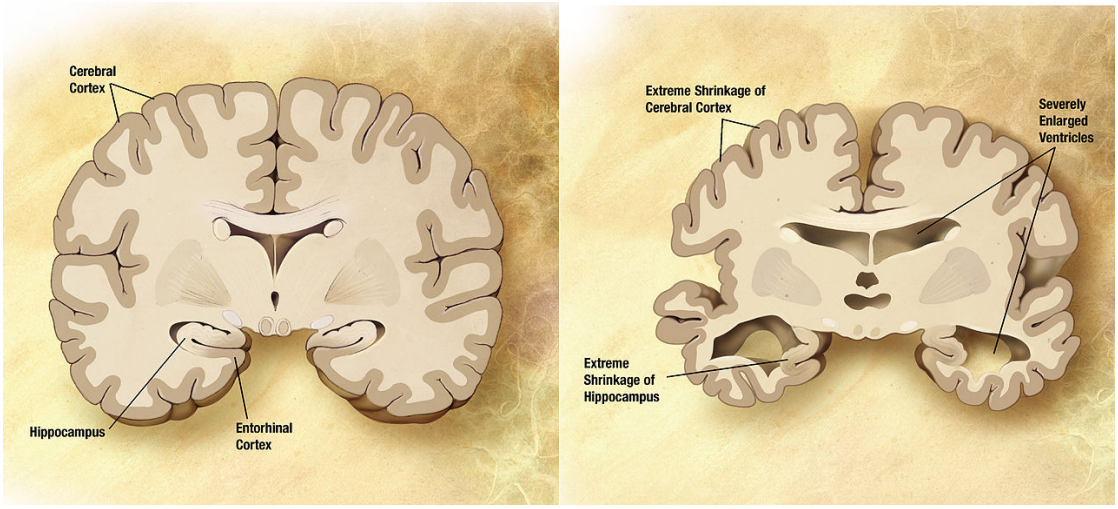


Figura 1.1.1.: Ilustración mostrando la diferencia entre un cerebro sano (izquierda) y uno con Alzheimer (derecha).

A nivel histológico la enfermedad esta asociada a la aparición de unas placas seniles extracelulares y unos ovillos neurofibrilares (Ver figura 1.1.2), estableciendo así tres hipótesis sobre la causa de la enfermedad.

- *Hipótesis amiloide* [3]: Como las placas extracelulares están formadas principalmente por una pequeña proteína llamada *amiloide,* esta hipótesis sostiene que la alteración de las concentraciones de esta proteína son la causa de la enfermedad. Estos cambios pueden ser provocados por mutaciones genéticas, generalmente en los genes *proteína precursora amiloide (*APP*), presenilinas 1 y 2,* o en el gen *TREM2.*

- *Hipótesis amiloide revisada* [4]: Después de que una vacuna experimental consiguió limpiar estas placas pero no tuvo efecto sobre el estado cognitivo de los sujetos, se cambió la hipótesis anterior. Esta hipótesis revisada sostiene que unos oligómeros cercanos a esta proteína conocidos como *ligandos difusibles derivados del amiloide* (*ADDLs*) son los causantes de la enfermedad, al unirse a algunos receptores entorpecen la sinapsis y dañan la comunicación neuronal. Se cree que uno de estos receptores es la misma proteína prión causante de la enfermedad de *Creutzfeldt–Jakob* en el hombre (en los animales encefalopatía espongiforme bovina), estableciendo así un nexo de unión entre dos enfermedades neurodegenerativas.

- *Hipótesis tau* [5]: Según esta hipótesis la proteína *tau hiperfosforilizada* se empareja con otros hilos de la proteína *tau*, y forman eventualmente ovillos neurofibrilares dentro de las neuronas, provocando la desintegración de los microtúbulos y un mal funcionamiento de las comunicaciones neuronales.

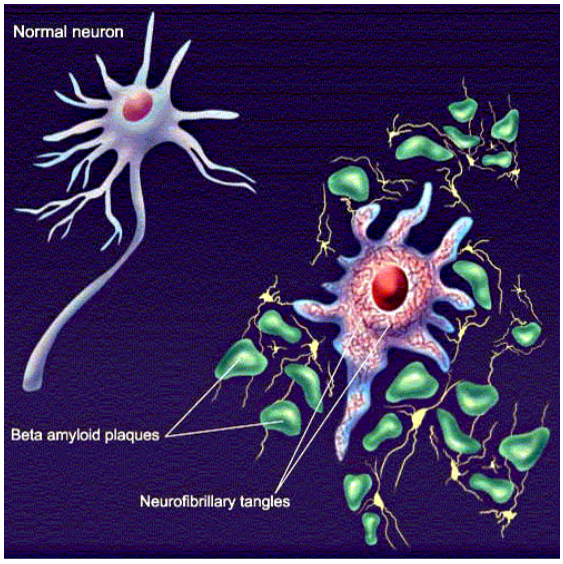


Figura 1.1.2.: Efectos del AD en el cerebro.

Hasta la fecha se han identificado varios factores de riesgo para padecer la enfermedad de Alzheimer. El primero y más importante es la edad, ya que la prevalencia de la enfermedad crece exponencialmente con la edad. El segundo factor de riesgo en importancia es la presencia del alelo $\epsilon 4$ del gen *apolipoproteína E* ($APOE$) [6], ya que entre el 40 y el 80 % de los enfermos de AD tienen al menos uno de estos alelos.

1.1.2. Etapas de la enfermedad

Según el avance de la enfermedad se diferencian tres etapas: temprana, intermedia y tardía. Los síntomas varían en cada etapa y también difieren de la manifestación del envejecimiento natural.

Efectos del envejecimiento en la memoria

- Pérdida de memoria ocasional.
- No recordar ocasionalmente la ubicación de los objetos.
- Débil pérdida de la memoria a corto plazo.
- Olvidarse de haber tenido alguna pérdida de memoria.

Etapa temprana

- Distracciones.
- Olvidarse de citas.

- Familiares cercanos aprecian un débil deterioro.
- Confusión en algunas situaciones fuera del ámbito familiar.

Etapa intermedia

- Dificultad para recordar cosas aprendidas recientemente.
- Mayor confusión en muchas circunstancias.
- Dificultad en el habla.
- Iniciar repetidamente la misma conversación.

Etapa tardía

- Alteraciones de la conducta. Aumento de la agresividad o la pasividad.
- Dependencia en las actividades de la vida diaria y pérdida del autocuidado.
- Déficit cognitivo.
- Aumento de la ansiedad y la paranoia.

1.1.3. Diagnóstico

El *National Institute of Neurological and Communicative Disorders and Stroke (NINCDS)* y la *Alzheimer's Disease and Related Disorders Association (ADRDA*) establecen los criterios de clasificación de la enfermedad de *Alzheimer,* revisados por última vez en el 2007 [7].

Aunque para realizar un diagnóstico definitivo es necesaria una examinación post-morten en el microscopio de una muestra de tejido cerebral, hay numerosas formas de realizar un buen diagnóstico. Los dos tipos de pruebas principales son: los test cognitivos y las imágenes cerebrales. Los test sirven para establecer la existencia de demencia y su severidad, y las imágenes se realizan para cuantificar el volumen de las distintas áreas del cerebro y determinar si hay atrofia y/o daño cerebral. También son importantes otros datos como la historia familiar.

Pero el principal reto es determinar si una persona padecerá *AD* en un futuro sin haber mostrado ningún síntoma actualmente. Estos test deben basarse en las causas y no en los efectos de la enfermedad, siguiendo las *hipótesis amiloide y tau* (Ver subsección *1.1.1)* se ha conseguido elaborar un test diagnóstico con una sensibilidad del 94 % a partir de los valores de la concentración de dichas proteínas en muestras de líquido cefalorraquídeo [8]. Para extraer el líquido cefalorraquídeo se realiza una punción lumbar que conlleva cierto dolor y riesgo para el paciente por posibles efectos adversos. Recientemente se ha conseguido estimar estos niveles a través de una tomografía de emisión de positrones *(PET)* mediante el uso de algunos radioisótopos *(Florbetapir F 18 y otros)*, esta prueba además de dar una medida más directa de la concentración de estas proteínas nos informa sobre su distribución espacial, por contra el sujeto recibe ciertas dosis de radioactividad y tiene un elevado coste económico.

1.2. El proyecto ADNI

Desde el año 2004 el proyecto *Alzheimer's Disease Neuroimaging Initiative (ADNI)* [2] monitoriza el seguimiento de hasta 1650 pacientes, midiendo las variaciones en diversas variables como pruebas cognitivas, neuroimágenes, algunos biomarcadores (niveles de proteínas en sangre, líquido cefalorraquídeo,...) y variables genéticas.

Los principales objetivos del proyecto son:

- Conseguir un diagnóstico temprano del *Alzheimer*.
- Seguimiento de posibles tratamientos o medidas de prevención.
- Contribuir a crear una base de datos de amplio acceso sobre la enfermedad.

En la fase inicial *(ADNI-1)* se reclutaron 800 sujetos de los que 200 eran sanos o normales (*NL)*, 400 tenían deterioro cognitivo leve (*Mild cognitive impairment o MCI)* y 200 pacientes con enfermedad de Alzheimer (*AD).* Los sujetos debían cumplir unos criterios de inclusión:

- Edad entre 55 y 90 años y hablar inglés o español.
- Tener un acompañante que pueda aportar una evaluación independiente.
- Aceptar todos los procedimientos y autorizar el seguimiento longitudinal.
- Entre un 20 y un 50 % deben aceptar punciones lumbares espaciadas al menos un año.
- Cumplir los requisitos de alguno de los tres diagnósticos posibles, principalmente quedan determinados por la puntuación en los test cognitivos (Ver apéndice B):
  - *NL:* Obtener una puntuación entre 24 y 30 en el *MMSE* test, 0 en el *CDR* y ausencia de depresión y demencia.
  - *MCI:* Puntuación de entre 24 y 30 para el *MMSE*, 0.5 en el *CDR,* tener pérdida de memoria pero no daño significativo en otras áreas cognitivas, ausencia de demencia y capacidad para realizar las actividades diarias intacta.
  - *AD:* Puntuación entre 20 y 26 en el test *MMSE*, 0.5 ó 1 en el *CDR* y cumplir los criterios *NINCDS/ADRDA* para declarar alta probabilidad de *AD.*

  Además la distribución de edad en los tres grupos debe ser similar.

En la segunda fase *(ADNI-GO)* se introdujeron 200 nuevos sujetos buscando un perfil de deterioro coginitivo temprano (early *MCI*) y unos 450-500 continuaron desde la fase (*ADNI-1*).

En la actualidad nos encontramos en la tercera fase (*ADNI-2*) que incorporó 650 nuevos sujetos (150 *NL,* 350 *MCI* y 150 *AD)*, además de que 300 (*NL* y *MCI)* de la fase *ADNI-1* continuaron así como los 200 *eMCI* de la fase *ADNI-GO.*

Los criterios de inclusión en las dos últimas fases fueron similares a los de la fase *ADNI-1*. La muestra del estudio no representa a la población, sino que ha sido escogida siguiendo unos criterios similares a los de ensayos clínicos que evalúan la eficacia terapéutica de fármacos.

1.2.1. La base de datos: ADNIMERGE

Además de la elaboración de una base de datos se ha creado la librería *ADNIMERGE* [9] para el lenguaje de programación *R* [10]. Esta librería contiene una cantidad de datos ingente, pero ya ha sido elaborada previamente una tabla que cruza la información importante formando el conjunto de datos *adnimerge* [11].

Podemos agrupar las variables de esta tabla en varios grupos:

- *Metadatos*: Identificador del paciente, del centro de recogida de datos, del subproyecto y fecha del examen o código de la visita.
- *Demográficas:* Edad, sexo, años de educación, etnia, raza y estado civil.
- *Test cognitivos:* Puntúan la capacidad del paciente en diversas áreas cognitivas (aritmética, lenguaje, autocuidado, memoria,...), hay muchos test pero explicamos los principales en el apéndice B.
- *Neuroimágenes: MRI* para estimar los volúmenes de las diferentes estructuras cerebrales y *PET* para determinar el metabolismo de la glucosa *y* los valores de la proteína amiloide.
- *Genéticas:* Número de alelos $\epsilon 4$ del gen *APOE*.

Lamentablemente por diversas cuestiones (económicas, éticas, ...) para algunas de estas variables no hay una muestra significativa. En concreto es una lástima no poder usar los valores de las proteínas *tau* y *amiloide,* ya que hay muy pocas mediciones de estos niveles a través del líquido cefalorraquídeo por cuestiones éticas y la medición a través de la *PET* sólo está en los últimos pacientes debido a la reciente aprobación de está prueba además de su alto coste.

2. Introducción a la Minería de Datos

La Minería de Datos [12] es el proceso de analizar bases de datos para encontrar patrones que ayuden a construir sistemas de decisión. Es una disciplina que se enmarca entre muchas (bases de datos, programación, estadística, inteligencia artificial, aprendizaje automático,...).

Tiene numerosas aplicaciones en varios sectores, algunos ejemplos son:

- *Banca:* evitar fraudes, estimación de riesgos, ...
- *Empresa:* predicción de ventas, preferencias de los clientes, ...
- *Sanidad:* test diagnósticos, identificación de factores de riesgo...
- *Industria:* automatización de procesos, optimización, ...
- *Otras:* control del tráfico, publicidad, ...

Básicamente el objetivo es estimar un comportamiento a partir de un conjunto de datos, si en nuestros datos ya conocemos este comportamiento estaremos en el caso de *aprendizaje supervisado*, y por el contrario si no lo conocemos será *aprendizaje no supervisado*.

Nos centraremos en el primer tipo de técnicas. Un ejemplo famoso de estas técnicas es el conjunto de datos *iris* [13] recogido por R.A. Fisher, el objetivo sería clasificar la especie de una planta a partir de las dimensiones de su sépalo y pétalo.

Altura sépalo	*Anchura sépalo*	*Altura pétalo*	*Anchura pétalo*	*Especie*
5.1	3.5	1.4	0.2	Setosa
4.9	3.0	1.4	0.2	Setosa
4.7	3.2	1.3	0.2	Setosa
7.0	3.2	4.5	1.5	Versicolor
6.4	3.1	4.9	1.5	Versicolor
6.9	2.3	4.0	1.3	Versicolor
6.3	3.3	6.0	2.5	Virginica
5.8	5.8	5.1	1.9	Virginica
7.1	7.1	5.9	2.1	Virginica

Cuadro 2.1.: Pequeña porción del conjunto de datos *iris*.

Vemos en el ejemplo anterior que los datos se muestran en forma de tabla o matriz, donde cada columna es una variable o atributo y las filas son instancias (entradas o

individuos, en nuestro caso pacientes o *s*ujetos). Así pues una formulación más rigurosa de este problema de predicción sería:

Dada una matriz de datos X de dimensiones n x p con n vectores entrada $\mathbf{x} \epsilon R^p$ por filas y la variable objetivo a predecir $\mathbf{y} \epsilon R^n$. El objetivo será determinar una función $f(\mathbf{x})$ tal que dado cualquier $\mathbf{x}$ de X prediga su valor y de la variable objetivo $\mathbf{y}$. La función será aquella que minimice el *error esperado de predicción (EPE):*

$$EPE = E\left[(y - f(\mathbf{x}))^2\right] = \int (y - f(\mathbf{x}))^2 P(dx, dy) = E_x\left[E_{Y|X}\left((y - f(\mathbf{x}))^2 |\mathbf{x}\right)\right] \tag{2.0.1}$$

Es decir, si conocemos $P(y|\mathbf{x})$ sin incertidumbre tenemos el problema resuelto. Si la variable objetivo $\mathbf{y}$ es continua diremos que es un problema de *regresión* y cuando es categórica de *clasificación.*

Cuando se recurre a la Minería de Datos es porque desconocemos con exactitud estas probabilidades y la función $f(\mathbf{x})$, y nos vemos obligados a usar un elenco de funciones propuestas por distintos modelos, esto supone un *"handicap"* de entrada para estas técnicas. Para subsanar este problema necesitamos una *buena muestra de datos* [14] que permitirá mediante diferentes técnicas (mínimos cuadrados, máxima verosimilitud, métodos bayesianos,...) estimar los parámetros de $f(\mathbf{x})$ y por consiguiente estimar $P(y|\mathbf{x})$.

Aquí nace el segundo problema que es el *sobreajuste* [14]. Si usamos todos los datos para entrenar podríamos llegar al caso que ajustáramos hasta el ruido obteniendo un *EPE* muy pequeño, pero al extrapolar nuestro modelo ya ajustado a un nuevo conjunto de datos obtendríamos malos resultados. Una solución para evitar el sobreajuste es dividir el conjunto de datos en dos: *conjunto de entrenamiento y conjunto de validación.* En el primero ajustamos el modelo y en el segundo lo validamos (medir el rendimiento, error, ...). Esto provoca una disminución de la cantidad de datos usados para entrenar, si la pérdida de datos es significativa recurriremos a la *validación cruzada* explicada posteriormente en la subsección 2.2.4.

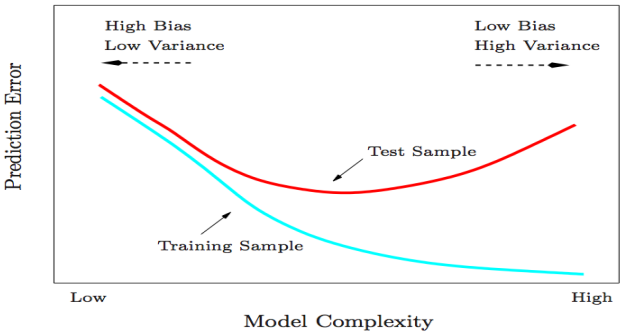


Figura 2.0.1.: *Errores en el conjunto de entrenamiento y de validación* [16]. Representación de cómo minimizar el error en el conjunto de entrenamiento no es equivalente con minimizar el de error en el conjunto de validación. La misma imagen nos sirve para ilustrar el dilema *sesgo-varianza.*

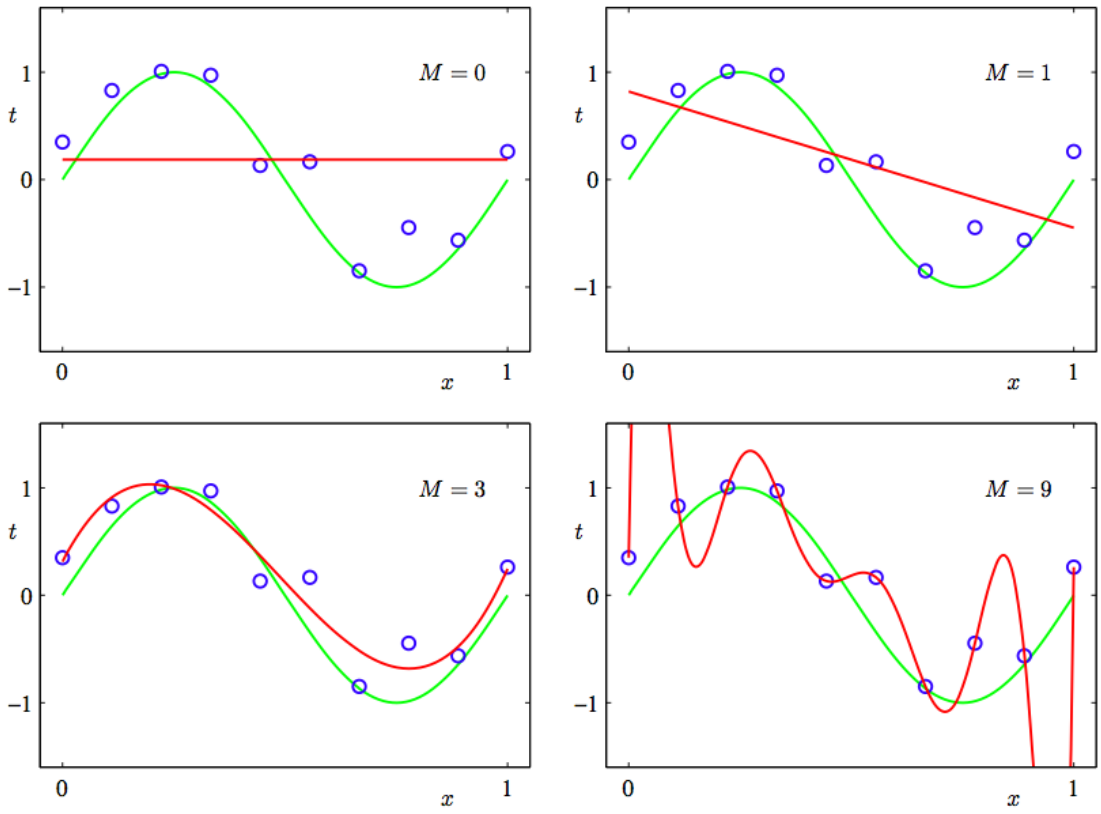


Figura 2.0.2.: *Ejemplo sobreajuste con ajustes polinómicos* [15]. Vemos como el polinomio de grado 9 ajusta a la perfección todos los puntos pero no sigue el comportamiento real de los puntos (línea verde), esto es un caso de sobreajuste. Los casos $M = 1 = 2$ son lo opuesto, el grado del polinomio no es lo suficientemente alto por lo que tendríamos un alto *EPE*. El ajuste adecuado que recoge el comportamiento de los datos sin un elevado *EPE* sería el caso $M = 3$.

Formalmente esto es el dilema *sesgo-varianza*, descomponiendo el *EPE*

$$EPE = E\left[(y - f(\mathbf{x}))^2\right] = E\left[(y - E[f(\mathbf{x})] + E[f(\mathbf{x})] - f(\mathbf{x}))^2\right] =$$

$$E\left[(y - E[f(\mathbf{x})])^2\right] + E\left[(f(\mathbf{x}) - E[f(\mathbf{x})])^2\right] + 2E\left[(y - E[f(\mathbf{x})])\,(f(\mathbf{x}) - E[f(\mathbf{x})])\right]$$

El primer término el el sesgo *(bias)* que disminuye al aumentar la complejidad del modelo, el segundo la *(varianza)* aumenta con la complejidad del modelo, y el último es el término cruzado que se anula. Quedando

$$EPE = B + V \tag{2.0.2}$$

Ya que el objetivo es minimizar el *EPE* y reducir uno de sus términos implica aumentar el otro y viceversa tenemos un dilema.

2.1. Modelos

Hay muchas técnicas de aprendizaje supervisado pero hemos escogido cuatro modelos. En concreto su versión de clasificación y no de regresión, luego podemos simplificar la expresión del *EPE* a

$$Precisión = \frac{1}{n}\sum_{i=1}^{n} \delta_{y_i,\, predicción_i}, \quad \delta_{i,j} = \begin{cases} 1 & si\ i = j \\ 0 & si\ i \neq j \end{cases} \tag{2.1.1}$$
$$EPE = 1 - Precisión$$

2.1.1. K vecinos cercanos

El modelo de *k nearest neighbors (KNN)* [16] en la tarea de clasificación es una aproximación local que consiste en asignar a una entrada $\mathbf{x}$ la salida y más común entre sus k vecinos más próximos. Es un modelo simple con resultados razonablemente buenos que generalmente se usa como modelo inicial a batir.

Primero es necesario que los atributos o variables estén normalizados de alguna forma para que cada variable pese lo mismo a la hora de medir las distancias.

Algorithm 2.1 Pseudo-código del modelo *KNN*.

```
1: Normalizar X
2: Medir distancias entre todos los x_k                ▷ Distancia Euclidea
3: for k = 1 to n do                                   ▷ Pasar por todas las entradas
4:     Encontrar los k vecinos más cercanos
5:     predicción_k =clase predominante
6: end for
```

¿Cómo elegir el parámetro *k*?

Aumentar k hace la predicción menos sensible a datos anómalos pero pierde el carácter de localidad empeorando los resultados. Una opción es escoger el k que minimice el *EPE* (se puede usar otros criterios como el de información de Alkaike *(AIC)* [17] o el de Bayes *(BIC)* [18]).

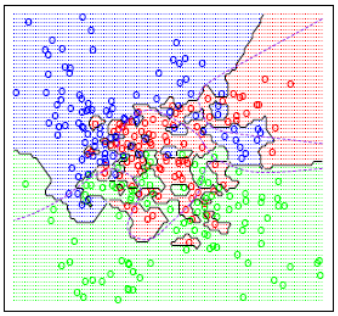
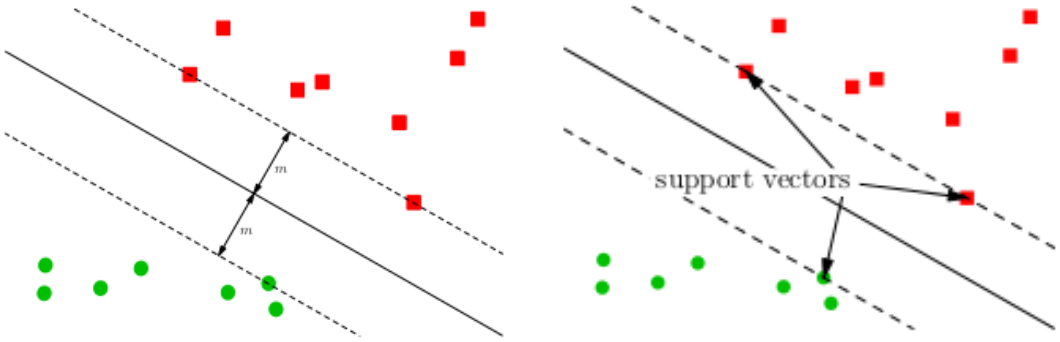


Figura 2.1.1.: Ejemplo de superficie de clasificación usando $K = 1$ [16].

2.1.2. Máquinas de vector soporte

Las support vector machines (*SVM)* [16] se basan en buscar el hiperplano de separación máxima (o menor confusión en su defecto) entre las regiones con diferentes clases de nuestro conjunto de datos.



(a) Hiperplano de separación máxima. (b) Vectores soporte.

Figura 2.1.2.: Ejemplo de máquinas de vector soporte.

El hiperplano queda definido por la expresión

$$y_i \left[\frac{\mathbf{w} \cdot \mathbf{x}_i}{||\mathbf{w}||} - b \right] \geq m, \quad i = 1, 2, ..., n \tag{2.1.2}$$

donde $\mathbf{w}$ es el vector ortogonal del hiperplano, b el término independiente, m el margen de separación entre el plano y las distintas regiones, $\mathbf{x}_i$ una entrada e y_i su valor de la función objetivo.

Si dividimos la expresión (2.1.2) por m y definimos $\mathbf{w}' = \frac{\mathbf{w}}{m||\mathbf{w}||}$ y $b' = {}^{b}/_{m}$ pasamos de tener que maximizar m a minimizar $\mathbf{w}'$ y b' obteniendo un problema cuadrático con solución única

$$min_{\mathbf{w}',b'} \frac{||\mathbf{w}'||^2}{2} \; sujeto \; a \; y_i \left[\mathbf{w}' \cdot \mathbf{x}_i - b' \right] \geq 1 \;\; i = 1, 2, ..., n \tag{2.1.3}$$

Utilizando el método de los multiplicadores de Lagrange tenemos

$$\mathcal{L}(\mathbf{w}', \mathrm{b}', \boldsymbol{\alpha}) = \frac{||\mathbf{w}'||^2}{2} - \sum_{i=1}^{n} \alpha_i \left[y_i \left(\mathbf{w}' \cdot \mathbf{x}_i - b' \right) - 1 \right] \tag{2.1.4}$$

que al realizar las derivadas parciales queda

$$\mathbf{w}_*^{'} = \sum \alpha_i y_i \mathbf{x}_i$$
$$\textstyle\sum_{i=1}^{n} \alpha_i y_i = 0$$

Si en la expresión (2.1.3) sustituimos $\mathbf{w}_*^{'}$ nos queda un nuevo problema cuadrático conocido como el problema *dual* que no depende de $\mathbf{w}$ sino de los productos $\mathbf{x}_i \cdot \mathbf{x}_k$

$$max_\alpha \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,k=1}^{n} \alpha_i \alpha_k y_i y_k \mathbf{x}_i \cdot \mathbf{x}_k \tag{2.1.5}$$

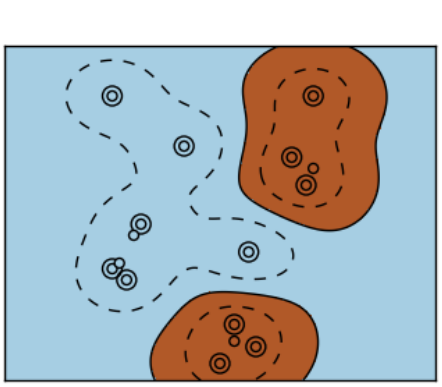
sujeto a $\sum_{i=1}^{n} \alpha_i y_i = 0$ y $\alpha_i \geq 0$, $\forall i$.

Cuando $\alpha_i > 0$ tenemos que $y_i \left[\mathbf{w}' \cdot \mathbf{x}_i - b'\right] = 1$, a estos puntos se les llama *vectores soporte* dando nombre a este modelo (ver figura 2.1.2b).

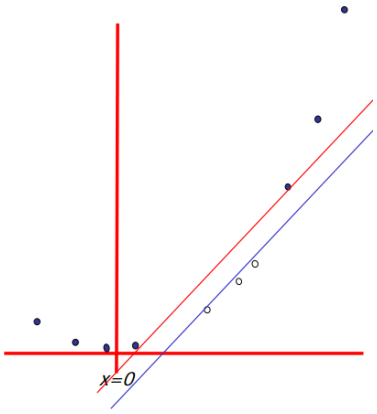
Ahora introducimos el *kernel trick* que consiste en usar por entradas $\phi(\mathbf{x})$ en vez de las $\mathbf{x}$, siendo $\phi(\mathbf{x})$ las funciones propias de una función *kernel* $K(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x})^T \phi(\mathbf{y})$. Trabajaremos pues en un *espacio de variables expandido*, este truco permitirá clasificar datos linealmente no separables (ver figura 2.1.3b). El problema *dual* de la expresión 2.1.5 quedará

$$max_\alpha \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,k=1}^{n} \alpha_i \alpha_k y_i y_k K(\mathbf{x}_i \cdot \mathbf{x}_k) \tag{2.1.6}$$

que nos permite no tener que trabajar con las $\phi(\mathbf{x})$ que aunque son la descomposición del *kernel* suelen ser funciones más complicadas. Se pueden usar numerosos *kernels* (lineal, polinómico, sigmoidal, gaussiano,...) pero hemos escogido el gaussiano $K(x, y) = e^{\gamma(x-y)^2}$ y entrenaremos el parámetro γ para que minimice el *EPE*.



(a) Ejemplo de superficie de separación con un *kernel gaussiano.*



(b) *Ejemplo de cómo las SVM resuelven datos no linealmente separables ampliando la dimensionalidad.* En una dimensión los datos estarían mezclados sobre una recta imposibilitando la separación lineal.

Figura 2.1.3.: *Kernel tricks.*

Por último queda saber cómo asignaremos los valores de la variable objetivo a cada entrada, en el caso de clasificación binaria bastará con

$$signo\left[\sum_{i=1}^{n}\alpha_i y_i K(\mathbf{x}_k\cdot\mathbf{x})-b\right]$$

si la variable objetivo tiene más de dos clases dividiremos el problema en muchos de clasificación binaria.

2.1.3. Árboles de decisión

Para elaborar un modelo de tipo *árbol de decisión* [16], necesitamos definir un *criterio de partición*, un *criterio de parada* y seguir este algoritmo.

Algorithm 2.2 Pseudo-código de un árbol.

```
1: Situar todos los datos en el nodo raíz
2: Buscar partición óptima en dos nodos hijos
3: while Algún nodo no cumpla el criterio de parada do
4:     for k = 1 to #_nodos hijos do          ▷ Recorrer todos los nodos hijos
5:         if Se cumple el criterio de parada then
6:             Siguiente nodo hijo
7:         else
8:             Buscar partición óptima y nueva división
9:         end if
10:    end for
11: end while
```

El nombre de estos modelos viene de que tienen una estructura en forma de árbol. Como vemos en la siguiente figura cada nodo t queda definido por: el número de individuos n_t de los cuales $n_{j,t}$ pertenecen a la clase j de la variable objetivo.

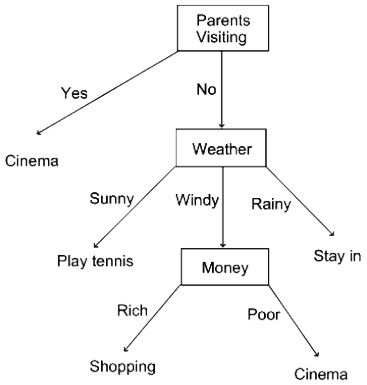


Figura 2.1.4.: Ejemplo de un árbol de decisión.

Hemos escogido el modelo *CART (classification and regression trees)* que escoge las particiones óptimas según la impureza del nodo y no tiene criterio de parada. En ocasiones es interesante renunciar a la partición óptima y optar por una secundaria. La principal ventaja de esto es que cuando para una entrada no haya valor de la partición óptima el modelo buscará la siguiente mejor partición para la que hay datos, esto permite trabajar con cierta cantidad de datos faltantes.

Criterio de partición

Para escoger la partición óptima nos ayudaremos de una magnitud llamada *impureza.* Hay muchas funciones para medir la *impureza* pero usaremos la más extendida que es la de *Gini*

$$i(t) = \sum_{i \neq j} p(j|t)p(i|t) \tag{2.1.7}$$

donde $p(j|t)$ es la probabilidad de tener clase j si estas en el nodo t.

Vemos que a menor *impureza* mejor es la partición así que el criterio será escoger la partición que maximice el decremento de la *impureza*

$$\Delta i(t) = i(t) - \frac{n_{t_r} i(t_r) + n_{t_l} i(t_l)}{n_t}$$

donde t_r es el hijo derecho y t_l el nodo hijo izquierdo.

Criterio de parada

No hay criterio de parada. Se creará un árbol máximo (nodos terminales puros) y después lo podaremos. Para realizar el proceso de poda necesitamos introducir el concepto de *coste de un árbol* $R(T)$, que será la probabilidad de mala clasificación

$$R(T) = \frac{\sum_{t \epsilon T} p(t) r(t)}{r(root)} \cdot 100$$

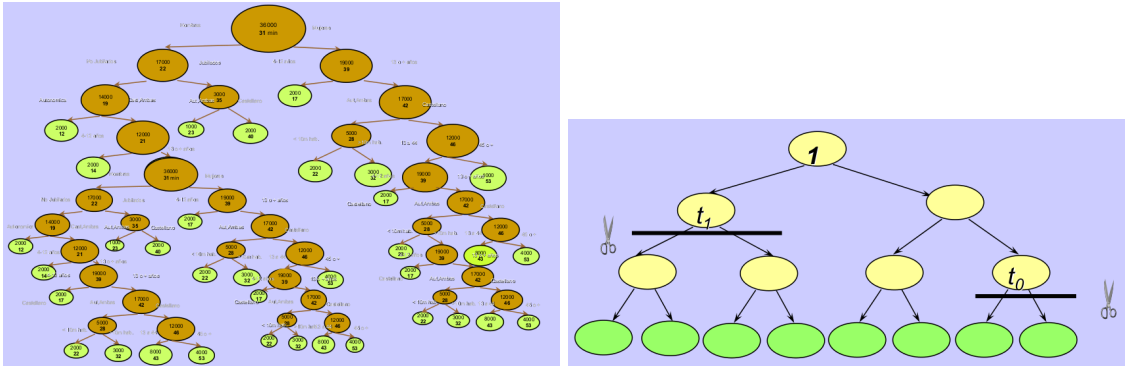
siendo $r(t)$ el coste de un nodo $r(t) = min_i \sum_j c(i|j)p(j|t)$ donde $c(i|j)$ es la penalización o coste de malclasificación y $r(root)$ es una constante de normalización igual al coste del nodo raíz (nodo inicial).

El objetivo será minimizar $R(T) + \alpha|T|$, siendo α un término de complejidad que para cada nodo del árbol máximo valdrá

$$\alpha = \frac{R(t) - R(T_t)}{|T_t| - 1}$$

donde $R(t)$ es el coste del subárbol descendiente del nodo t y $R(T_t)$ el coste del nodo t después de haber podado su subárbol.

Se calculará $R(T_\alpha)$ para varios valores de α y por validación cruzada (Ver subsección 2.2.4), y nos quedaremos con el mayor valor de α que este por debajo del valor $R(T_\alpha) + std.dev(R(T_\alpha))$.



(a) Árbol máximo. (b) Árbol máximo después de la poda.

Figura 2.1.5.: Poda de árboles de decisión.

La principal ventaja de los árboles es que simulan el proceso humano de toma de decisiones. Son computacionalmente muy eficientes pero por el contrario tienden a sobreajustar los datos a la muestra que disponemos.

2.1.4. Bosques aleatorios

Los *random forest (RF)* [16] son una extensión de los árboles de decisión que corrigen el problema del sobreajuste y mejoran notablemente el rendimiento aunque se pierde la interpretabilidad de los resultados.

Este modelo consiste en crear n árboles de decisión pero en cada uno sólo usamos aleatoriamente m de las p variables, a cada entrada se le asignará una predicción y probabilidad según los votos de cada árbol.

Tendremos que elegir los dos parámetros n y m. De nuevo escogeremos los que minimicen el *EPE* (o el *AIC* o *BIC*), pero sólo optimizaremos el parámetro m ya que es el único que afecta sensiblemente el error (ejemplos en las figuras C.2.4 y C.3.4).

Otra ventaja de estos métodos es que previenen el sobreajuste sin tener que recurrir a otras técnicas como ocurre en los demás métodos, ya que en la construcción de cada árbol no se usan todos los datos sino que se realiza un muestreo dejando fuera una cantidad de datos para la validación. Esto se conoce como *out-of-bag error (oob)*.

2.2. Validación de modelos

Tanto para medir el rendimiento de un modelo como para prever el sobreajuste es necesario utilizar unas medidas adecuadas del error.

2.2.1. Error esperado de predicción

Como ya venimos diciendo, el primer método para validar un modelo es medir su *precisión* o *EPE* definidos anteriormente en la expresión (2.1.1).

2.2.2. Tabla de confusión

Otra forma de validar un modelo es una tabla de frecuencias donde cada casilla es el número de casos donde el valor real coincide con el de la predicción del modelo para cada clase de la variable objetivo.

	Setosa	*Versicolor*	*Virginica*
Setosa	45	3	2
Versicolor	0	49	1
Virginica	2	1	47

Cuadro 2.2.: Ejemplo de tabla de confusión para el conjunto *iris* [13]. En este ejemplo el modelo asigna correctamente 45 casos de *Setosa* pero confunde 2 *Virginica* asignándoles también *Setosa*.

2.2.3. Curva ROC

La *Receiver Operating Characteristic* o curva *ROC* [19] es la forma más extendida de medir el rendimiento de un clasificador binario, ya que no sólo tiene en cuenta el valor de la predicción sino la probabilidad con la que asignamos dicho valor. Un punto de la curva *ROC* se construye con el ratio de verdaderos positivos (TPR) y de falsos positivos (FPR)

$$TPR = \frac{TP}{P} = \frac{TP}{TP+FN}$$
$$FPR = \frac{FP}{N} = \frac{FP}{FP+TN}$$

donde TP es el número de verdaderos positivos, es decir, el número de veces que el clasificador predice un positivo y es correcto, y FP es el número de veces que se predice un resultado positivo siendo en realidad uno negativo.

En realidad un clasificador no nos da una predicción sino la probabilidad de que una entrada sea un positivo, y cuando esta probabilidad es $\geq 0{,}5$ (generalmente) le asignamos una predicción positiva. Si en vez de fijar el umbral de decisión en 0,5 lo variamos en el intervalo $[0, 1]$ y medimos el TRP y el FPR obtendremos una curva *ROC*.

Algorithm 2.3 Pseudo-código para obtener una curva *ROC*.

```
for umbral ϵ [0,1] do          ▷ Recorrer unos cuantos valores posibles del umbral
    for i = 1 to n do                                  ▷ Recorrer todos las entradas
        if prob_i > umbral then
            pred_i = possitive
        else
            pred_i = negative
        end if
    end for
    Medir el TPR y el FPR
end for
Representar gráficamente todos los TPR frente a los FPR
```

Un clasificador perfecto sería el que para cualquier valor del umbral $TPR = 1$ formando una línea horizontal en 1, mientras que un clasificador aleatorio sería la bisectriz. Como no se puede hacer una clasificación peor que la aleatoria esto establece un rango de posibles rendimientos motivando así la creación de la magnitud *área bajo la curva ROC (AUC)* que será un número en el rango [0,5, 1].

El AUC es la posibilidad de que un clasificador asigne una puntuación mayor a una entrada positiva que a una negativa (ambas escogidas aleatoriamente).

$$P(y_+ > y_-) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} p(y_+|+)p(y_-|-)[y_+ > y_-]dy_+dy_- =$$

$$\int_{-\infty}^{\infty} p(y_-|-)\int_{y_-}^{\infty} p(y_+|+)dy_+dy_- = \int_{-\infty}^{\infty} p(y_-|-)TPR(y_-)dy_- =$$

$$\int_{-\infty}^{\infty} \frac{dFPR(y_-)}{dy_-}TPR(y_-)dy_- = \int_0^1 TPR(FPR)dFPR = AUC$$

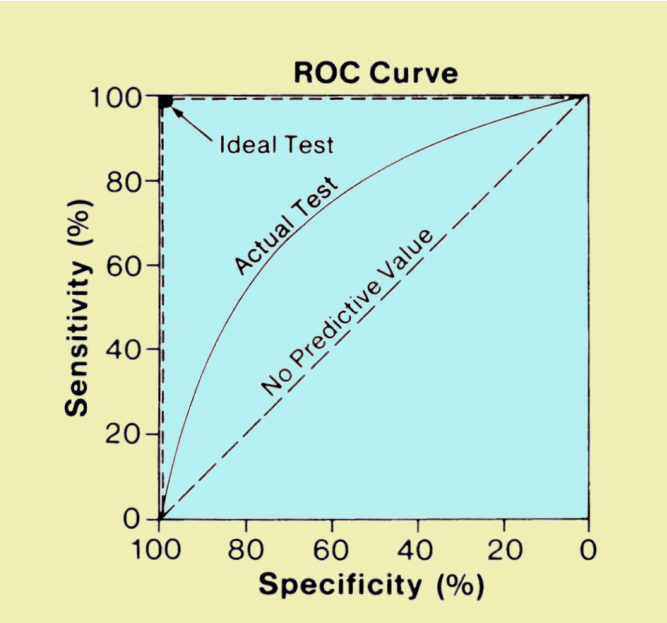


Figura 2.2.1.: *Ejemplo curvas ROC.* Los ejes y e x son equivalentes a TPR y FPR respectivamente, aparecen varios clasificadores como un clasificador aleatorio (bisectriz), un clasificador intermedio (línea continua) y el mejor clasificador (borde izquierdo y superior).

Por último podemos extender estas curvas y el AUC a clasificadores multiclase dividiendo el problema en tantos clasificadores binarios como clases tenga la variables objetivo.

2.2.4. Validación cruzada

Como se explico anteriormente, para evitar el sobreajuste se pueden dividir los datos en dos conjuntos: *entrenamiento* y *validación*. En el primero entrenamos los parámetros de los modelos y en el segundo realizábamos la predicción y mediamos el *EPE*, de esta forma tenemos una medida del error robusta frente a sobreestimaciones.

La *cross validation (CV)* nos permite realizar este proceso pudiendo usar el conjunto de validación en el entrenamiento. Para ello dividimos los datos en k partes aleatoriamente, usamos $k-1$ para entrenar y predecimos en la restante, repetimos el proceso para que todas las partes pasen por el conjunto de entrenamiento, juntamos todas las predicciones y medimos el *EPE*.

La mejor elección sería que k sea igual al número de entradas, esto es computacionalmente costoso y normalmente se escoge $k = 10$. Hay casos en los que hay que buscar un valor adecuado de k (por ejemplo si hay pocos datos), ya que necesitamos que cada partición tenga una proporción similar de entradas de cada clase de la variable objetivo.

Algorithm 2.4 Pseudo-código para validación cruzada.

```
1: Dividir los datos X aleatoriamente en k partes
2: for i = 1 to k do                          ▷ Recorrer todos los grupos
3:     Entrenar el modelo usando todos los datos menos la parte i
4:     Validar el modelo en la partición i
5: end for
6: Juntar todas las validaciones
7: Medir el EPE
```

2.3. Preprocesamiento y selección de variables

Antes de aplicar un modelo a nuestros datos tenemos que verificar que están correctamente rellenados. Normalmente no es el caso y suele haber numerosos inconvenientes (errores tipográficos, datos faltantes, variables con errores muy altos, ...) por lo que es necesario realizar un tratamiento previo llamado *preprocesamiento* que generalmente es la parte más costosa de todo el proceso.

Pasamos a explicar algunas de las fases del preprocesamiento.

2.3.1. Tratamiento datos faltantes

Es un inconveniente muy común, en nuestro caso hay veces que por criterios éticos no se realiza una prueba a un paciente, o en un centro no tienen el equipamiento necesario o el propio protocolo es el que establece cuándo se mide cada prueba dejando huecos.

Algunos modelos pueden trabajar con un número razonable de datos faltantes pero una buena opción siempre es conseguir un conjunto de datos completos. Esto no quiere decir necesariamente quedarnos con el subconjunto de datos donde no falte ninguno, sino que hay diversas técnicas para subsanar o imputar los lugares donde faltan datos. Esta imputación de datos se realiza por columnas (variables).

Usamos las siguientes técnicas:

- *Interpolación:* Al tener los datos evolución temporal, interpolamos localmente dentro de cada paciente (y siempre que hubiera suficientes datos) algunos valores ausentes.
- *Modelización:* Como hemos expuesto en la subsección *2.1* los modelos nos devuelven las probabilidades de que una entrada tenga un valor o clase. Aplicamos el modelo *RF* (Ver subsección *2.1.4*) siendo cada variable con datos faltantes la variable objetivo. El problema es que suele ocurrir que si hay un dato faltante para una entrada y variable lo haya también para la misma entrada y otras variables, en caso de no reunir un mínimo de variables completas no utilizaremos este método.
- *Reemplazo:* Por último, otro método es reemplazar los valores ausentes por la *mediana* de modo que estos datos afecten lo mínimo a los modelos pero no les hagan fallar.

2.3.2. Localización de datos anómalos

Los datos anómalos pueden afectar sensiblemente a los modelos provocando una disminución del rendimiento, por ello es bueno identificarlos y eliminarlos. Hay numerosas técnicas pero nosotros usaremos el *cluster based noise removal algorithm [20]* que usa el algoritmo *k-means cluster* [21] para agrupar la población en diversos grupos o clusters según la distancia entre entradas y elimina los individuos más lejanos.

Algorithm 2.5 Pseudo-código del algoritmo cluster para la detección de datos anómalos.

1: Aplicar a X el algoritmo k-means ▷ Tantos clusters como clases tenga y
2: **for** $i = 1$ to n **do** ▷ Recorrer todos las entradas
3: Medir distancia entre el sujeto i y los centroides generados por el paso 1
4: Guardar la distancia menor en d
5: **end for**
6: **for** $i = 1$ to n **do** ▷ Recorrer todos las entradas
7: Calcular la media μ_d de las distancias
8: Calcular la desviación σ_d de las distancias
9: **if** $d_i > \mu_d + n \cdot \sigma_d$ **then** ▷ El parámetro n lo elige el usuario
10: El individuo i es considerado anómalo
11: **end if**
12: **end for**

2.3.3. Selección de atributos

Otra buena práctica antes de empezar la modelización es eliminar las variables menos importantes, puede parecer que esto no es necesario, pero además del beneficio obvio de reducir el tiempo de computación algunos algoritmos fracasan para dimensiones altas, esto se conoce como el *desastre de la dimensionalidad.*

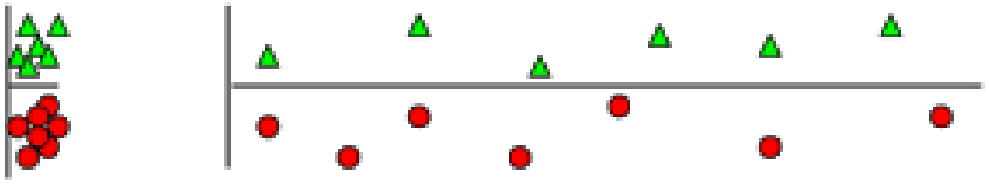


Figura 2.3.1.: *Ejemplo del desastre de la dimensionalidad.* A la izquierda tenemos un conjunto de variables apto para el algoritmo *KNN*, sin embargo a la derecha se muestran los mismo datos pero con una variable irrelevante en el eje x que los expande en una dirección burlando así al algoritmo *KNN*.

Hay numerosos métodos para la seleccionar pero hemos escogido un *filtro de consistencia* [22], que se basa en escoger las variables más consistentes. Una variable es consistente si a todas las entradas con el mismo valor les corresponde la misma clase de la variable objetivo, este concepto se puede extender a conjuntos de variables. El primer problema es que si las variables son continuas es improbable que dos entradas tengan el mismo valor por lo que hay que discretizarlas, y el segundo problema es que raramente se encuentra una variable completamente consistente por lo que mediremos nos quedaremos con la cantidad de entradas consistentes en cada variable.

Algorithm 2.6 Pseudo-código de un filtro de consistencia.

Discretizar las variables de X
while Criterio de parada no satisfecho **do** ▷ Ej.: Alcanzar valor de consistencia
    Escoger un subconjunto de variables $X_{filtered}$ de X ▷ Regla heurística
    Agrupar las entradas que tengan los mismos valores
    for $i = 1$ **to** $\#_{grupos}$ **do** ▷ Medir la consistencia del subconjunto
        $Y =$ valor más común de y en el grupo
        $cons_i = \sum_j^{\#_{entradas}} \delta_{y_j,Y}$ ▷ Número de veces que la variable objetivo $= Y$
    end for
    $consistencia_{X_{filtered}} = \sum_i^{n_{grupos}} cons_i$
end while
$X = X_{filtered}$

2.3.4. Estandarización de variables continuas

El último paso del preprocesamiento es normalizar o estandarizar los datos para que tengan escalas similares y una variable no se imponga a otras sólo por su magnitud. Normalmente se suele tipificar, es decir, restar la media y dividir por la desviación típica. Como esto distorsiona las variables y en nuestro caso al ser además todas positivas hemos preferido normalizar en el rango $[0, 1]$ para que conserven el comportamiento pero tengan todas la misma escala.

2.3.5. Separación de variables categóricas

Cada una de las q variables categóricas o nominales sólo pueden tomar un cierto número de valores n_i, $i = 1, 2, ..., q$, una buena práctica es separar cada una en $n_i - 1$ variables binarias (no se separa en n_i variables para evitar combinaciones lineales).

Individuo	*Estado civil*
1	Casado
2	Soltero
3	Casado
4	Viudo
5	Soltero
6	Casado
7	Divorciado

$\longleftrightarrow$

Individuo	*Casado*	*Soltero*	*Viudo*	*Divorciado*
1	1	0	0	0
2	0	1	0	0
3	1	0	0	0
4	0	0	1	0
5	0	1	0	0
6	1	0	0	0
7	0	0	0	1

Figura 2.3.2.: *Ejemplo separación de una variable categórica.* Faltaría eliminar la redundancia eliminando una columna cualquiera al ser combinación lineal de las demás.

3. Modelización y resultados

En esta sección comentaremos la implementación y los resultados obtenidos al aplicar las técnicas explicadas en la sección anterior. En todo momento nos servimos del lenguaje de programación R [10].

3.1. Preprocesamiento

Como hemos comentado anteriormente el primer paso es realizar un preprocesamiento para que los datos sean aptos para la modelización.

Los datos originales tienen unas dimensiones de 22889 filas y 85 columnas, comenzamos filtrando los datos para obtener sólo pacientes sanos o con demencia debida a la enfermedad de Alzheimer, comprobamos que no haya duplicidades, datos disparatados y eliminamos variables redundantes o sin importancia para nuestro problema. Tras este proceso los datos se han reducido a 2832 entradas y 43 variables.

El siguiente paso es subsanar los datos faltantes. Sólo aplicaremos las técnicas descritas en la subsección 2.3.1 a las variables que no tengan más de 1/3 de datos faltantes y el resto las eliminaremos (nos quedan 25 variables). En primer lugar y sólo si un paciente tiene suficientes entradas usaremos la interpolación, a continuación y solamente para las entradas con suficientes variables completas emplearemos el modelo RF (para clasificación y regresión), y finalmente si han fallado los dos métodos anteriores realizaremos un reemplazo por la mediana.

En tercer lugar buscaremos y eliminaremos datos anómalos usando el algoritmo de clusters (Algoritmo 2.5) con parámetro $n = 2$, sólo que extenderemos el algoritmo y en vez de aplicarse a entradas sueltas se aplicará a pacientes (cada uno contiene una o más entradas). Se catalogaron 18 pacientes como datos anómalos y fueron eliminados.

Finalmente separamos las variables categóricas en binarias obteniendo unas dimensiones de 2708 x 37 que se corresponde con 660 pacientes.

Este proceso forma el preprocesamiento general de la base de datos, posteriormente y antes de enfrentarnos a cada problema se hará otro pequeño tratamiento donde entre otras cosas normalizaremos en el intervalo $[0, 1]$ de las variables numéricas.

3.2. Conjunto de datos

Antes de analizar y modelizar vamos a describir brevemente cada una de las variables de nuestro conjunto de datos.

- *RID:* Código de identificación del paciente.

- *VISCODE*: Código de visita (basal, mes 3, mes 6, mes 12,...).
- *PTGENDER:* Género.
- *PTETHCAT:* Etnia.
- *PTRACCAT:* Raza.
- *PTMARRY:* Estado civil al inicio del estudio.
- *APOE4:* Número de alelos $\epsilon 4$ del gen *APOE*.
- *DX:* Diagnóstico, puede tomar los valores: sano o normal (*NL)*, demencia leve (*MCI)* y Alzheimer (*AD*).
- *AGE:* Edad.
- *PTEDUCAT:* Años de educación.
- *CDRSB:* Puntuación en el test cognitivo *CDRSB.*
- *ADAS11:* Puntuación en el test cognitivo *ADAS-Cog* versión 11.
- *ADAS13:* Puntuación en el test cognitivo *ADAS-Cog* versión 13.
- *MMSE:* Puntuación en el test cognitivo *MMSE.*
- *RAVLT:* Puntuación en el test cognitivo *Rey Auditory Verbal Learning Test.*
- *RAVLT.immediate:* Puntuación en el test cognitivo *Rey Auditory Verbal Learning Test immediate forgetting.*
- *FAQ:* Puntuación en el test cognitivo *Functional Assessment Questionnaire.*
- *Ventricles:* Volumen ventricular (mm3).
- *Hippocampus:* Volumen del hipocampo (mm3).
- *WholeBrain:* Volumen del cerebro (mm3).
- *Entorhinal:* Volumen de la corteza entorrinal (mm3).
- *Fusiform:* Volumen del giro fusiforme (mm3).
- *MidTemp:* Volumen del lóbulo temporal medio (mm3).
- *ICV:* Volumen intracraneal (mm3).

Además cada variable no demográfica se divide en dos: una el valor de la visita actual del paciente y otra el valor de la visita inicial (*VISCODE*=baseline).

3.3. Análisis estadístico

Ya tenemos los datos listos para ser modelizados pero conviene realizar un análisis estadístico previo para entender mejor las variables y su comportamiento. Para ello usaremos los datos preprocesados pero sin el último paso (sin dividir las variables nominales y sin normalizar).

3.3.1. Distribuciones de las variables

En primer lugar realizamos unos gráficos de densidad y de barras (Ver apéndice C.1) para ver la distribución de cada variable según su valor de la función objetivo.

3.3.2. Multicorrelación

A continuación buscamos correlación lineal entre las variables continuas (las variables nominales no mostraron correlación entre ellas). Realizamos un grafo donde las líneas indican una correlación lineal entre las variables que unen y vemos que se forman dos grupos de variables correladas: puntuaciones en los test cognitivos y variables de neuroimagen.

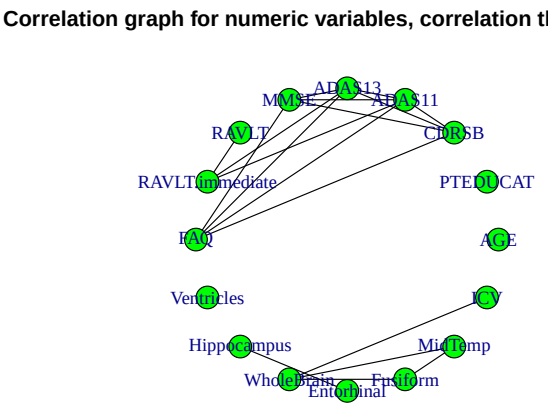
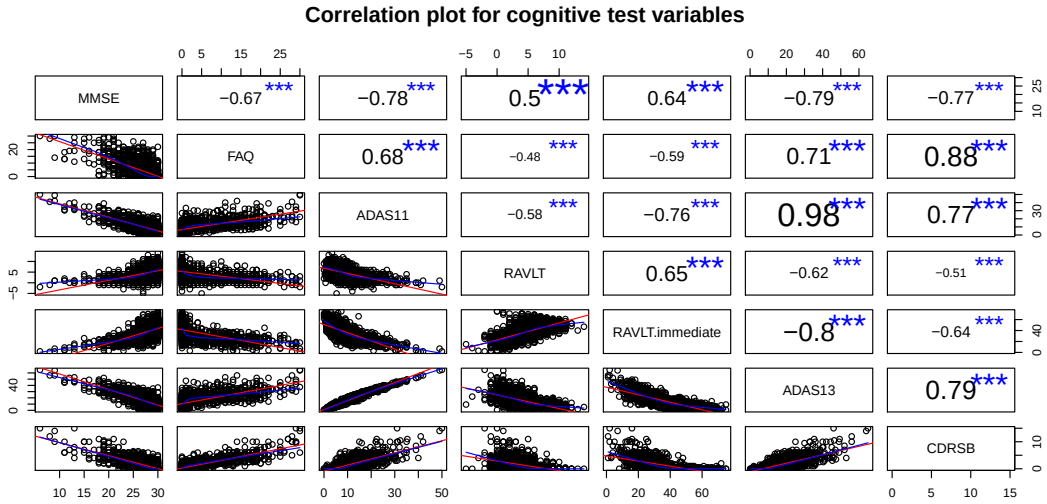
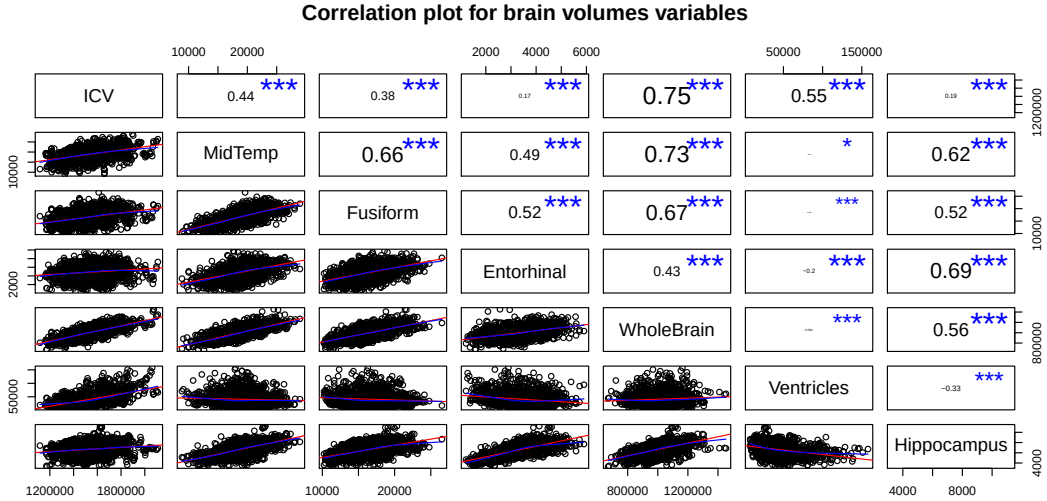


Figura 3.3.1.: *Grafo de correlación.* Realizamos el test de correlación de Pearson [23] entre cada variable y si el valor absoluto del test es mayor que 0,65 pintamos una línea en el grafo.

Pasamos a realizar unos gráficos de correlación entre los dos grupos de variables anteriores y vemos que ambos grupos están muy correlados internamente.



(a) Test cognitivos.

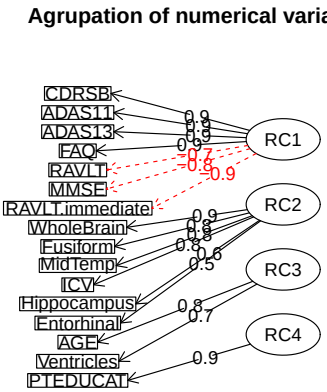


(b) Neuroimágenes.

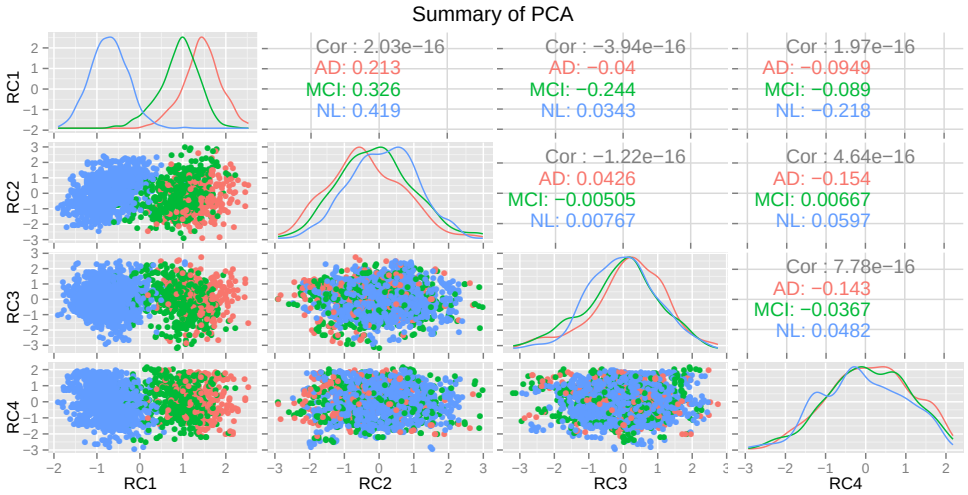
Figura 3.3.2.: *Gráficos de correlación.* Las celdas inferiores a la diagonal muestran la representación gráfica entre variables con un ajuste lineal (línea roja) y uno polinómico (línea azul). La parte superior a la diagonal muestra el valor del coeficiente de correlación lineal y el nivel de significación del test $(*, **, ***) = (0{,}05, 0{,}03, 0{,}01)$.

3.3.3. Análisis de componentes principales

Por último realizamos un análisis de componentes principales o *PCA* [24] para encontrar agrupaciones entre variables que puedan simplificar el estudio.



(a) *Agrupación de las variables numéricas en 4 componentes principales.* Vemos que en general los test cognitivos se agrupan en la primera componente, las neuroimágenes en la segunda y otras variables en la tercera y cuarta componente.



(b) *Representación de las poblaciones para cada valor de la variable objetivo según las coordenadas principales.* Vemos cómo al combinar la primera componente (test cognitivos) con cualquiera de las demás (en especial la segunda) quedan las tres poblaciones separadas y casi linealmente facilitando una clasificación (aunque las poblaciones *AD* y *MCI* quedan con una frontera difusa).

Figura 3.3.3.: Análisis de componente principales.

3.4. Diagnóstico

Nuestro primer problema es realizar un clasificador de ayuda al diagnóstico con el menor número de variables posibles y sin usar variables con interpretación humana. La variable objetivo será el diagnóstico *DX*.

En nuestro caso cada paciente tiene varias entradas (una para cada valor de *VISCODE*), esto los modelos no lo entienden y tratan cada entrada como un individuo independiente. Al haber muchos pacientes estables (diagnóstico constante en el estudio) se subestimaría el error del modelo ya que sería como repetir algunas entradas, por ello en este problema hemos escogido sólo la visita del mes 12 al ser la más equilibrada en las proporciones de pacientes con cada diagnóstico (525 pacientes).

Utilizamos un filtro de consistencia (Algoritmo 2.6) obteniendo como subconjunto de 13 variables: *PTGENDER(Male), PTMARRY(Married), PTMARRY(Divorced), APOE4(0), CDRSB, ADAS11, ADAS13, MMSE, RAVLT, RAVLT.immediate, FAQ, Fusiform, MidTemp.*

A continuación describimos la implementación de los modelos descritos en la subsección 2.1 y su validación por validación cruzada con $K = 10$.

3.4.1. Vecinos cercanos

En primer lugar aplicamos el modelo *KNN,* optimizamos el parámetro K obteniendo $K = 3$ (ver apéndice C.2), es decir, asignaremos a cada entrada una probabilidad y clasifiación según sus 3 vecinos más cercanos.

Validamos el modelo representando su curva *ROC* y obtenemos excelentes valores del *área bajo la curva.* El diagnóstico peor clasificado es el de *MCI* esto es lógico debido a que es el caso intermedio entre *AD* y *NL.*

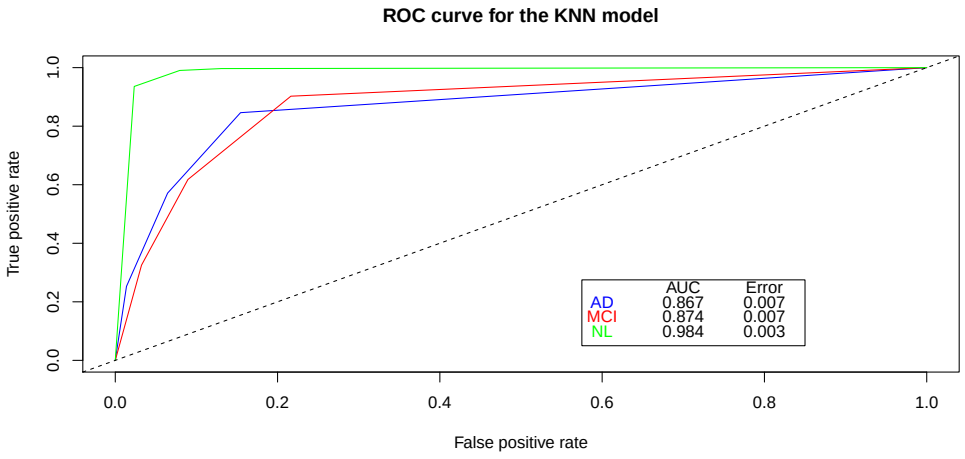


Figura 3.4.1.: *Curva ROC de los 3 vecinos más cercanos.* La tabla inferior muestra el valor del *área bajo la curva (AUC)* de cada subclasificador binario y el error en la medida del *AUC.*

3.4.2. Máquina de vector soporte

A continuación implementamos una *SVM* con núcleo gaussiano siendo $\gamma = 0{,}25$ el mejor valor encontrado para el parámetro γ. En la curva *ROC* vemos cómo este modelo mejora sensiblemente la clasificación de los pacientes enfermos.

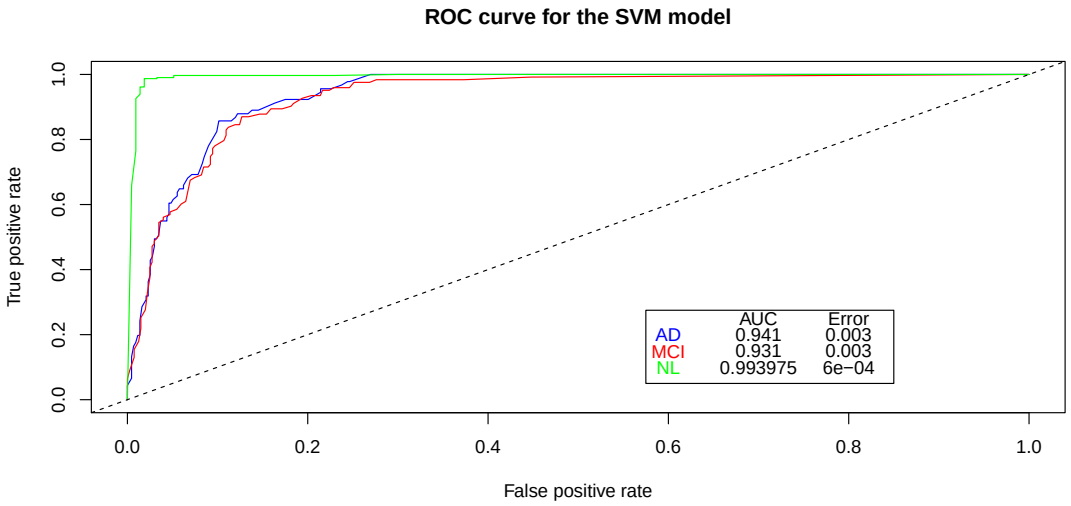


Figura 3.4.2.: Curva ROC de la máquina de vector soporte.

3.4.3. Árbol de decisión

En el caso de un *árbol de decisión* el parámetro de complejidad óptimo para la poda fue $\alpha = 0{,}071$ (ver apéndice C.2).

Representamos gráficamente el árbol resultante y vemos que simplemente con el test cognitivo *CDRSB* podemos realizar un buen diagnóstico donde clasificamos entre pacientes con *AD*, *MCI* o *NL* con probabilidades 0,71, 0,71 y 0,98 respectivamente. Pero si nos quedamos un poco más atrás en el árbol podemos encontrar un resultado mucho mejor y es que usando los nodos 2 y 3 podríamos diferenciar entre pacientes sanos y enfermos ($CDRSB < 0{,}75 \Rightarrow sano$) con unas probabilidades del 0,98 y 0,95. Esto puede ser muy útil y usarse como primera criba a la hora del diagnóstico o cómo criterio de selección de pacientes para futuros estudios. El principal punto a favor de estos resultados es que sólo con una prueba barata, rápida y no invasiva se puede realizar un diagnóstico. Podemos ver en la siguiente figura que el modelo resultante es

$$predicción(\mathbf{x}) = \begin{cases} NL & si\ CDRSB(\mathbf{x}) < 0{,}75 \\ MCI & si\ 0{,}75 \leq CDRSB(\mathbf{x}) < 3{,}2 \\ AD & si\ CDRSB(\mathbf{x}) \geq 3{,}2 \end{cases}$$

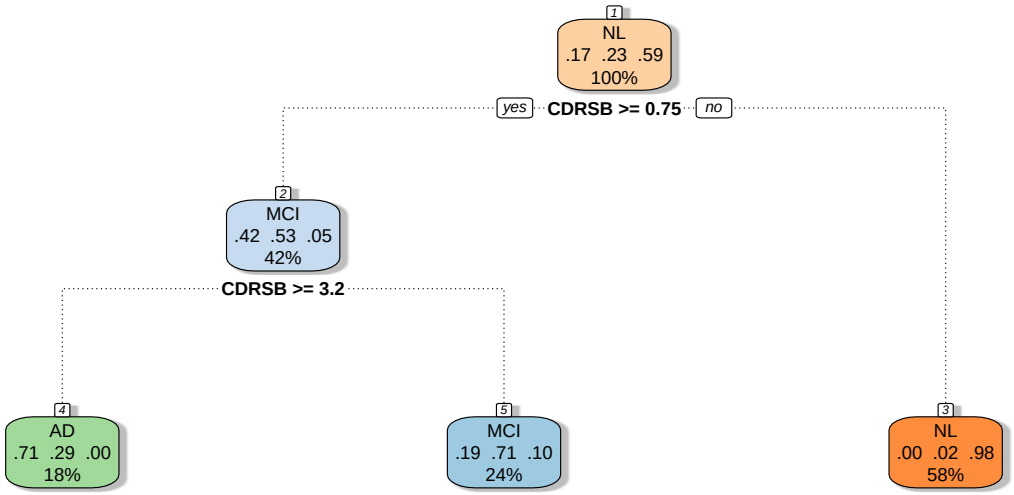


Figura 3.4.3.: *Árbol de decisión ya podado para el diagnóstico del Alzheimer.* Cada nodo del árbol contiene mucha información: el número que aparece encima es el número del nodo, después aparece el diagnóstico asignado a ese nodo, justo debajo las probabilidades de tener cada valor del diagnóstico estando en ese nodo $(prob_{AD}, prob_{MCI}, prob_{NL})$ y por último el porcentaje de los datos que caen en ese nodo.

Al validar el modelo vemos que hemos perdido algo de rendimiento pero sigue siendo bueno y su interpretación es fácil para el lector medio.

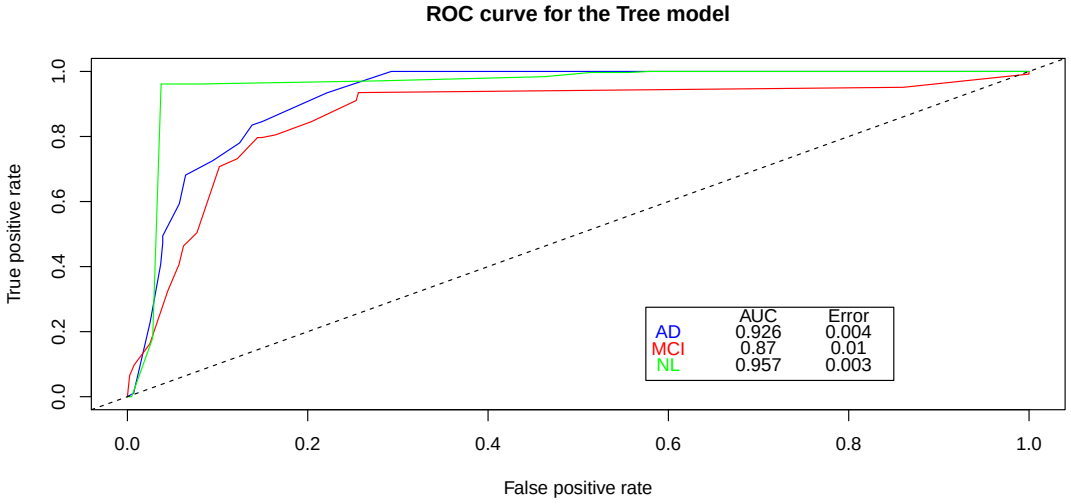


Figura 3.4.4.: Curva ROC para el árbol de decisión.

3.4.4. Bosque aleatorio

Por último aplicamos un *bosque aleatorio* con 200 árboles y entrenando el parámetro m obtuvimos el mejor rendimiento para $m = 4$ (ver apéndice C.2), es decir, cada árbol del bosque se construye usando sólo 4 variables aleatoriamente escogidas del conjunto de datos.

Aunque con este modelo perdemos la interpretabilidad del *árbol de decisión* nos devuelve un gráfico interesante donde se ordenan las variables según su importancia (ver expresión 2.1.7).

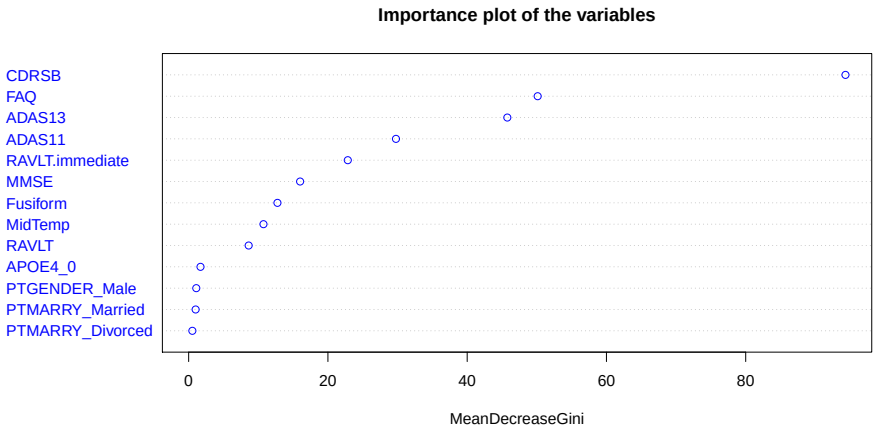


Figura 3.4.5.: *Importancia de las variables.* Vemos cómo dominan los test cognitivos hasta que en el puesto 7 aparecen ya neuroimágenes y otras variables.

En la validación vemos que tiene unos valores del AUC excelentes siendo el mejor modelo de los empleados en términos de rendimiento.

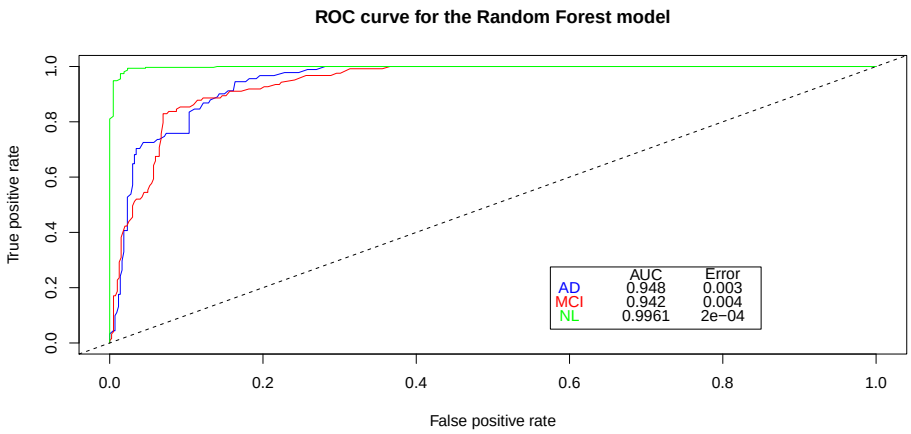


Figura 3.4.6.: Curva ROC del bosque aleatorio.

3.4.5. Conclusiones

Vemos que todos los modelos han obtenido excelentes resultados. La clasificación entre pacientes sanos y enfermos es casi perfecta en algún modelo, siendo más difícil clasificar el nivel de la demencia aunque también se alcanza un buen rendimiento.

Aunque ya hemos medido el rendimiento de los modelos con las curvas *ROC* realizamos una tabla comparativa de los errores de cada modelo

Modelo	*EPE*
KNN	0,16
SVM	0,13
Árbol	0,15
RF	0,097

Cuadro 3.1.: *EPE* para cada modelo en el problema del diagnóstico.

Aunque el *árbol de decisión* sólo supera en rendimiento al modelo *KNN* sigue teniendo valores muy buenos y su interpretabilidad es inmejorable, cualquier personal sanitario sin conocimientos estadísticos podría realizar el test *CDRSB* a un paciente y obtener un diagnóstico muy bueno siguiendo el árbol.

3.5. Conversión a corto plazo

El segundo problema al que nos enfrentamos es realizar un clasificador que identifique conversiones a un estadio más avanzado de la enfermedad o pacientes sanos que en un futuro pasen a estar enfermos. Debido a que en la base de datos no había suficiente información de los niveles de las proteínas *tau* y *amiloide* no tenemos información sobre las dos hipótesis principales de la causa del Alzheimer, quedándonos limitados a valores de variables que se alteran con los síntomas de la enfermedad. Esto supone un gran obstáculo de partida.

Para recuperar la tratabilidad del problema, esta vez la variable objetivo será la conversión *"inmediata"* descrita en la variable *conv.imme* que indicará si ha habido conversión en *DX* desde la última visita, es decir, estaremos creando un clasificador que nos dirá si un paciente convertirá en la próxima visita o no.

$$conv.imme_i = \begin{cases} converter & si\ DX_{i-1} \neq DX_i \\ stable & DX_{i-1} = DX_i \end{cases}, \ i = 1, 2, ...n$$

En este caso hay que realizar un poco de preprocesamiento de nuevo. Comenzamos eliminando los pacientes que sólo tengan una visita ya que no sabemos si han sido estables o no. También tenemos que quitar la última visita de cada paciente por las mismas razones reduciendo así los datos de 2708 a 2048 entradas.

Para añadir significado al estudio longitudinal de cada paciente incluimos nuevos atributos. Para cada variable no demográfica añadimos en cada entrada dos variables nuevas:

diferencia respecto al valor de la visita anterior del mismo paciente y diferencia respecto a la visita inicial. Esta última variable es en realidad redundante al ser combinación lineal del valor actual y el basal, pero la dejamos porque podría ser mejor interpretable. Este proceso provoca un aumento de los atributos, pasando de 35 a 63 variables.

Aparece otro problema aquí y es que la proporción de pacientes *converter* es muy pequeña $\sim 10\,\%$, esto facilita que la *validación cruzada* falle al ser probable que el grupo de validación sólo contenga a sujetos de la clase *stable*. Además los modelos se ven engañados y devuelven una clasificación trivial en la que para cualquier entrada devuelven la etiqueta *stable* y al haber sólo un $10\,\%$ de estables tendrá una precisión del $90\,\%$. Para solucionar el problema cogemos todas las entradas con valor *converter* y una muestra aleatoria de igual tamaño de las estables (436 entradas en total), ya que la muestra del estudio *ADNI* no representa a la población no hay problema en realizar este muestreo, posteriormente comprobaremos que ha sido un método válido.

De nuevo aplicamos un filtro de consistencia (Algoritmo 2.6) obteniendo un subconjunto de 20 variables: *PTGENDER(Male), PTETHCAT(Not.Hisp.Latino), PTMARRY(Married), PTMARRY(Divorced), APOE4_0, APOE4_1, CDRSB, ADAS11, RAVLT, FAQ, Ventricles, Hippocampus, Entorhinal, Fusiform, MidTemp, CDRSB.imme.diff, FAQ.imme.diff, CDRSB.total.diff, FAQ.total.diff y Ventricles.total.diff* .

Pasamos a describir la implementación y validación (CV con $K = 10$) de los modelos.

3.5.1. Vecinos cercanos

El número óptimo de vecinos fue $K = 7$ (ver apéndice C.3).

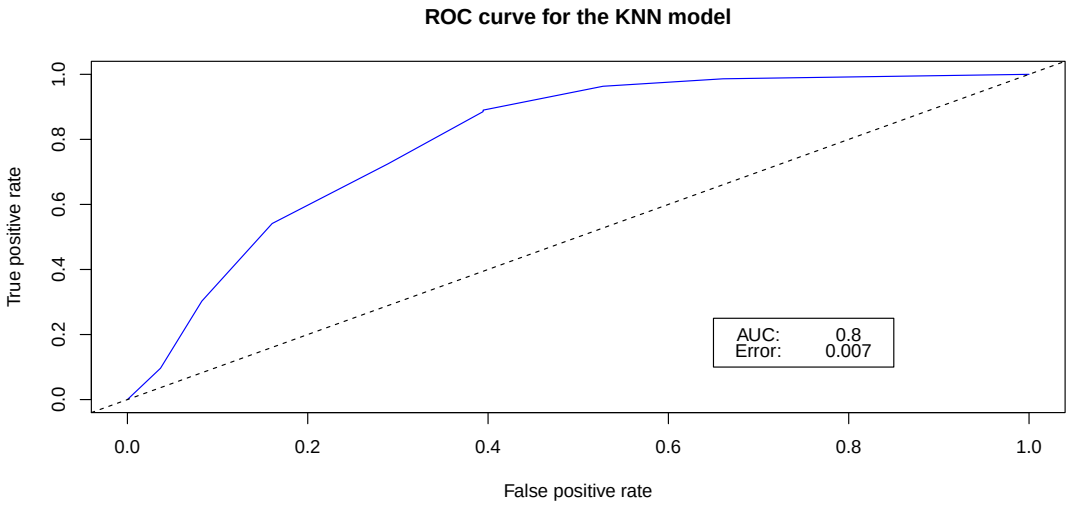


Figura 3.5.1.: *Curva ROC del modelo KNN.*

3.5.2. Máquina de vector soporte

El valor óptimo del parámetro del núcleo gaussiano fue $\gamma = 0{,}6$, vemos también que en esta ocasión no mejora el rendimiento del anterior modelo.

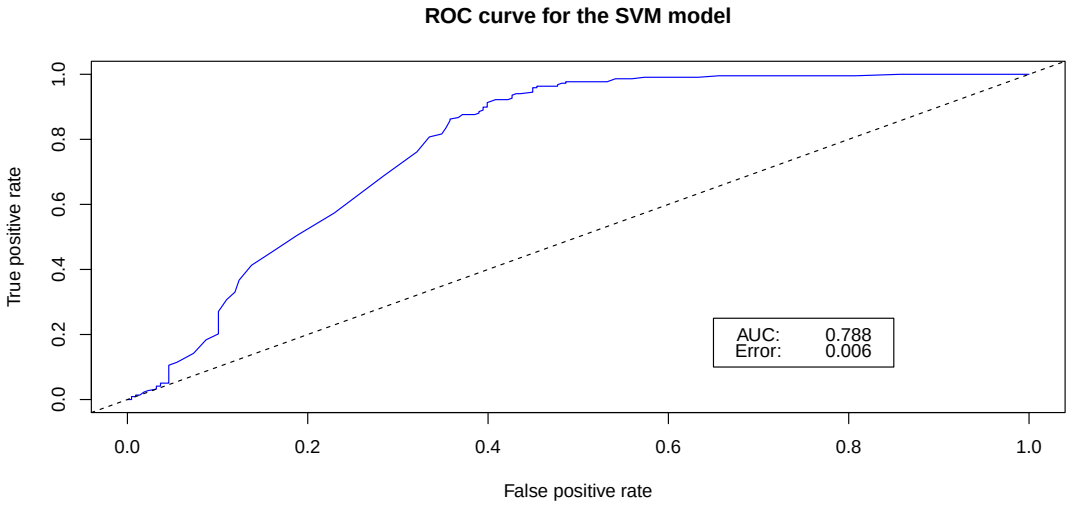


Figura 3.5.2.: Curva ROC de la máquina de vector soporte.

3.5.3. Árbol de decisión

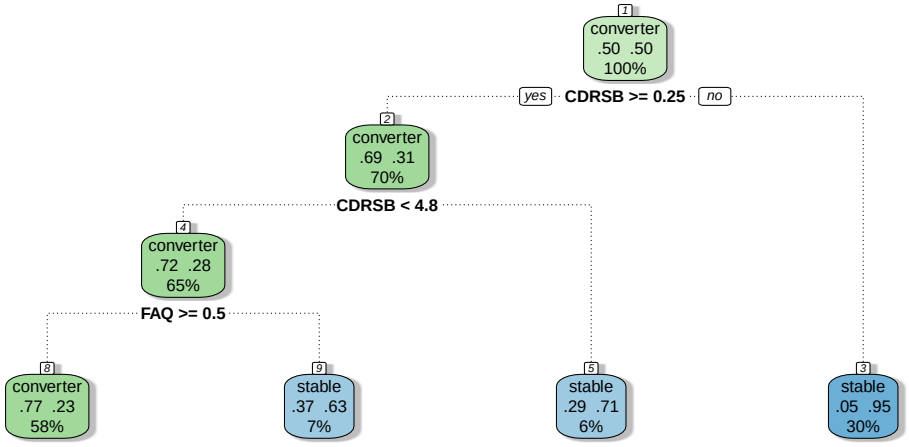
Podamos el árbol usando un parámetro de complejidad α de 0,019 (ver apéndice C.3) y representamos gráficamente el árbol de decisión.

En esta ocasión necesitamos dos pruebas para utilizar el clasificador aunque siguen siendo pruebas baratas, rápidas y no invasivas. Esta vez vemos que si un paciente tiene una puntuación en el *CDRSB* entre [0.25,4.8) y en el $FAQ \geq 0{,}5$ le asignaremos la clase *converter* con probabilidad 0,77. En caso contrario será *stable* con una probabilid dependiente de en que nodo hayamos caído. Podríamos establecer una reglas de seguimiento para cada paciente según los nodos:

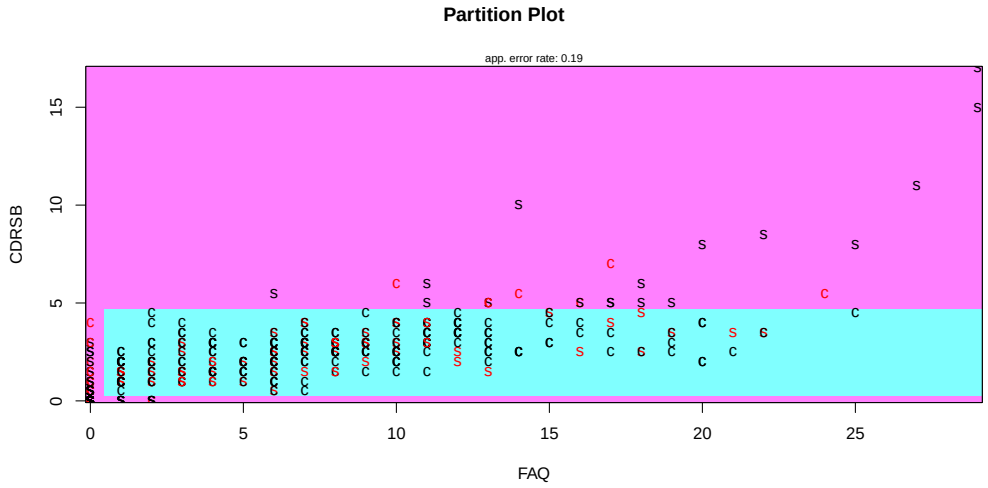
- Los pacientes pertenecientes al nodo 3 casi con certeza se mantendrán en su estado actual luego no hará falta cambiar su tratamiento hasta la siguiente visita donde se repetiría el test.
- Los pacientes del nodo 8 probablemente empeoren su estado luego habrá que comenzar o cambiar el tratamiento.
- Pacientes en los nodos 8 y 5 aunque seguramente no empeoren se les deberían realizar pruebas cruzadas o repetir el test antes de la siguiente visita para confirmar o desmentir el diagnóstico.

En esta ocasión el modelo sería

$$predicción(\mathbf{x}) = \begin{cases} stable & si\ CDRSB < 0{,}25 \\ stable & si\ 0{,}25 \leq CDRSB < 4{,}8\ \wedge\ FAQ < 0{,}5 \\ converter & si\ 0{,}25 \leq CDRSB < 4{,}8\ \wedge\ FAQ \geq 0{,}5 \\ stable & si\ CDRSB \geq 4{,}8 \end{cases}$$



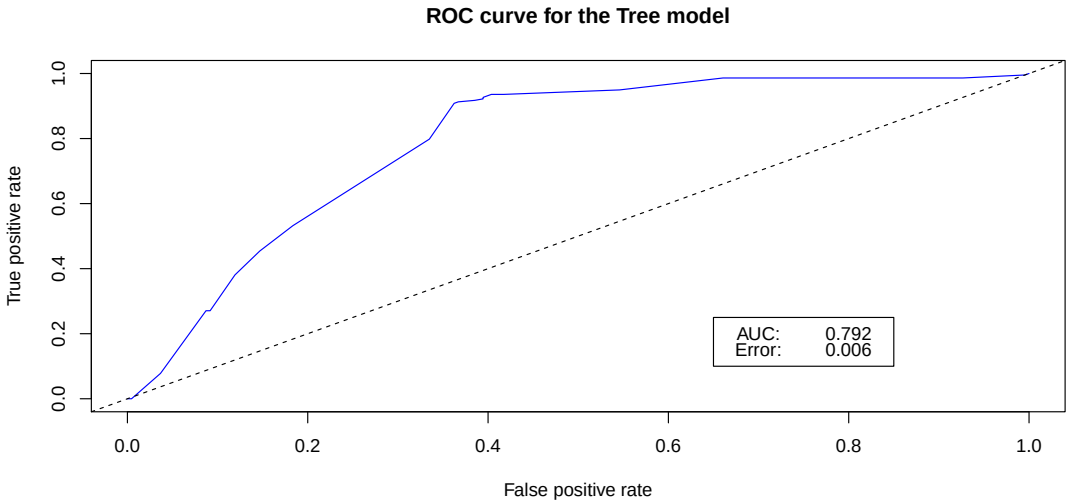
(a) Árbol de decisión ya podado para la predicción de conversión en la siguiente visita.



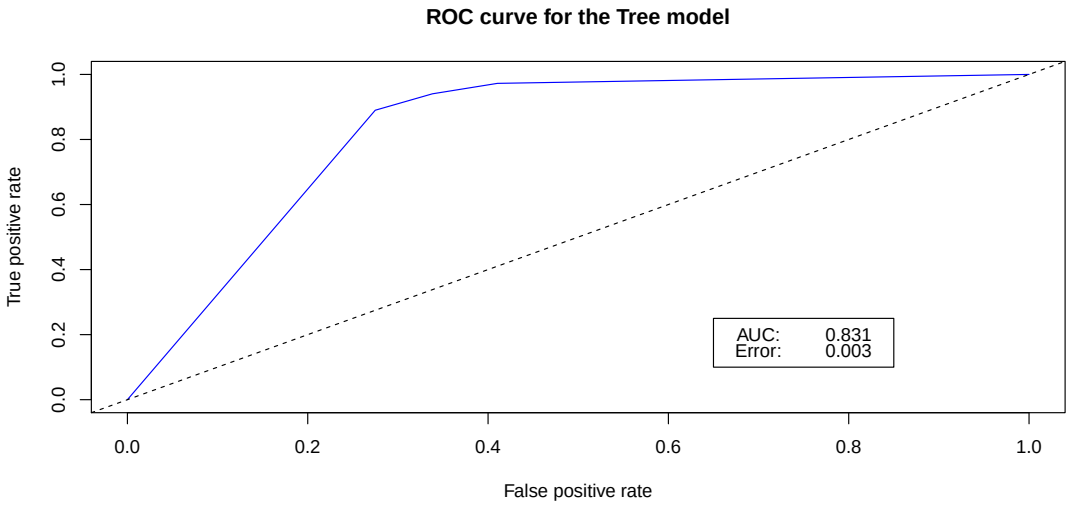
(b) *Superficies de separación del árbol de decisión*. Al ser un modelo bidimensional es muy representativo dibujar el plano con las líneas de separación de las clases de la variable objetivo, en este caso la zona rosa corresponde con las zonas asignadas a los pacientes estables y la azul a los conversores, vemos que en la zona azul hay algunos pacientes mal clasificados (color rojo) mientras que las zonas rosas prácticamente están todos bien clasificados.

Figura 3.5.3.: Representaciones del árbol de decisión.

Al validar el modelo vemos que tiene un rendimiento similar a los dos modelos anteriores, y también repetimos la validación con el mismo modelo pero utilizando todos los datos para comprobar que el muestreo realizado era razonable y no hay sobreajuste.



(a) Con muestreo.



(b) Sin muestreo.

Figura 3.5.4.: Curvas *ROC* para el árbol de decisión.

3.5.4. Bosque aleatorio

Al optimizar el parámetro m vemos que 3 variables es el valor óptimo (ver apéndice C.3), el número de árboles se mantuvo en 200. Al validar este modelo volvemos a obtener rendimientos similares a los demás modelos en este problema

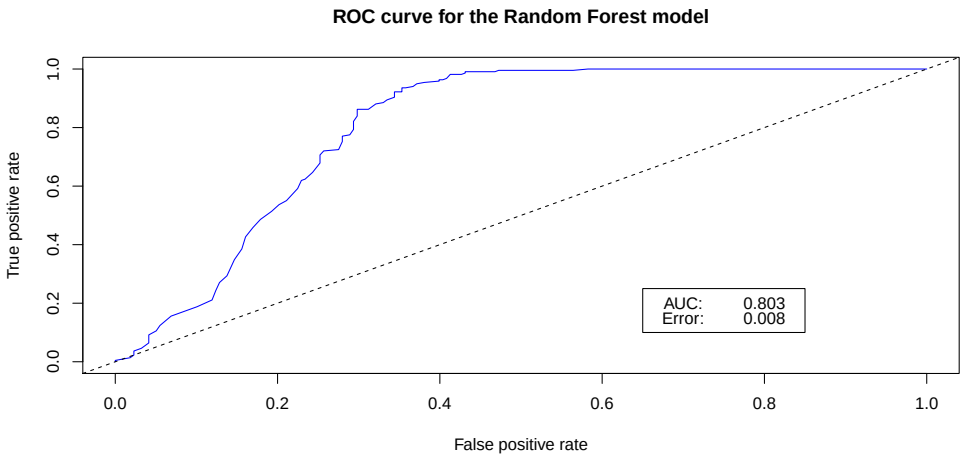


Figura 3.5.5.: Curva *ROC* del bosque aleatorio.

Representamos de nuevo la importancia de las variables

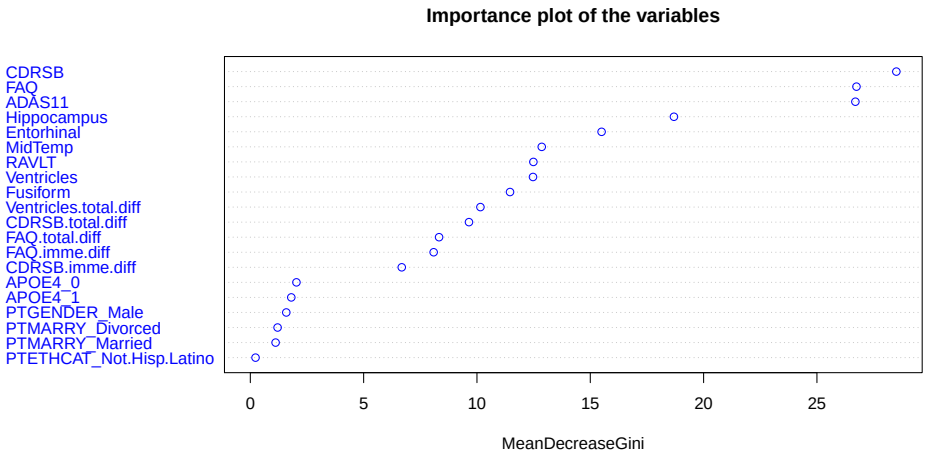


Figura 3.5.6.: *Importancia de las variables.* De nuevo dominan los test cognitivos aunque enseguida aparecen algunas neuroimágenes pero con valores pequeños de importancia en comparación con los test.

3.5.5. Conclusiones

Como ya hemos comentado este problema de predicción a futuro es mucho más difícil que el anterior donde clasificabámos el diagnóstico actual, pero al limitar la predicción a una visita futura redujimos la complejidad.

Al validar los modelos obtenemos $AUC \simeq 0{,}8$ que en muchos problemas sería un resultado bastante bueno pero en nuestro caso no son tan buenas noticias ya que sólo estamos previendo conversión a una visita futura y además para obtener un test diagnóstico deberíamos alcanzar valores como los del problema anterior.

En esta clasificación el rendimiento de los modelos ha disminuido pero se sigue manteniendo en niveles muy buenos, comparamos el *EPE* de los cuatro modelos

Modelo	*EPE*
KNN	0,28
SVM	0,24
Árbol	0,23
RF	0,22

Cuadro 3.2.: *EPE* para cada modelo en el problema de la conversión a corto plazo.

Esta vez tenemos más fácil la elección del modelo ya que todos tienen eficiencias similares y de nuevo escogemos el *árbol de decisión* por su gran interpretabilidad.

En comparativa, otros estudios [25] [26] obtuvieron precisiones entre el 67 y el 84 % con valores del AUC de 0,796 en el problema de conversión sólo entre demencia leve (MCI) y Alzheimer (AD). Hemos conseguido mejorar estos rendimientos con un clasificador simple, además hemos contemplado la conversión de NL a MCI y usado una muestra significativamente más grande de pacientes.

3.6. Conversión a largo plazo

De nuevo escogimos el modelo de *árbol de decisión* por su gran interpretabilidad y además en este caso su rendimiento era similar al de los demás modelos. El siguiente paso es traducir al verdadero objetivo, predecir la enfermedad en cualquier visita, para ello usaremos teoría de probabilidad básica.

Con los resultados del *árbol de decisión* anterior (figura 3.5.3a) y asumiendo que podemos extrapolar los resultados a toda la población del estudio, tenemos las siguientes probabilidades de convertir o no en la siguiente visita dados los valores de una entrada

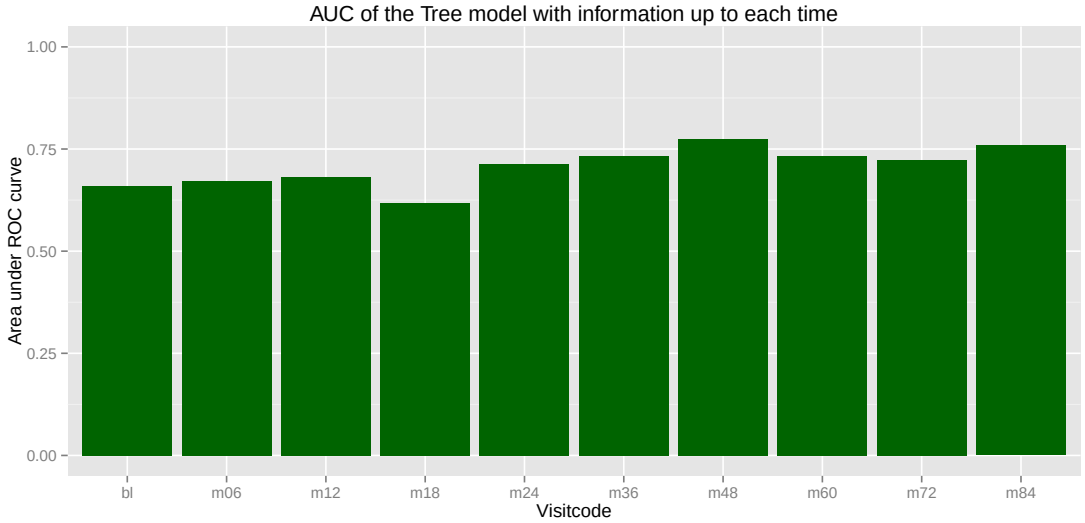
$$P(conv.imme(\mathbf{x}) = converter) = \begin{cases} 0{,}77 & si\ \mathbf{x} \in nodo_8 \\ 0{,}37 & si\ \mathbf{x} \in nodo_9 \\ 0{,}29 & si\ \mathbf{x} \in nodo_5 \\ 0{,}05 & si\ \mathbf{x} \in nodo_3 \end{cases}$$

La probabilidad de que el paciente i convierta alguna vez conociendo sólo su información hasta la visita j será

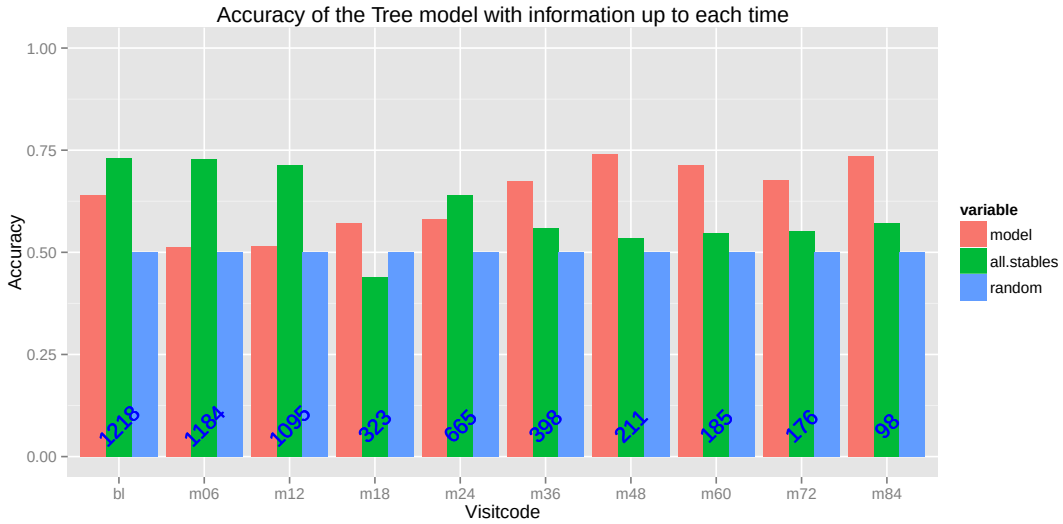
$$P(conv(i_j) = converter) = \\ P(conv.imme(\mathbf{x}_j^i) = converter) \cdot \prod_{k=1}^{j-1} 1 - P(conv.imme(\mathbf{x}_k^i) = converter)$$

siendo $\mathbf{x}_b^a$ la entrada del paciente a para la visita b.

Procedemos a la validación del modelo, vemos en la figura 3.6.1a que el rendimiento es mediocre, aunque se consiguen resultados razonables cuando disponemos de un amplio seguimiento de los pacientes. En este problema el rival a batir no es un modelo aleatorio de AUC 0,5, sino un modelo "tonto" donde todos los pacientes reciben una predicción de estabilidad, este modelo tendría unos valores del AUC entre 0,4 y 0,7 dependiendo de la visita. Vemos en la figura 3.6.1b que nuestro modelo comienza con un rendimiento similar, aunque enseguida empeora y finalmente llega a mejorar a este modelo "tonto". Por tanto nuestros resultados no son buenos inicialmente que es lo que nos interesa. Volviendo a comparar con los estudios [25] [26] necesitamos llegar a las últimas visitas para igualar los rendimientos.



(a) *AUC del modelo.* En cada visita se cogía información de las visitas anteriores también, por eso el rendimiento del modelo aumenta con el tiempo al disponer de más información.



(b) *Precisión del modelo.* En esta gráfica se muestra la precisión (1-*EPE*) de nuestro modelo, un modelo aleatorio y un modelo simple de asignar a todo el mundo un diagnóstico de estabilidad. Los números azules que aparecen en cada visita son el número de pacientes en cada tiempo del estudio, la paulatina disminución se debe a la gente que abandona secuencialmente el estudio por diversas razones (salvo en el mes 18 que por el protocolo no todos los pacientes tenían que acudir a esa visita).

Figura 3.6.1.: Validación del modelo para predicción temprana.

4. Conclusiones

Este proyecto contaba con numerosos problemas de entrada. En primer lugar entrenábamos nuestros modelos con la variable diagnóstico que tenía cierta interpretabilidad (hasta que el paciente no muere y se le examina el cerebro no se sabe con certeza que tenga Alzheimer). En segundo lugar nos vimos obligados a desestimar los mejores modelos por su difícil interpretación ya que el objetivo final sería que un médico sin conocimientos en Minería de Datos pudiera aplicar el modelo. Finalmente el problema de conversión a largo plazo que es el gran objetivo no puede ser resuelto eficientemente con nuestra base de datos ya que las únicas variables con suficiente muestra eran sintomáticas.

En el caso del diagnóstico actual obtuvimos unos resultados excelentes usando sólo una prueba barata y no invasiva. En cuanto a predecir conversiones del diagnóstico, el rendimiento fue bueno, mejorando a otros estudios prediciendo a 6-12 meses vista mientras que a largo plazo no llegamos a igualar la eficiencia de estos estudios.

Como futuras líneas de investigación habría que completar la base de datos con mucha más información sobre las proteínas *tau* y *amiloide* ya que son las dos hipótesis más importantes sobre la causa del Alzheimer, con dicha información sí que se podría abordar correctamente el problema del diagnóstico temprano con técnicas de Minería de Datos.

Bibliografía

[1] Alzheimer Disease
https://en.wikipedia.org/wiki/Alzheimer's_disease

[2] Alzheimer's Disease Neuroimaging Initiative
http://www.adni-info.org/

[3] Hardy J, Allsop D. Amyloid Deposition as the Central Event in the Aetiology of Alzheimer's Disease. *Trends in Pharmacological Sciences*. 1991;12(10):383–88. doi:10.1016/0165-6147(91)90609-V. PMID 1763432.

[4] Holmes C, Boche D, Wilkinson D, Yadegarfar G, Hopkins V, Bayer A, Jones RW, Bullock R, Love S, Neal JW, Zotova E, Nicoll JA. Long-term Effects of Abeta42 Immunisation in Alzheimer's Disease: Follow-up of a Randomised, Placebo-controlled Phase I Trial. *Lancet*. 2008;372(9634):216–23. doi:10.1016/S0140-6736(08)61075-2. PMID 18640458.

[5] Mudher A, Lovestone S. Alzheimer's disease-do tauists and baptists finally shake hands?. *Trends in Neuroscience*s. 2002;25(1):22–26. doi:10.1016/S0166-2236(00)02031-2. PMID 11801334.

[6] Mahley RW, Weisgraber KH, Huang Y. Apolipoprotein E4: a causative factor and therapeutic target in neuropathology, including Alzheimer's disease. Proceedings of the National Academy of Sciences of the United States of America. 2006;103(15):5644–51. doi:10.1073/pnas.0600549103. PMID 16567625.

[7] McKhann G, Drachman D, Folstein M, Katzman R, Price D, Stadlan EM. Clinical Diagnosis of Alzheimer's Disease: Report of the NINCDS-ADRDA Work Group under the Auspices of Department of Health and Human Services Task Force on Alzheimer's Disease. Neurology. 1984;34(7):939–44. doi:10.1212/wnl.34.7.939. PMID 6610841.

[8] De Meyer G, Shapiro F, Vanderstichele H, Vanmechelen E, Engelborghs S, De Deyn PP, Coart E, Hansson O, Minthon L, Zetterberg H, Blennow K, Shaw L, Trojanowski JQ. Diagnosis-Independent Alzheimer Disease Biomarker Signature in Cognitively Normal Elderly People. Archives of Neurology. 2010;67(8):949–56. doi:10.1001/archneurol.2010.179. PMID 20697045.

[9] ADNIMERGE
http://adni.bitbucket.org/

[10] The Comprehensive R Archive Network
http://cran.r-project.org/

[11] Key ADNI tables merged into one table
http://adni.bitbucket.org/adnimerge.html

[12] Data minning
https://en.wikipedia.org/wiki/Data_mining

[13] R. A. Fisher (1936). The use of multiple measurements in taxonomic problems. *Annals of Eugenics* 7 (2): 179–188. doi:10.1111/j.1469-1809.1936.tb02137.x.

[14] Robert Nisbet, John Elder, Gary Miner (2009). Top 10 Data Mining Mistakes. En *Handbook of Statistical Analysis and Data Mining Applications* (Páginas 733-754).

[15] Christopher M. Bishop (2006). *Pattern Recognition and Machine Learning*, Springer.

[16] Trevor Hastie, Robert Tibshirani, Jerome Friedman (2009). *The Elements of Statistical Learning:Data Mining, Inference, and Prediction. Second Edition,* Springer.

[17] Alkaike information criterion
https://en.wikipedia.org/wiki/Akaike_information_criterion

[18] Bayesian information criterion
https://en.wikipedia.org/wiki/Bayesian_information_criterion

[19] Receiver operating characteristic
https://en.wikipedia.org/wiki/Receiver_operating_characteristic

[20] Hua Yin, Hongbin Dong, Yuxuan Li (2009). A Cluster-based Noise Detection Algorithm. *First International Workshop on Database Technology and Applications.*

[21] k-means clustering
https://en.wikipedia.org/wiki/K-means_clustering

[22] Manoranjan Dash a, Huan Liu (2003). Consistency-based search in feature selection. *ELSEVIER, Artificial Intelligence 151 (2003) 155–176.*

[23] Pearson product-moment correlation coefficient
https://en.wikipedia.org/wiki/Pearson_product-moment_correlation_coefficient

[24] Daniel, Peña (2002). Componentes principales. En *ANÁLISIS DE DATOS MULTIVARIANTES* (Páginas 137-178), MCGRAW-HILL.

[25] Yue Cui, et al (2011). Identification of Conversion from Mild Cognitive Impairment to Alzheimer's Disease Using Multivariate Predictors. *PLoS One*; 6(7): e21896.

[26] Robert M. Chapman, et al (2011). Predicting Conversion from Mild Cognitive Impairment to Alzheimer's Disease Using Neuropsychological Tests and Multivariate Methods. *J Clin Exp Neuropsychol.* Feb 2011; 33(2): 187–199.

A. Librerías del lenguaje R

Recopilación alfabética de los paquetes con software de terceros usados en este proyecto, se añade una breve descripción de para qué fue utilizada cada librería.

- *ADNIMERGE:* Contiene la base de datos del proyecto *ADNI*, es la única librería usada de acceso restringido a los investigadores del proyecto.
  http://adni.bitbucket.org/
- *car:* Tranformaciones Yeo and Jonhson para conseguir normalidad en el análisis de componentes principales.
  http://cran.r-project.org/web/packages/car/index.html
- *caret:* Modelo vecinos cercanos.
  http://cran.r-project.org/web/packages/caret/index.html
- *e1071:* Usada para implementar las máquinas de vector soporte.
  http://cran.r-project.org/web/packages/e1071/index.html
- *FSelector:* Filtro de consistencia para la selección de variables.
  http://cran.r-project.org/web/packages/FSelector/index.html
- *GGally:* Extension de la librería ggplot2.
  http://cran.r-project.org/web/packages/GGally/index.html
- *ggplot2:* Representaciones gráficas elegantes de análisis estadísticos.
  http://cran.r-project.org/web/packages/ggplot2/index.html
- *igraph:* Implementación y representación de grafos.
  http://cran.r-project.org/web/packages/igraph/index.html
- *klaR:* Representaciones gráficas de algunos modelos.
  http://cran.r-project.org/web/packages/MASS/index.html
- *matrixcalc:* Comprobación requisitos del análisis de componentes principales.
  http://cran.r-project.org/web/packages/klaR/index.html
- *MASS:* Modelo análisis discriminante lineal (no recogido en el informe).
  http://cran.r-project.org/web/packages/MASS/index.html
- *plyr:* Herramientas para aplicar funciones a la bases de datos.
  http://cran.r-project.org/web/packages/plyr/index.html

- *psych:* Se usó en el análisis de componentes principales.
  http://cran.r-project.org/web/packages/psych/index.html
- *randomForest:* Modelo bosque aleatorio.
  http://cran.r-project.org/web/packages/randomForest/index.html
- *rattle:* Interfaz de ayuda para aplicar técnicas de minería de datos.
  http://cran.r-project.org/web/packages/rattle/index.html
- *RColorBrewer:* Paleta de colores para algunas representaciones gráficas.
  http://cran.r-project.org/web/packages/RColorBrewer/index.html
- *reshape:* Herramientas para operar con bases de datos.
  http://cran.r-project.org/web/packages/reshape/index.html
- *reshape2:* Herramientas para operar con bases de datos.
  http://cran.r-project.org/web/packages/reshape2/index.html
- *rpart:* Modelo árbol de decisión.
  http://cran.r-project.org/web/packages/rpart/index.html
- *rpart.plot:* Gráficos elegantes para las funciones de la librería *rpart.*
  http://cran.r-project.org/web/packages/rpart.plot/
- *zoo:* Series temporales, usada en la interpolación de datos faltantes.
  http://cran.r-project.org/web/packages/zoo/index.html

Estas librerías contenían sólo una porción de las funciones usadas, el resto de algoritmos, el preprocesamiento y la implementación de todas las rutinas a una base de datos en particular fue con código propio. Para todo esto se crearon librerías propias con código genérico extrapolable a otros problemas.

B. Principales test cognitivos

Los test cognitivos son pruebas de evaluación de la cognición según preguntas en diversas áreas, hay numerosas pruebas procedentes de test cognitivos, pero explicaremos sólo los tres principales.

- *Mini Mental State Examination (MMSE):* Sirve para detectar la presencia o ausencia de demencia. Es un cuestionario de 30 preguntas que se realiza en menos de 10 minutos, examina diversas funciones cognitivas como la aritmética, memoria y orientación. No permite clasificar el grado de demencia.
- *Alzheimer's Disease Assessment Scale – Cognitive (ADAS-cog):* Mejora al *MMSE* permitiendo diferenciar entre demencia leve y grave y se enfoca principalmente en la memoria y el lenguaje. Tiene 11 partes y su realización dura unos 30 minutos, esto constituye el test *ADAS11* pero también existe una versión ampliada con 13 items que es el test *ADAS13.*
- *Clinical Dementia Rating (CDR):* Además de detectar la presencia de demencia distingue entre más niveles de severidad. Se puntúan 6 áreas: memoria, orientación, juicio y resolución de problemas, comunidad , hogar y ocio y autocuidado. Cada área puede tener una puntuación entre 0 y 3, la suma de todas forma el test *CDR-sum of boxes* ó *CDRSB,* que es el usado en nuestros datos.

C. Gráficas

C.1. Análisis estadístico

Recopilación de las representaciones gráficas realizadas para conocer la distribución de cada variable segmentando la población según el valor del diagnóstico.

Figura C.1.1.: Gráfico de densidad del test *ADAS11*.

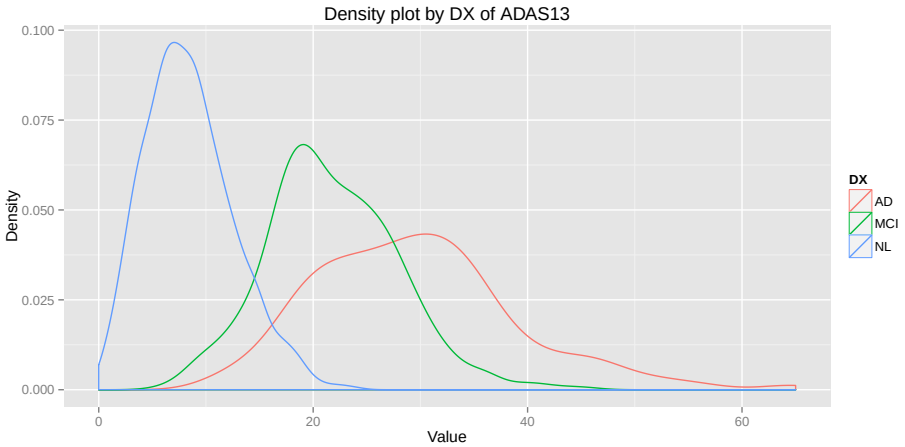


Figura C.1.2.: Gráfico de densidad del test *ADAS13*.

Figura C.1.3.: Gráfico de densidad de la edad.

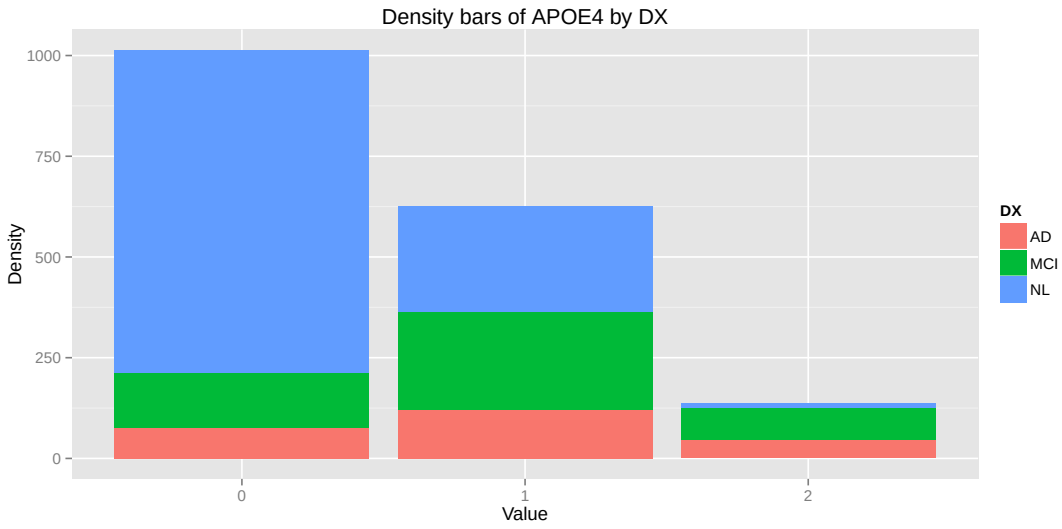


Figura C.1.4.: Gráfico de barras para la variable *APOE4*.

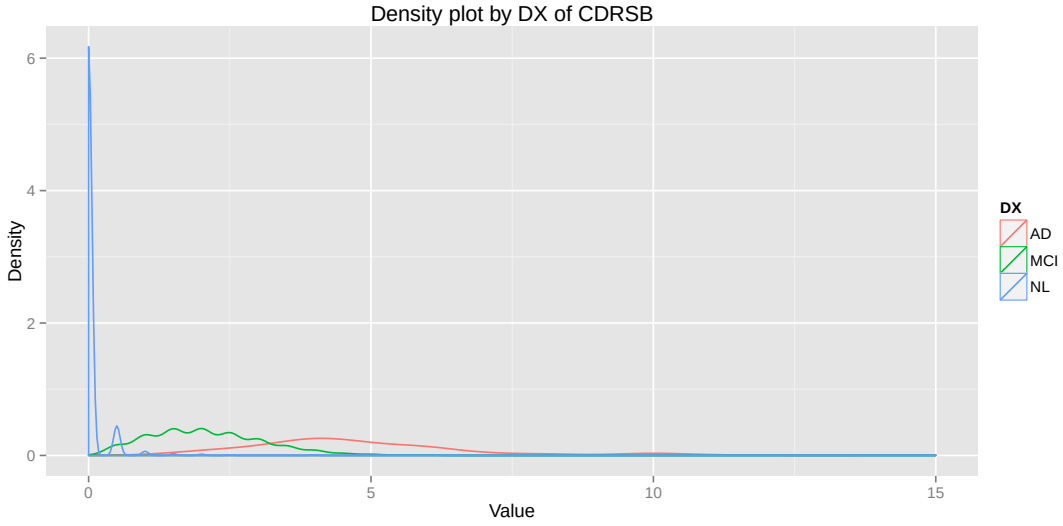


Figura C.1.5.: Gráfico de densidad del test *CDRSB*.

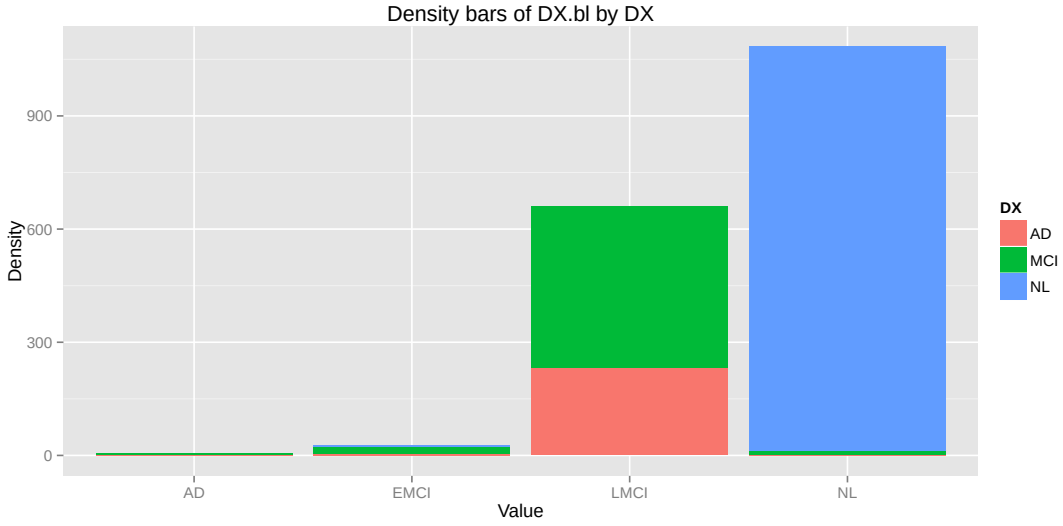


Figura C.1.6.: Gráfico de barras de la variable *DX.bl* (diagnóstico inicial). Los valores *EMCI* y *LMCI* son demencia leve temprana y tardía respectivamente.

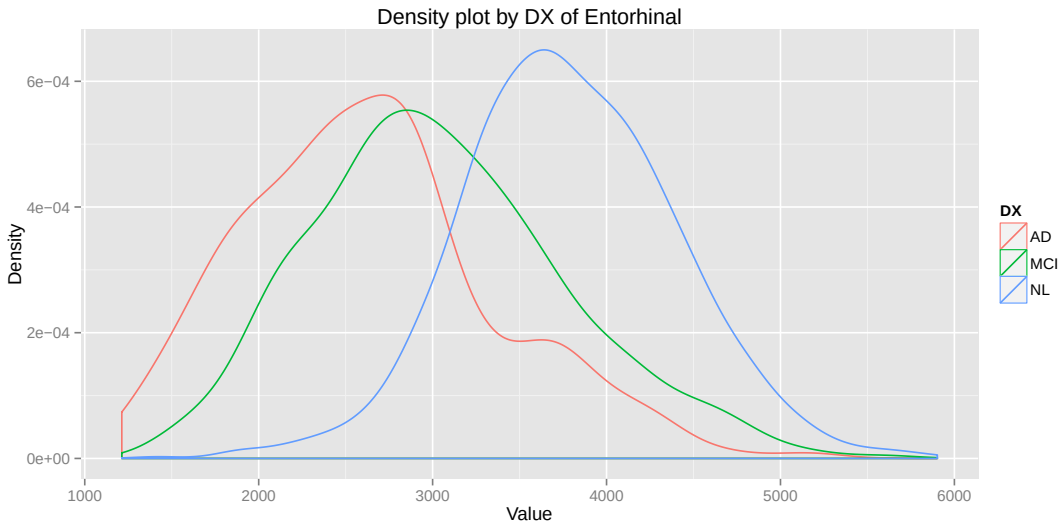


Figura C.1.7.: Gráfico de densidad de la corteza entorrinal.

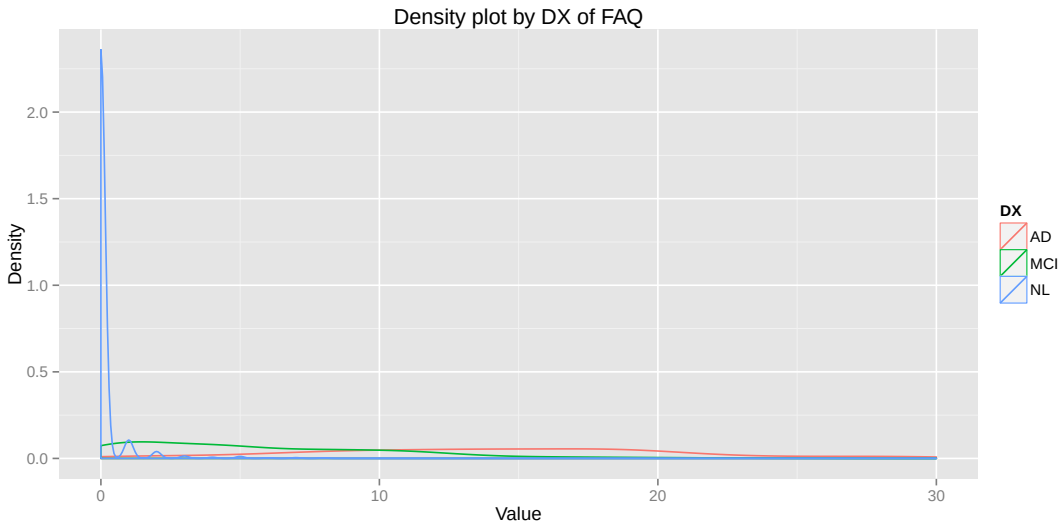


Figura C.1.8.: Gráfico de densidad del test *FAQ*.

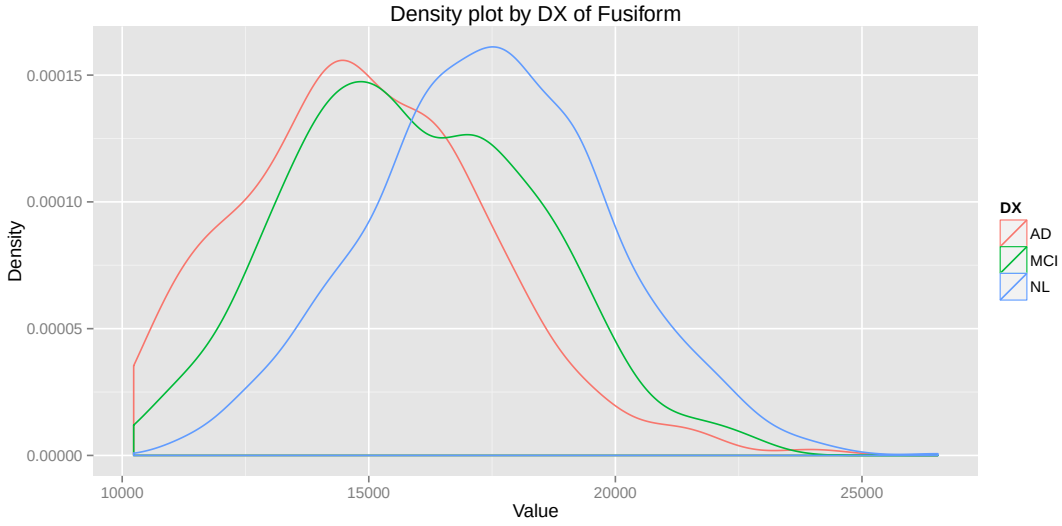


Figura C.1.9.: Gráfico de densidad del giro fusiforme.

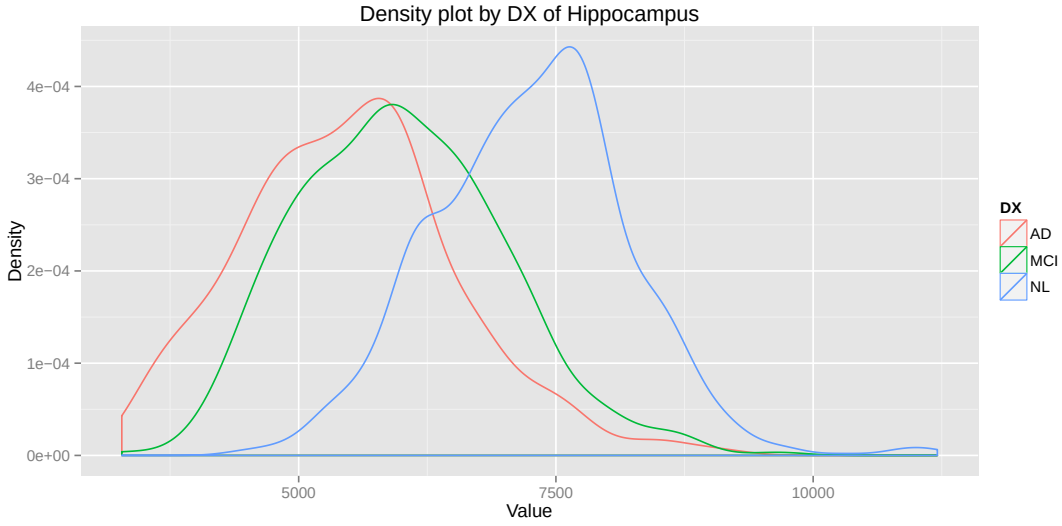


Figura C.1.10.: Gráfico de densidad del hipocampo.

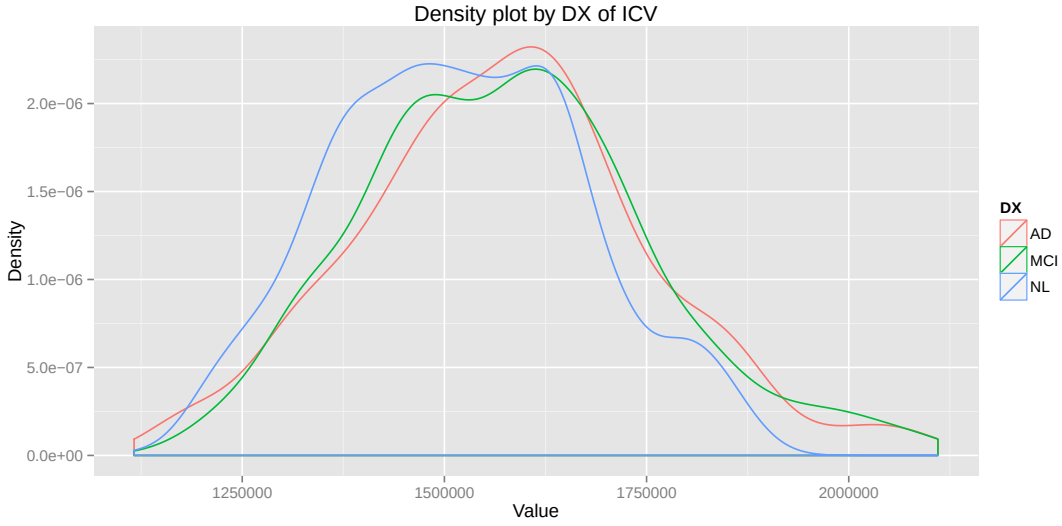


Figura C.1.11.: Gráfico de densidad para el volumen intracraneal.

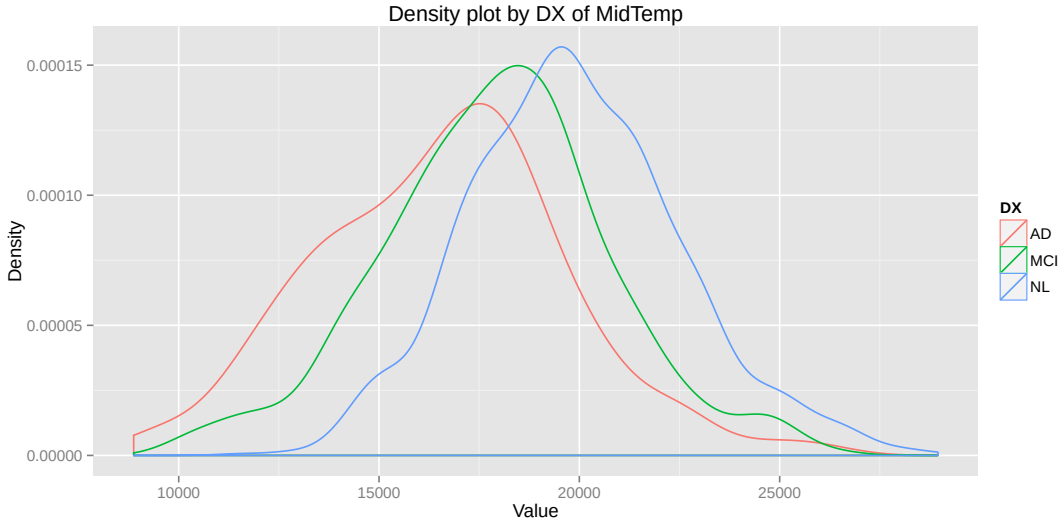


Figura C.1.12.: Gráfico de densidad del lóbulo temporal medio.

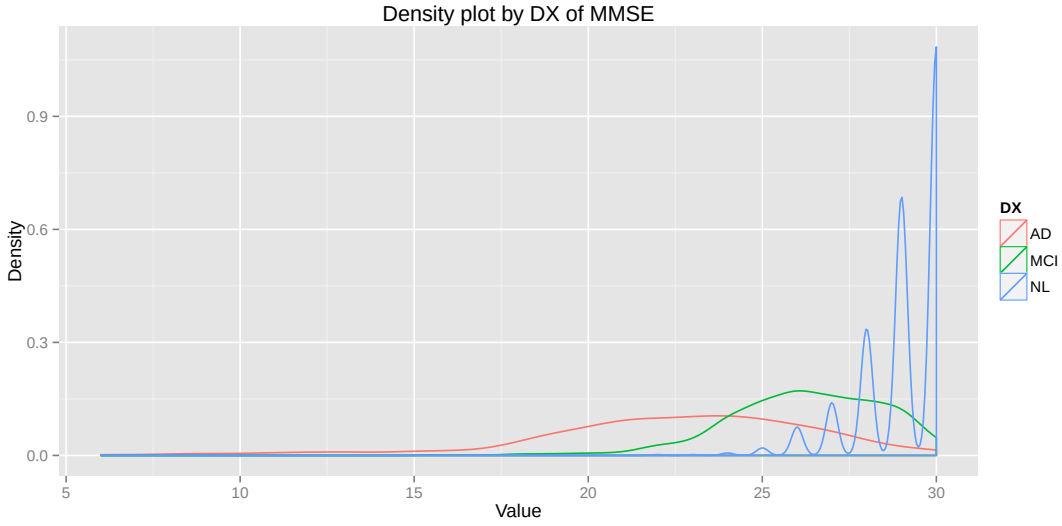


Figura C.1.13.: Gráfico de densidad del test *MMSE*.

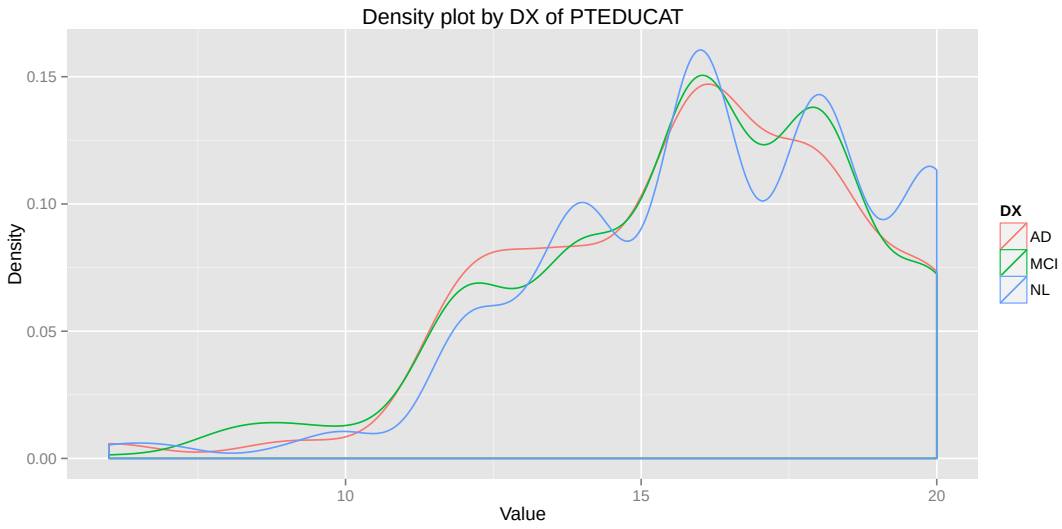


Figura C.1.14.: Gráfico de densidad de los años de educación.

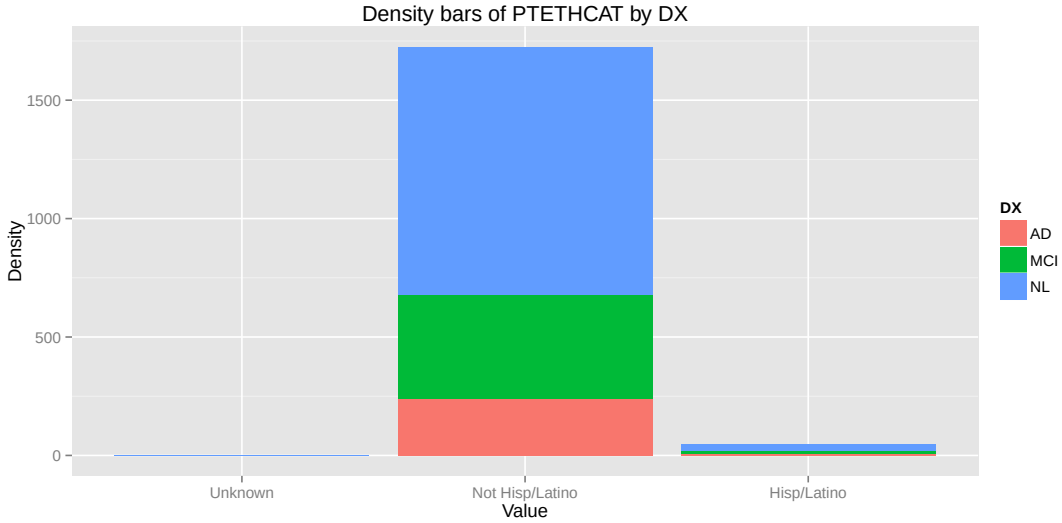


Figura C.1.15.: Gráfico de barras para la étnia.

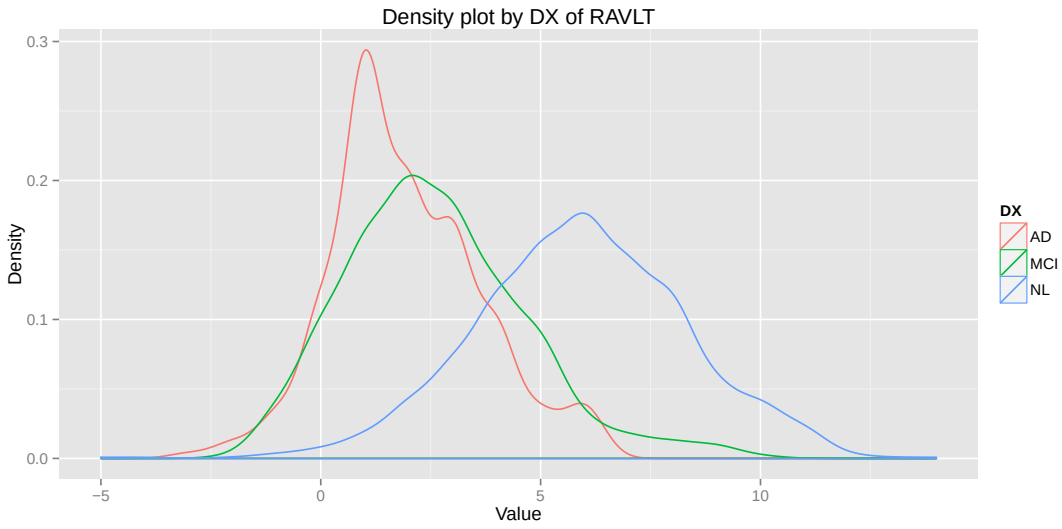


Figura C.1.16.: Gráfico de densidad del test *RAVLT*.

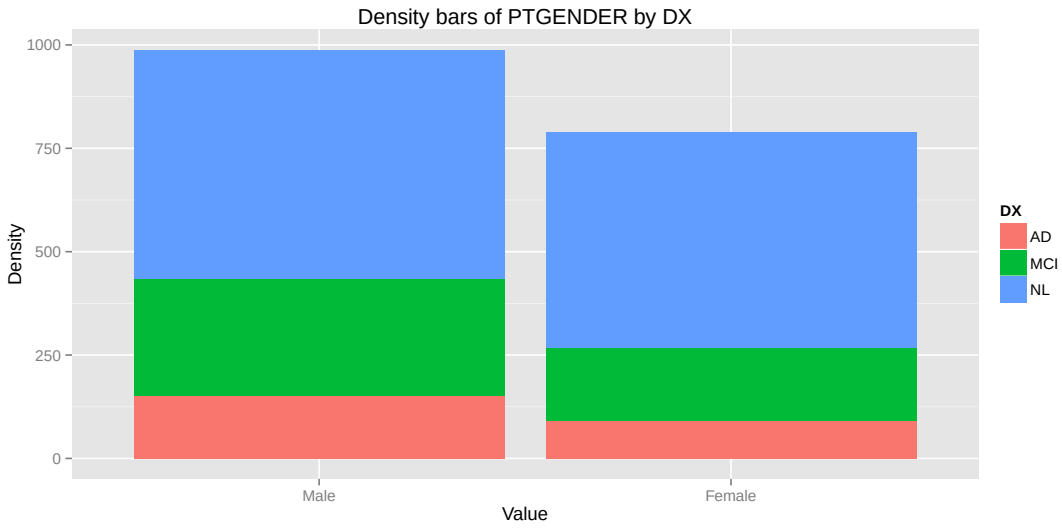


Figura C.1.17.: Gráfico de barras para el género.

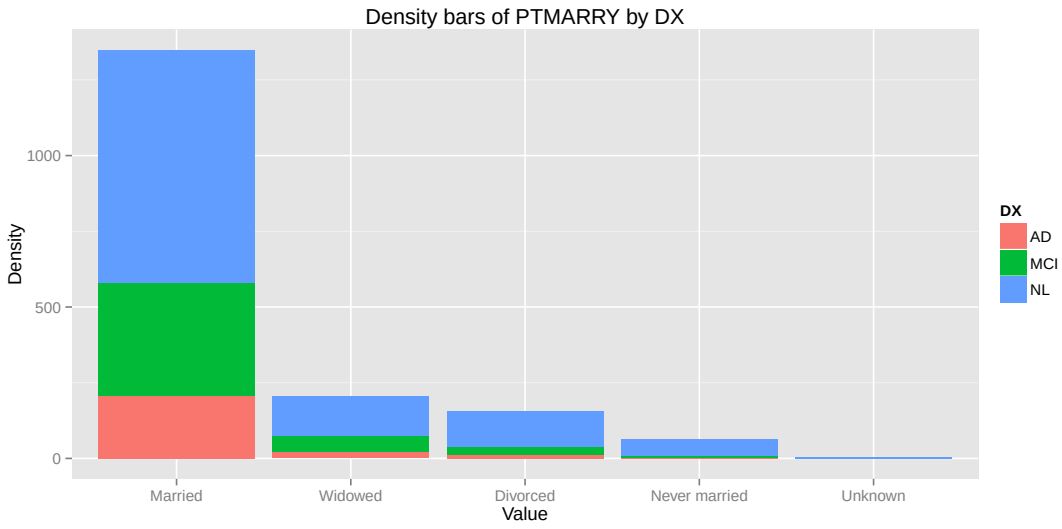


Figura C.1.18.: Gráfico de barras para el estado civil.

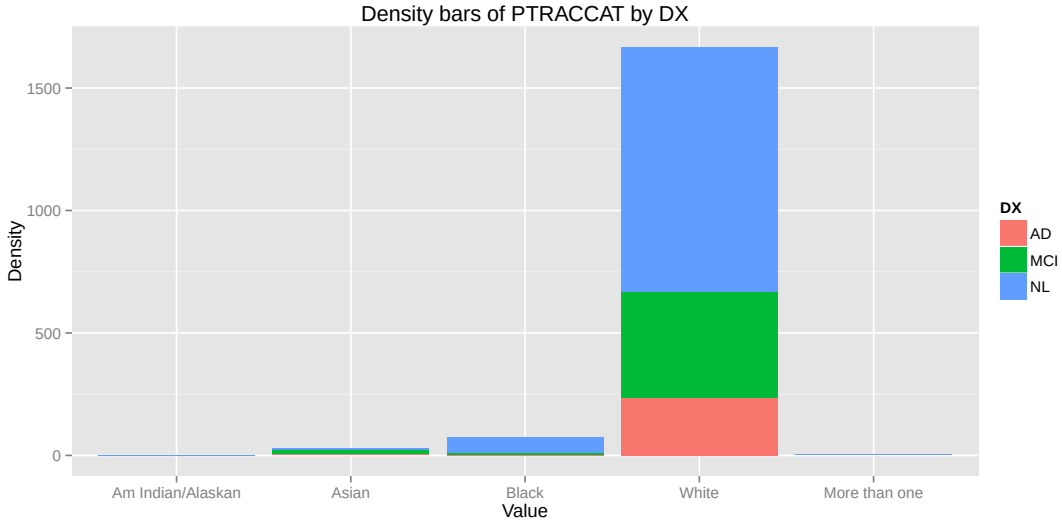


Figura C.1.19.: Gráfico de barras de la raza.

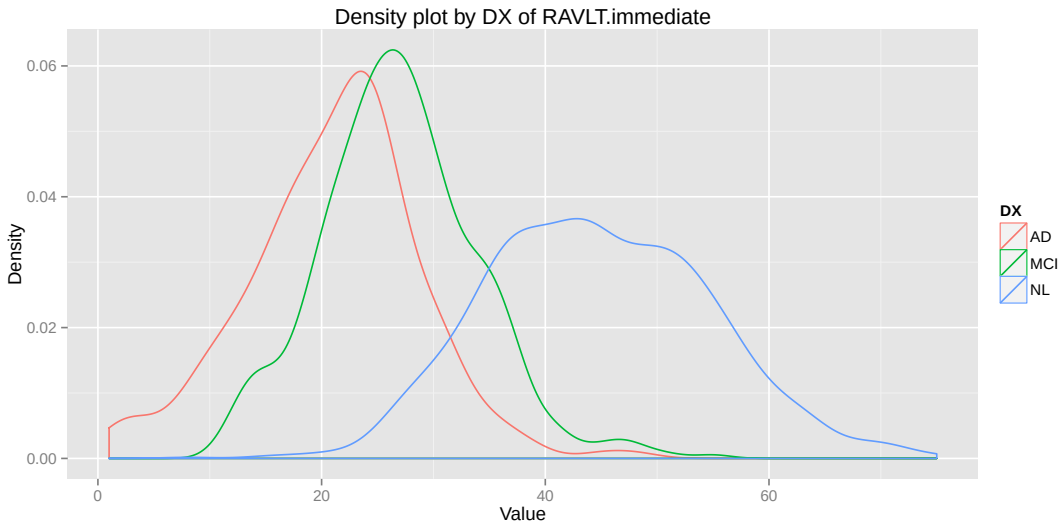


Figura C.1.20.: Gráfico de densidad del test *RAVLT.immediate*.

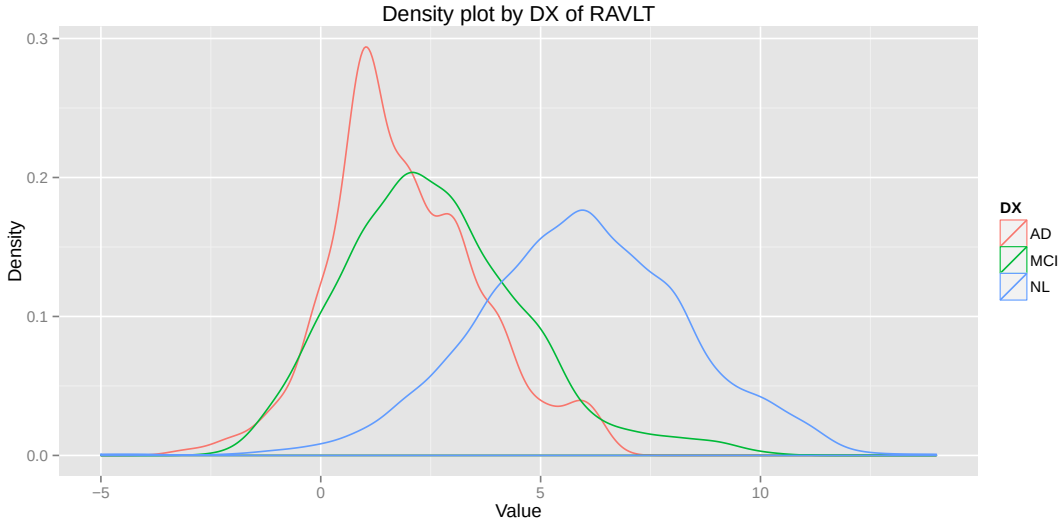


Figura C.1.21.: Gráfico de densidad del test *RAVLT*.

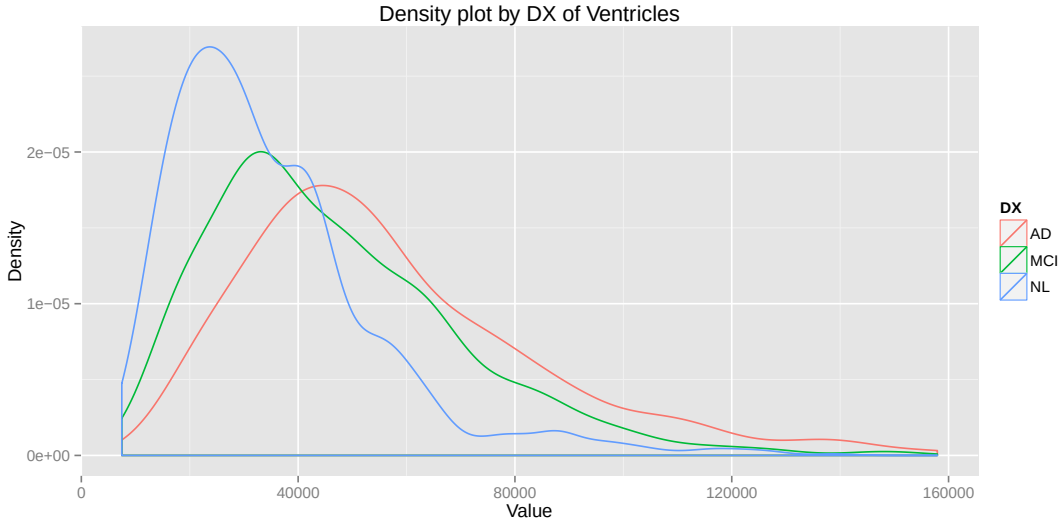


Figura C.1.22.: Gráfico de densidad para el volumen ventricular.

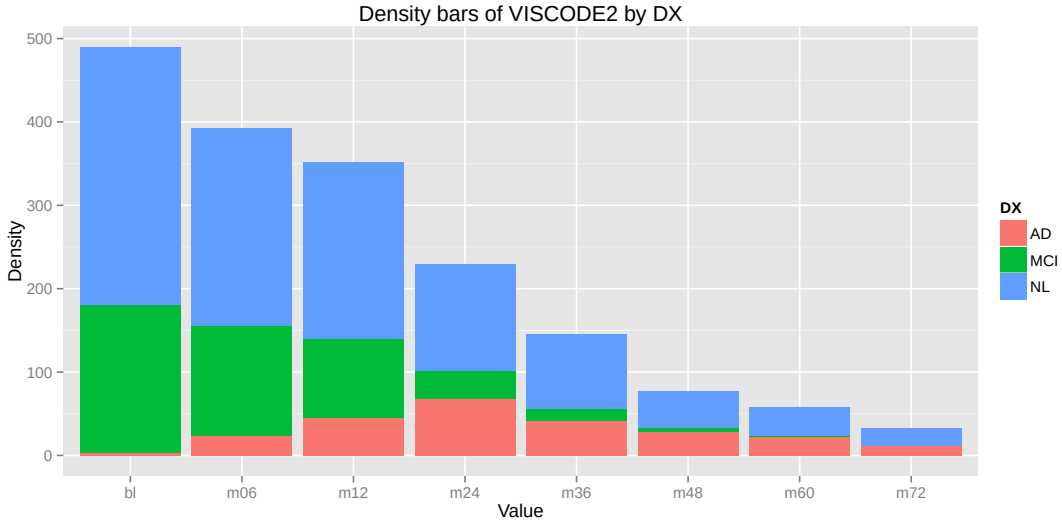


Figura C.1.23.: Gráfico de barras para el código de visita.

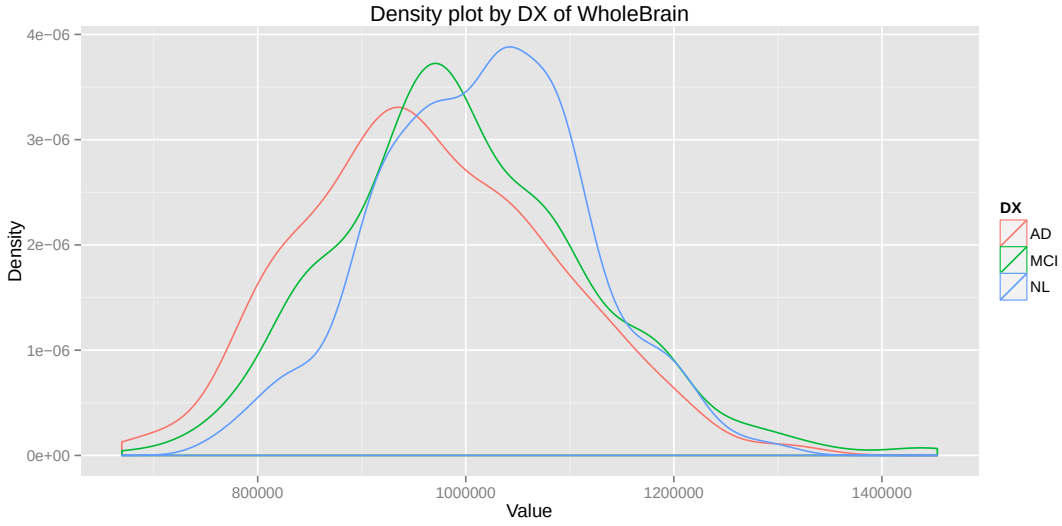


Figura C.1.24.: Gráfico de densidad del volumen cerebral.

C.2. Diagnóstico

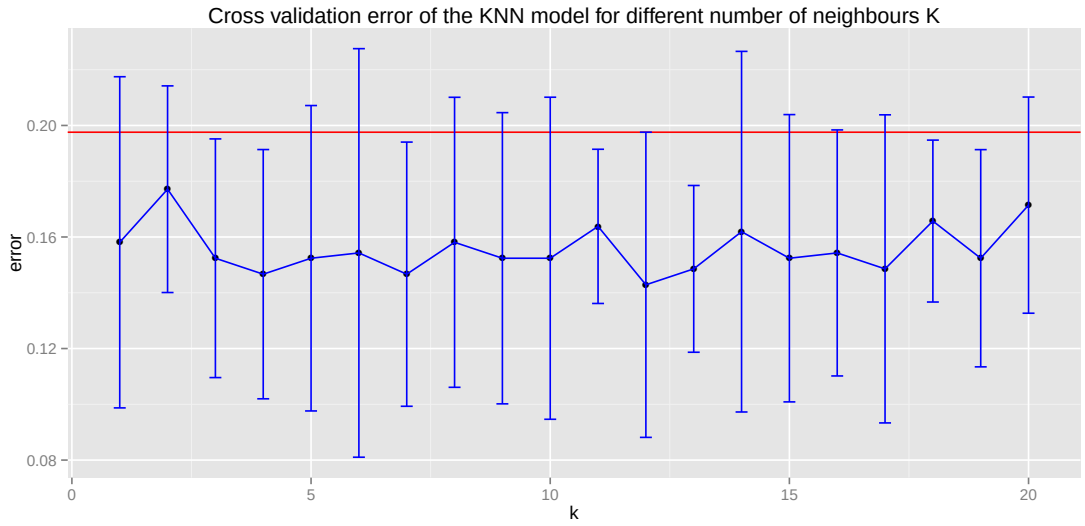


Figura C.2.1.: *Entrenamiento del* parámetro k *del modelo* KNN *para el problema del diagnóstico (subsección 3.4.1)*. En la gráfica vemos representados los valores medios del error y sus desviaciones (10CV), como criterio escogimos el menor k que su cota superior estuviera por debajo de la del k con el error medio más pequeño (línea roja).

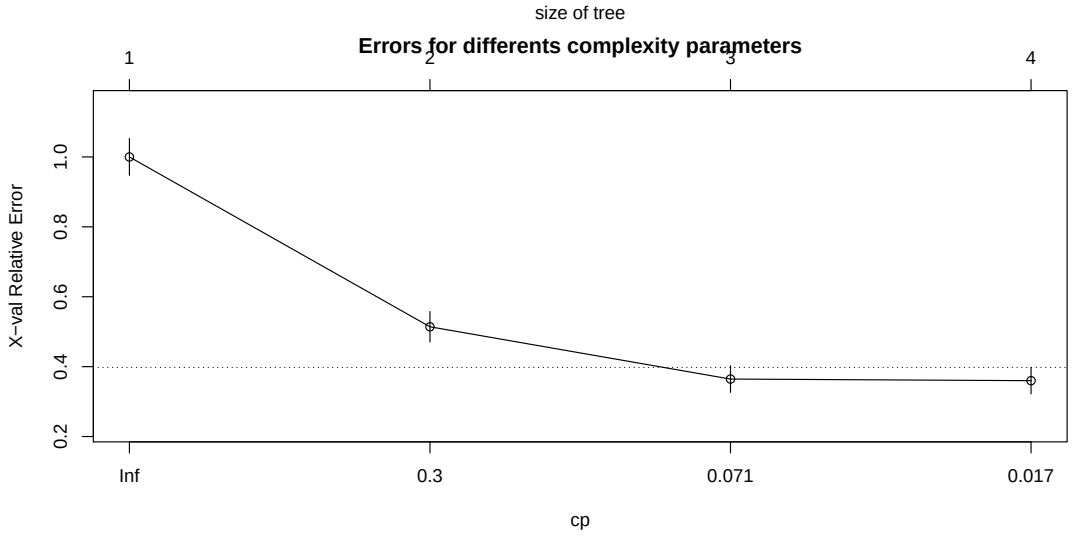


Figura C.2.2.: Selección del *parámetro de complejidad* del *árbol de decisión* para el problema del diagnóstico (subsección 3.4.3).

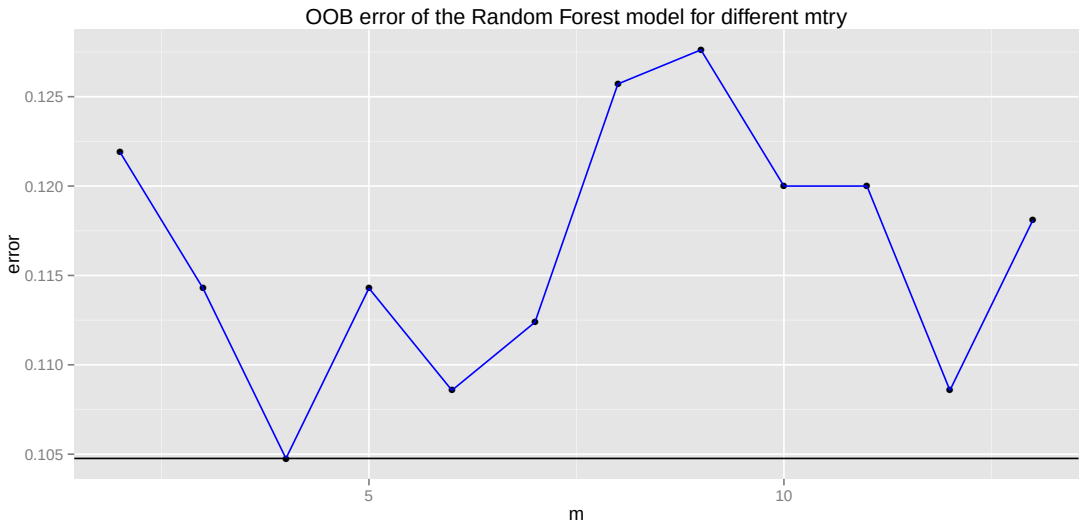


Figura C.2.3.: Entrenamiento del parámetro m en el modelo de *bosque aleatorio* en el problema del diagnóstico (subsección 3.4.4).

Figura C.2.4.: Evolución del error según el número de árboles en el modelo de *bosque aleatorio* en el problema del diagnóstico (subsección 3.4.4).

C.3. Conversión

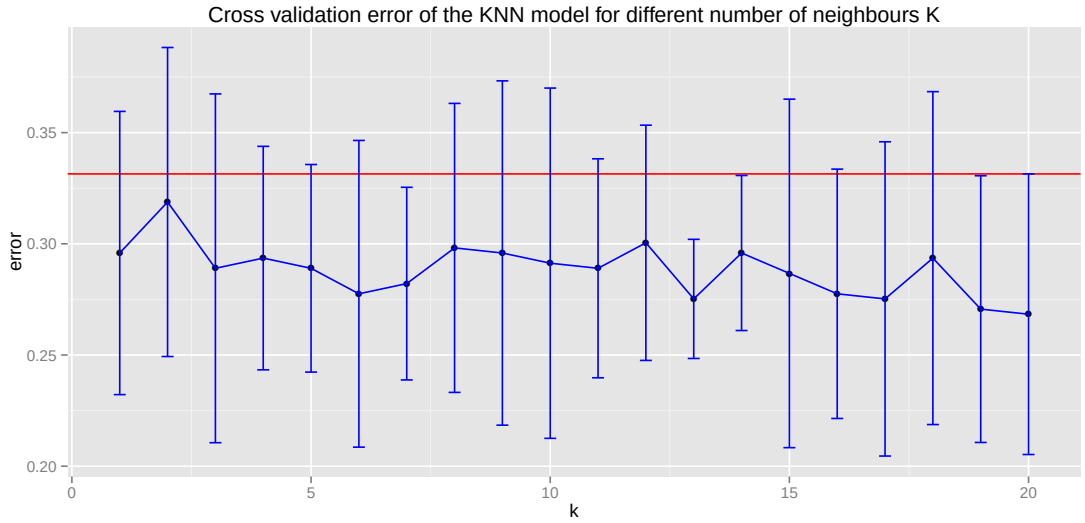


Figura C.3.1.: *Entrenamiento del* parámetro k *del modelo* KNN *para el problema de la conversión a corto plazo (subsección 3.5.1).* En la gráfica vemos representados los valores medios del error y sus desviaciones (10CV), como criterio escogimos el menor k que su cota superior estuviera por debajo de la del k con el error medio más pequeño (línea roja).

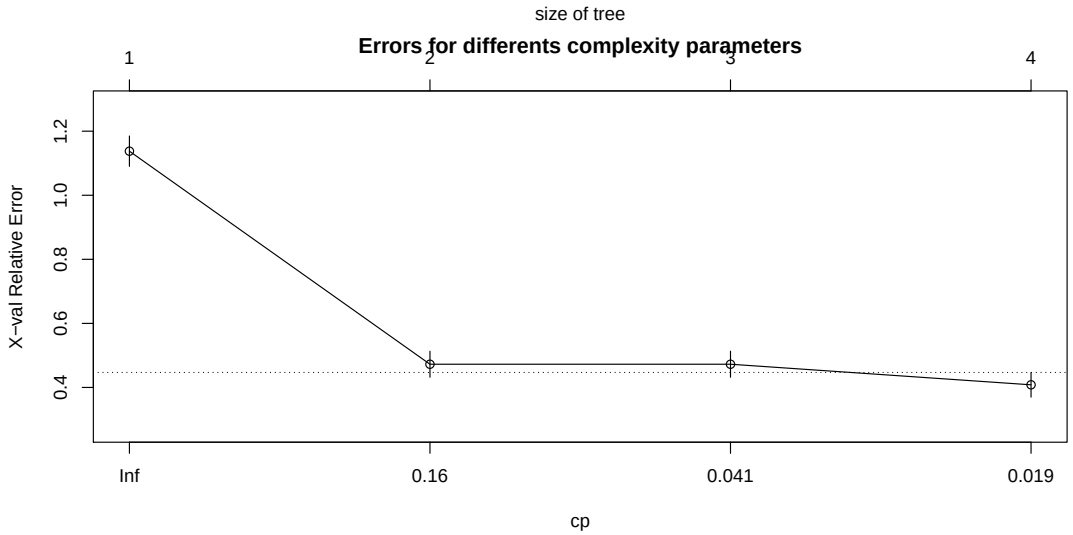


Figura C.3.2.: Selección del *parámetro de complejidad* del *árbol de decisión* para el problema de la conversión a corto plazo (subsección 3.5.3).

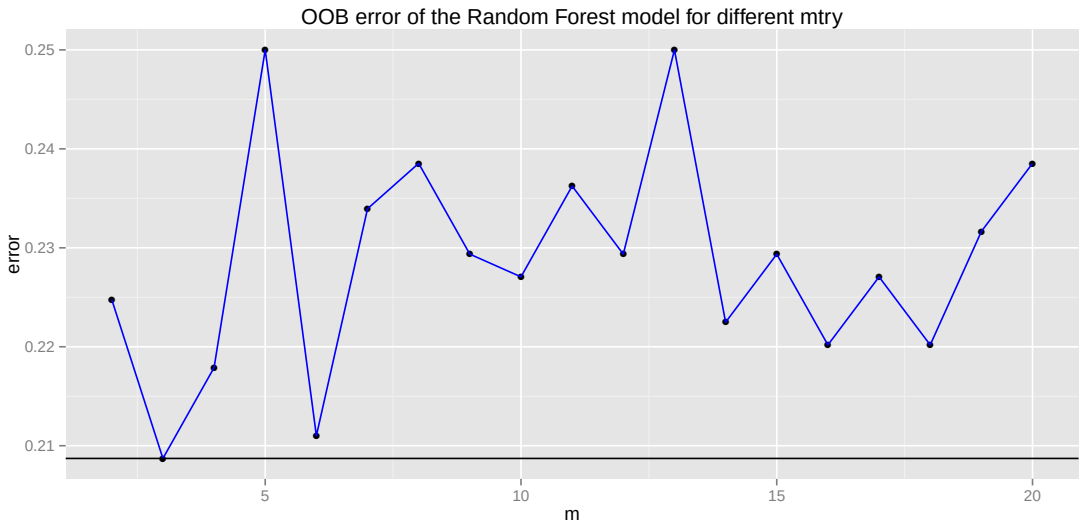


Figura C.3.3.: Entrenamiento del parámetro m en el modelo de *bosque aleatorio* en el problema de la conversión a corto plazo (subsección 3.5.4).

Figura C.3.4.: Evolución del error según el número de árboles en el modelo de *bosque aleatorio* en el problema de la conversión a corto plazo (subsección 3.5.4).