El modelo presentado en este trabajo, teóricamente pesado y con una implementación tediosa que recae en técnicas de simulación, resultó ser realmente efectivo en la práctica. A lo largo de este capítulo, se hará una exploración intuitiva y muy visual de sus capacidades. Todas las gráficas presentadas, se generaron con el mismo paquete que realiza la estimación, pues los mismos objetos que las funciones de `R` arrojan, pueden ser utilizadas para hacer gráficas que reflejan la intuición subyacente del modelo.

En particular, se simularon cinco bases de datos en dos dimensiones, es decir, se tienen dos covariables $\mathbf{X} \in \mathbb{R}^2$, con diferentes patrones para la respuesta y tanto lineales como no lineales. Esto, con el objetivo de poder *visualizar* los grupos, como se separan por medio de fronteras no lineales y para poder visualizar la función $f(\mathbf{x})$ en tres dimensiones. Posteriormente, se aplica a bases de datos reales, particularmente una de cáncer y otra financiera, donde, al aumentar la dimensionalidad no se pueden visualizar. Sin embargo, se dan una serie de resúmenes numéricos y medidas que evalúan la precisión del modelo, abriendo la discusión a limitaciones de este.

0.1. Evalución del modelo - función log-loss y matrices de confusión

Dos buenas medidas de evaluar la efectividad (y precisión) de un modelo de clasificación binaria, son las *matrices de confusión* y la función *log-loss* (LL).

Sea $\mathbf{y} = (y_1, \ldots, y_n)^t$ el vector de respuestas verdaderas; $\hat{\mathbf{p}} = (\hat{p_1}, \ldots, \hat{p_n})^t$ el vector de probabilidades ajustadas, donde $\hat{p}_i = \hat{P}_{\text{modelo}}(y_1 = 1|\mathbf{x}_i)$ es la probabilidad estimada por el modelo de que la observación y_i sea igual a 1, definiendo el vector de respuestas ajustadas $\hat{\mathbf{y}} = (\hat{y}_1, \ldots, \hat{y}_n)^t$, donde, $\hat{y}_i = 1 \iff \hat{p}_i > .5$. La función log-loss $ll : \{0, 1\}^n \times [0, 1]^n \to \mathbb{R}^+$ es:

$$ll(\mathbf{y}, \hat{\mathbf{p}}) = -\sum_{i=1}^{n} [y_i \ln(\hat{p_i}) + (1 - y_i) \ln(1 - \hat{p_i})] \tag{1}$$

La ventaja de usar la función ll, es que da una métrica que, no solo para mide que tan buena es la clasificación binaria, sino, que toma en cuenta la precisión de la predicción. Esto se debe a la función es convexa y se penaliza cuando las probabilidades ajustadas están muy lejos de la real. Asimismo, si la predicción fue incorrecta pero la probabilidad fue cercana a 0.5 no se penaliza tanto. Idealmente $ll = 0$ si se da una clasificación perfecta y conforme crezca, el modelo es peor. En la práctica y bajo un enfoque frequentista, la función LL es la que usualmente se utiliza para entrenar y comparar modelos de clasificación como redes neuronales.

El segundo método, la matriz de confusión, no es más que un método descriptivo, con base en *tablas de*

contingencia que calcula las frecuencias para aciertos y errores, separando en grupos. Esto es:

	$\hat{y} = 0$	$\hat{y} = 1$	
$y = 0$	#0's ✓	#0's clasificados como 1	# de observaciones cero
$y = 1$	#1's clasificados como 0	#1's ✓	# de observaciones uno
	# de estimados cero	# de estimados uno	Total de obs. $= n$

Matriz de confusión

De donde se puede ver la exactitud en las predicciones del modelo. Estos dos métodos, serán los usados para evaluar cada ejemplo.

0.2. Análisis a fondo de una modelo sencillo

El primer ejemplo que se analizará, es un ejemplo muy sencillo, que busca ejemplificar cada componente del modelo. Se simularon un total de $n = 350$ observaciones separadas en dos grupos, cada uno con tamaños $n_0 = 200$ y $n_1 = 150$ respectivamente ($n = n_0 + n_1$). Los puntos se muestraron de distribuciones normales bivariadas, esto es:

$$\text{Grupo 0: } \mathbf{x}_i \sim N_2\left(\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \middle| \mu_0 = \begin{pmatrix} 2 \\ 2 \end{pmatrix}, \Sigma_0 = \begin{pmatrix} 0.25 & 0.35 \\ 0.35 & 1 \end{pmatrix}\right) \qquad i = 1, \ldots, 200$$

$$\text{Grupo 1: } \mathbf{x}_i \sim N_2\left(\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \middle| \mu_1 = \begin{pmatrix} 4 \\ 4 \end{pmatrix}, \Sigma_1 = \begin{pmatrix} 1 & .24 \\ .24 & .64 \end{pmatrix}\right) \qquad i = 201, \ldots, 350$$

Las medias μ se toman diferentes para dar una clara separación y las covarianzas corresponden a las correlaciones $\rho_0 = 0.7$ y $\rho_1 = .3$ respectivamente. Codificando el grupo 0 con $y = 0$ de color rojo y el grupo 1 con $y = 1$ de color azul. Se tienen los datos presentados en la Figura 1[1]. Los parámetros se escogieron con un proceso de *prueba y error* para dar estructura pero a la vez separación en el espacio de covariables $\mathcal{X}^2 \approx [0.3, 7.5] \times [-0.5, 5.9]$. Esto, para que se tuviera una pequeña región donde las distribuciones se traslaparan y exista cierto grado de confusión. El objetivo del modelo, es poder hacer una separación de estas dos regiones sin sobreajustar; identificando, a grandes rasgos, dónde se encuentran los puntos rojos y dónde se encuentran los puntos azules.

1. Vale la pena mencionar, que todas las gráficas y presentadas en este Capítulo, fueron generadas usando las capacidades de la librería *ggplot2*, donde se incorporó su funcionalidad al paquete *bpwpm* para poder generar estos gráficos de una forma fácil y rápida para este tipo de modelos.

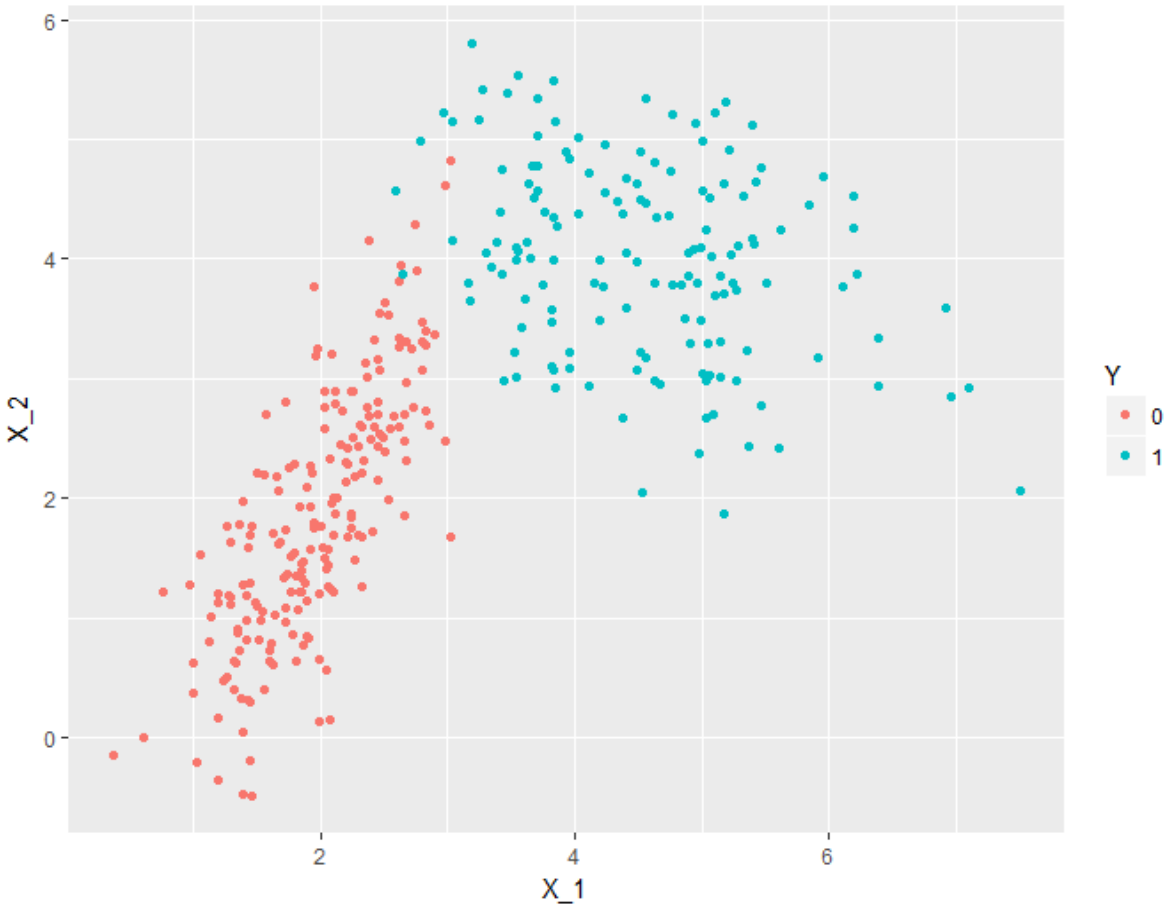


Figura 1: **Primer ejemplo**. Poco traslape entre puntos

Modelo probit frecuentista para comparar

En un modelo tradicional, solo se puede hacer una separación lineal. Para comparar, se corrió el siguiente modelo probit frecuentista en `R`, usando la función `glm(..., family = binomial(link = 'probit'))`:

$$\begin{aligned} p_i = P(y_i = 1) = \mathbb{E}[y|\mathbf{x}_i] = \Phi(f(\mathbf{x}_i)) \quad &\Rightarrow \\ \Phi^{-1}(p_i) = \beta_0 + \beta_1 x_{i,1} + \beta_2 x_{i,2} \qquad \forall i = 1, \ldots, n \end{aligned} \tag{2}$$

De donde se obtuvieron los siguientes resultados:

Parámetro	Valor Estimado
$\hat{\beta}_0$	-17.29
$\hat{\beta}_1$	4.43
$\hat{\beta}_2$	1.08
Métricas	**Valor**
ll	0.0399

	$\hat{y} = 0$	$\hat{y} = 1$	
$y = 0$	198	2	200
$y = 1$	2	148	150
	200	150	350

Dada la simplicidad de los datos, el modelo lineal probit, presentado en la ecuación (2) resulta ser una excelente forma de hacer la clasificación. Como se ve en la Figura: 2, los datos son fácilmente separables por una linea recta que cruza exactamente donde se empiezan a traslapar. Únicamente, existen 4 datos que quedan mal clasificados, pero que, dadas sus coordenadas, parecerían pertenecer a los grupos opuestos y se consideran como datos atípicos. El modelo presenta una precisión de 98.85 %, todos los parámetros fueron significativos[2] y se tiene un valor de la función log-loss muy bajo. Por lo tanto, se puede concluir que el modelo probit tradicional, es un muy buen modelo para este conjunto de datos.

Figura 2: Separación de los grupos por medio de un modelo probit lineal frecuentista

Ejemplo 1 - modelo bpwpm

Ahora, se prueba contra el modelo *bpwpm* de este trabajo. Se esperaría que, al menos, se comportara como el probit anterior, pues, dada la estructura tan robusta que se tiene, este podría colapsar en uno lineal.

Como un primer ejemplo sencillo de comprender, se corren un modelo donde los polinomios por partes son rectas disjuntas en cada segmento con un solo nodo. Se realizan mil simulaciones del muestreo Gibbs, de donde se descartan las primeras 500 observaciones y posteriormente se toma cada segunda observación. Estas especificaciones se resumen en la siguiente tabla que se presentará antes de cada modelo.

2. Usando las pruebas t clásicas de modelos lineales frequentistas.

Parámetros		Parámetro Sim.
$M = 2$	$N^* = 4$	$N_{\text{sim}} = 1000$
$J = 2$	$d = 2$	$k^* = 500$
$K = 0$	$n = 350$	$k_{\text{thin}} = 2$

Ejemplo 1

Dado que este es un modelo extremadamente sencillo y que se tiene un número pequeño de bases para los polinomios, $N^* = 4$, se presenta por única ocasión, la expansión completa para poderla comparar contra el modelo anterior (2). Esto es[3]:

$$
\begin{aligned}
p_i = P(y_i = 1) = \mathbb{E}[y|\mathbf{x}_i] = \Phi(f(\mathbf{x}_i)) \quad &\Rightarrow \\
\Phi^{-1}(p_i) = \beta_0 + \beta_1 f_1(x_{i,1}) + \beta_2 f_2(x_{i,2}) \qquad &\forall i = 1, n, \ldots, \qquad (3) \\
= \beta_0 \\
+ \beta_1[w_{1,1} + w_{2,1}x_{i,1} + w_{3,1} + w_{4,1}(x_{i,1} - \tau_{1,1})_+] \qquad &(4) \\
+ \beta_2[w_{1,2} + w_{2,2}x_{i,2} + w_{3,2} + w_{4,2}(x_{i,2} - \tau_{1,2})_+] \qquad &(5)
\end{aligned}
$$

Contrastando (2) contra (3), se puede ver la introducción del componente no lineal a través de las f_j, desglosadas en (4) y (5), para un total de 11 parámetros. Las expansiones truncadas de polinomios son relativamente sencillas y se ve claramente que se les da estructura de recta, permitiendo discontinuidades entre ellas pues $(\cdot)_+$ es una función por partes que se activa si $x_{i,j}$ es mayor que el nodo $\tau\cdot, j$. Aunque esta expansión es aparatosa, el modelo logra hacer excelentes predicciones en cuanto a las regiones. Cabe mencionar que, dado este es un ejemplo introductorio con un número relativamente bajo de observaciones, la estimación de los parámetros, se realizó *dentro de la muestra*[4], esto quiere decir, que el modelo se entrena con las mismas observaciones contra las que se busca predecir[5]. Previo al análisis de convergencia de las cadenas, se hace una exploración preliminar para explicar todos los detalles del modelo. Usando la función

3. Para $w_{l,j}$, se usa la convención de subíndices $l = 1, \ldots, N^*$ de la biyección de la Tabla **??** y $j = 1, \ldots, , d$ para indicar la dimensión

4. *in-sample*

5. El efecto que esto podría tener es que se sobreajuste o se hagan predicciones demasiado acertadas, sin embargo, es normal hacerlo para este tipo de modelos dado n. Además, por lo pronto el objetivo final es dar predicciones a través de las regiones formadas y no tanto para observaciones nuevas. De cualquier forma, se podría hacer separando, antes del análisis, la base de datos en dos, una para entrenar el modelo y otra para probarlo.

de perdida cuadrática, se obtienen los siguientes resultados:

Info. predicción	
Est. Puntual	Media posterior
Precisión	98.85 %
log-loss	0.03702

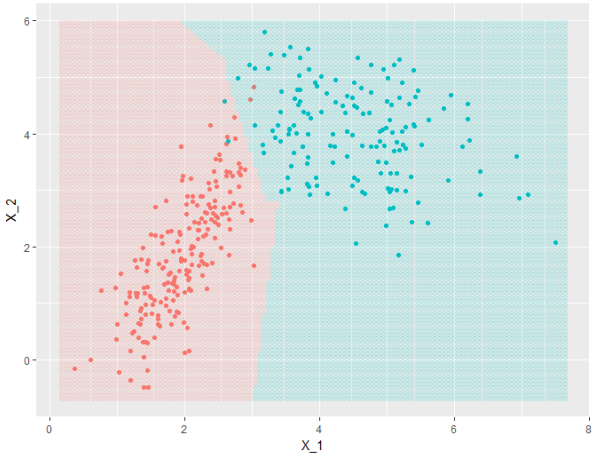
$\hat{\beta}$	
$\hat{\beta}_0$	-0.79
$\hat{\beta}_1$	3.35
$\hat{\beta}_2$	0.65

$\hat{\mathbf{w}}$	
-0.52	-1.48
0.39	-0.38
0.09	0.66
1.05	1.32

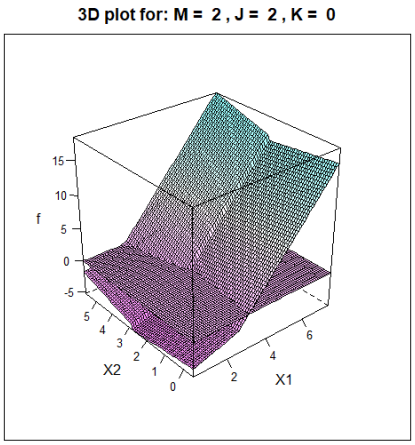
	$\hat{y}=0$	$\hat{y}=1$	
$y=0$	198	2	200
$y=1$	2	148	150
	200	150	350

Numéricamente, el modelo se ve bien; la matriz de confusión es idéntica a la del probit anterior y, por ende, la precisión. Sin embargo, se tiene un valor de la función log-loss un poco más bajo, indicando que se tiene un mejor modelo una vez consideradas las probabilidades ajustadas $\hat{\mathbf{p}}$. Sin embargo, a diferencia del modelo anterior, los parámetros estimados $\hat{\beta}$ y $\hat{\mathbf{w}}$ no se pueden interpretar de la misma manera. En modelos de ML, debido a la complejidad, es mejor tratar a los parámetros como partes funcionales del modelo, que como números con significado. Sin embargo en especial el vector $\hat{\beta}$ puede considerarse como los *pesos* que se le dan a cada transformación no lineal, y su magnitud corresponde a que tanta fuerza tiene esa dimensión en el modelo. De este ejemplo se ve claramente que la primer dimensión, es con la que mejor se están explicando los datos.

La Figura 3a, es clave para entender el modelo, en ella se presentan las dos regiones (de colores) que el modelo detecta para hacer las predicciones. A diferencia de los modelos lineales donde la frontera de



(a) Regiones de predicción para modelo con $M = 2$, $J = 2$ y $K = 0$

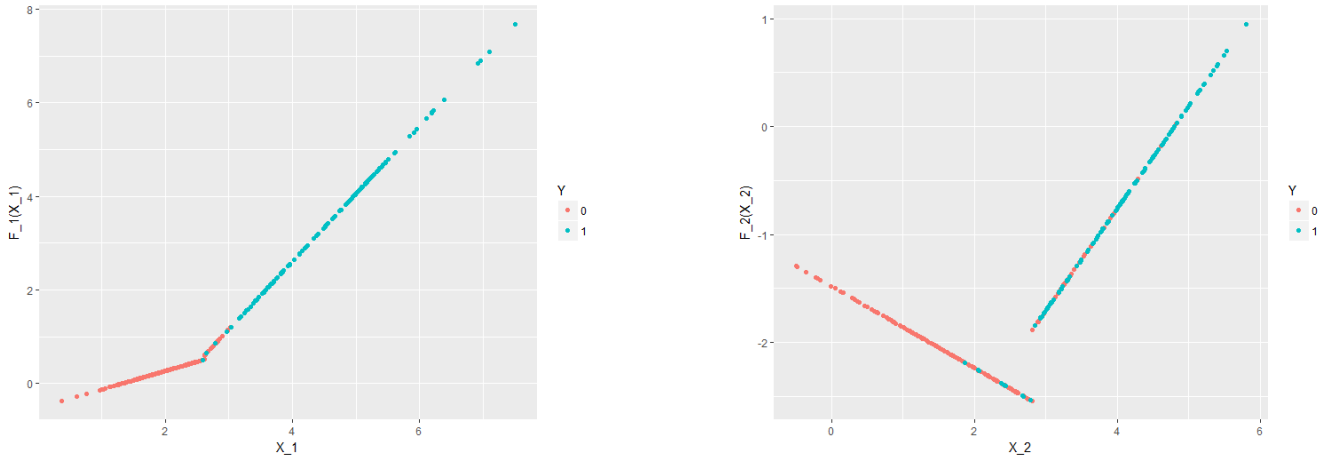


(b) Representación en 3D de $\hat{f}$

Figura 3: Visualización de los resultados del modelo

predicción binaria es, por ende, lineal, el modelo presentado en este trabajo permite tener fronteras tan complejas como se quiera (aunque no siempre necesarias). Para este ejemplo en particular, la frontera refleja que se está usando un solo nodo y polinomios lineales discontinuos, es por ello que se da la rugosidad.

Encontrar la frontera como tal, resulta una tarea mucho más complicada, pues correspondería a resolver la ecuación $\hat{f}(\mathbf{x}) \equiv 0$; en los GLM, esta frontera es perfectamente lineal y el despeje se puede hacer. Sin embargo, cuando se tienen modelos no lineales, se puede hacer una proyección de ella pues, recordando, lo que interesa es discernir cuando la función $\hat{f}$ sea positiva o negativa. Gracias al hecho que $d = 2$ para este ejemplo, se puede visualizar $\hat{f}(\mathbf{x})$ en la Figura 3b. En esta gráfica, se marca con un plano el *corte* cuando $\hat{f}(\mathbf{x})$ se vuele positiva. Además. se detectan los *pliegues* de discontinuidades derivados del nodo y de la especificación en los parámetros M, J y K. Se hace notar, que esta representación, no es más que la suma ponderada (por $\hat{\beta}$) de las transformaciones no lineales $f_j \quad j = 1, 2$, por lo cual vale la pena visualizarlas en la Figura 4.



(a) $\hat{f}_1(x_1)$ (b) $\hat{f}_2(x_2)$

Figura 4: Transformaciones no lineales para cada dimensión.

Aunque la escala vertical de f_j es arbitraria, pues en realidad están ajustando a los residuales parciales, están cumpliendo su propósito de detectar y capturar el patrón que deriva en la separación de los grupos. Para valores izquierdos de ambas variables, se tiene el grupo rojo, para valores mayores, se tiene el grupo azul. Ese efecto, es capturado en el salto que dan las rectas en el nodo, mediante una mayor pendiente en las rectas del lado derecho (mayoritariamente azules), pues, al ser más positivas conforme se avanza en el rango de x, estas tendrán más peso en la función de proyección $\hat{f}$, volviéndola más positiva y por ende, más probable que esa región contenga observaciones del grupo azul.

Con estas gráficas, se espera dar claridad a todos los componentes del modelo. Este ejemplo trivial, donde $d = 2$, tiene la ventaja que todo es visualizable gráficamente. Se hace énfasis en que muchas de las variables presentadas o métodos, son puramente estructurales y que cumplen una función meramente técnica en el algoritmo, como lo son las variables auxiliares $\mathbf{z}$. Al final, y como se presentó en el la Figura **??**, el modelo

tiene cierta coherencia y simplicidad fácilmente representable por regiones de dos colores. Posteriormente, se verá la flexibilidad de estas regiones, ahí donde recae la fuerza del modelo.

Aunque las funciones f_j sean relativamente sencillas presentadas en una dimensión como en la Figura 4, una vez colapsadas en la función de proyección f, se pueden tener efectos inesperados o relativamente extraños. Esto se debe a que la interacción entre los nodos en más de una dimensión puede ser complicada de visualizar y al juntar todo, se dan efectos como los pliegues de la figura 3b. Para solucionar esto, usualmente se pide cierta suavidad en los polinomios por partes haciéndolos splines para que f sea también suave. Sin embargo, dependiendo de la aplicación los parámetros M, J y K se van calibrando hasta que el modelo sea aceptable y las cadenas hayan convergido.

0.2.1. Diferentes tipos de fronteras - análisis de sensibilidad

Toda la flexibilidad del modelo, depende de los parámetros M, J y K que recaen en las manos del estadista. Estos parámetros controlan la suavidad de la frontera y es importante que se entienda como cada uno afecta la estimación de los polinomios. Por lo tanto y para pasar de esta base de datos sencilla a algunas más retadoras, se juega un poco con ellos para ver sus efectos en el modelo. Los parámetros se irán aumentando poco a poco, conforme

Ejemplo 2 - polinomios constantes

A principio, mientras se desarrollaba el paquete, parecía intuitivo que, si se está construyendo una clasificación binaria, a cada observación bidimensional $\mathbf{x}_i = (x_{i,1}, x_{i,2})^t$ se transformara en una especie de *observación de diseño*[6], la cual codificaría si su correspondiente respuesta y_i era 0 o 1. Esta linea de pensamiento, llevo a pensar que escoger los polinomios por partes como constantes sería la mejor opción para la predicción y separación de las regiones. Por lo tanto, se corre el modelo con los parámetros:

Parámetros		Parámetro Sim.
$M = 1$	$N^* = 2$	$N_{\text{sim}} = 1000$
$J = 2$	$d = 2$	$k^* = 500$
$K = 0$	$n = 350$	$k_{\text{thin}} = 2$

Ejemplo 2

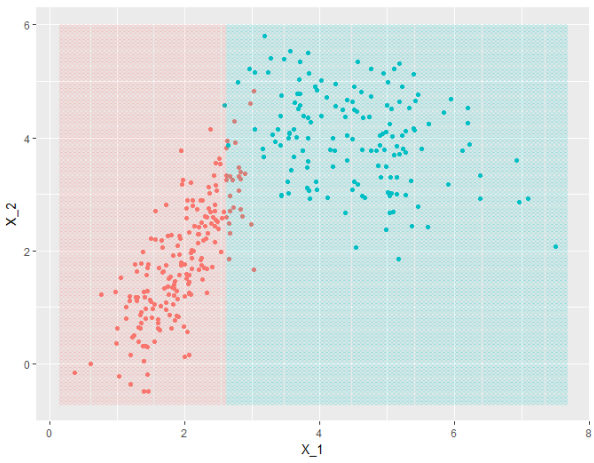
6. Derivado de las *matrices de diseño* que se construyen para los modelos ANOVA; si se pudiera mapear cada observación a un espacio más sencillo codificando la respuesta, se podría hacer una predicción aún mejor.

De donde se obtiene los siguientes resultados[7]:

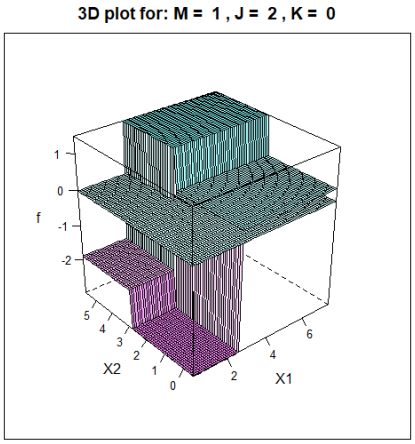
Info. predicción	
Est. Puntual	Media posterior
Precisión	92.3 %
log-loss	0.2081

	$\hat{y}=0$	$\hat{y}=1$	
$y=0$	174	26	200
$y=1$	1	149	150
	175	175	350

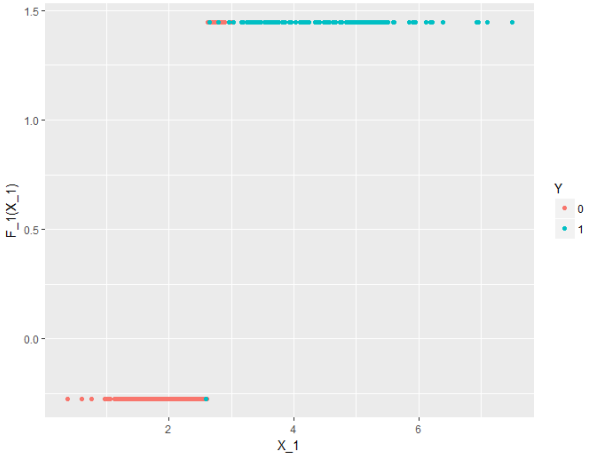
Claramente, esta no es una mejora al modelo. Tanto en precisión como en métrica log-loss el modelo empeoró significativamente. En las Figuras 5, se visualiza el porqué. Al tener únicamente 1 nodo y polinomios por partes constantes, la región de predicción es la más sencilla posible: dos planos que separan



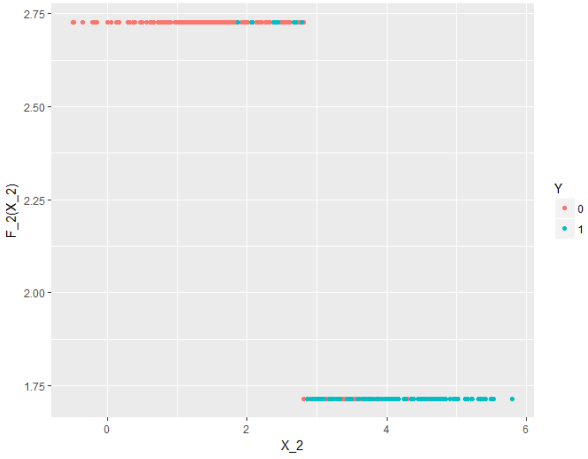
(a) Frontera



(b) Representación 3D de $\hat{f}$



(c) $\hat{f}_1(x_1)$



(d) $\hat{f}_2(x_2)$

Figura 5: Ejemplo 2, con $M=1$, $J=2$, $K=0$, derivando en una función escalonada

7. De ahora en adelante, hasta llegar al análisis de convergencia, dado que la complejidad de los modelos comenzará a aumentar, se opta por no mostrar los estimadores $\hat{\beta}$ y $\hat{\mathbf{w}}$ pues no aportan nada a la discusión y son dificilmente interpretables. Se opta mejor por presentar las gráficas. Si existe alguno digno de mencionar se hará referencia directamente a él.

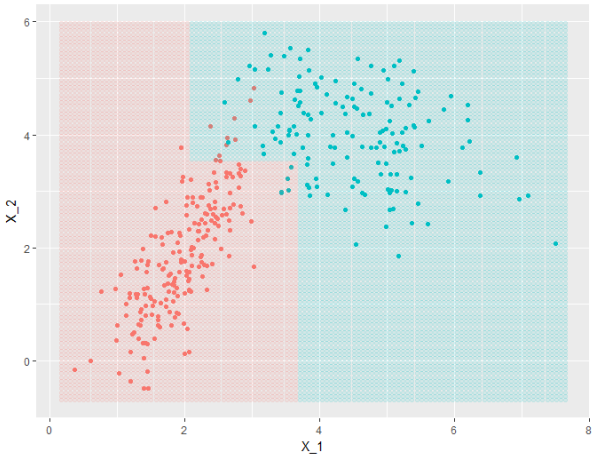
las regiones rojas y azules. En la imagen 5b, se visualiza la función escalonada resultante con 4 *mesetas* producto de las interacciones entre los dos nodos. Se hace notar que la región inferior izquierda es más negativa pues se tiene menor probabilidad de ser del grupo azul. En las imágenes 5c y 5d se ven estos polinomios constantes que logran lo que se buscaba, esto es codificar de forma más sencilla la variable real $\mathbf{x}$, sin embargo, al considerar las interacciones entre los nodos, la intuición pierde. Se hace notar que en la Figura 5d los polinomios están invertidos, esto se debe a que β_2 es negativo, por lo tanto, a mayores valores de $\hat{f}_2$,la f global aumenta. Lo cual indica que los parámetros están captando bien los patrones.

Como un tercer ejemplo, se podría pensar que aumentando el número de nodos y dejando todo lo demás constante mejora la predicción, sin embargo, lo hace de forma marginal. Tomando $J = 3$ (equivalente a dos nodos) se obtiene una precisión del 95.7 %, en las Figuras **??** se presentan los resultados visuales. Se hace notar, que en este caso en especifico, si la posición del segundo nodo en x_1 fuera ligeramente menor (más cercano a $x_1 = 3$) la región mejoraría. De igual forma, se hace notar que se tienen $J^2 = 9$ mesetas en la representación 3D y que, en las regiones de confusión, las f_j tienen valores más cercanos a cero, identificando esta incertidumbre. Finalmente, seguir aumentando el número de nodos, no mejora sustancialmente la estimación. Por lo tanto, lo que se debe de hacer es romper la linealidad aumentando M.

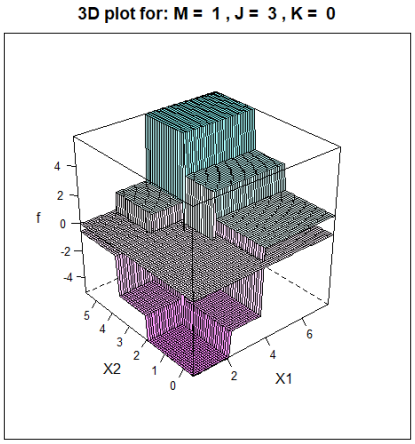
Ejemplo 4 - linealidad + continuidad

Para el primer ejemplo de la Sección 0.2, se usaron los parámetros, $M = 2$, $J = 2$ y $K = 0$. Esto corresponde a expansiones polinómicas lineales pero discontinuas. Parecería contraintuitivo usar polinomios continuos en los nodos pensando en que se buscan predicciones binarias, sin embargo, resulta que al aumentar la *suavidad* en los polinomios, se logra al menos igualar el nivel de precisión para este ejemplo. En otras bases de datos se verá que esta suavidad es inclusive necesaria para dar mejores predicciones. Asimismo, se ve empiricamente, que al aumentar el numero de parámetros y la suavidad, se tienen estimaciones más robustas que convergen mejor a distribuciones estacionarias.

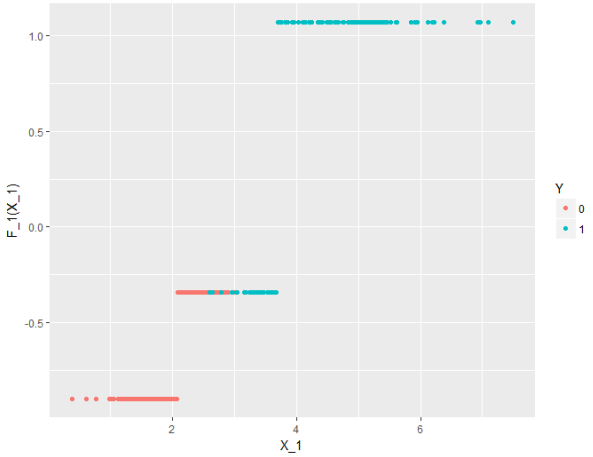
Para este nuevo ejemplo, se aumenta $K = 1$ haciendo que la doble suma en la expansión de bases de **??** de la página **??** se desvanezca regresando a la definición de splines lineales. Se tiene el modelo:



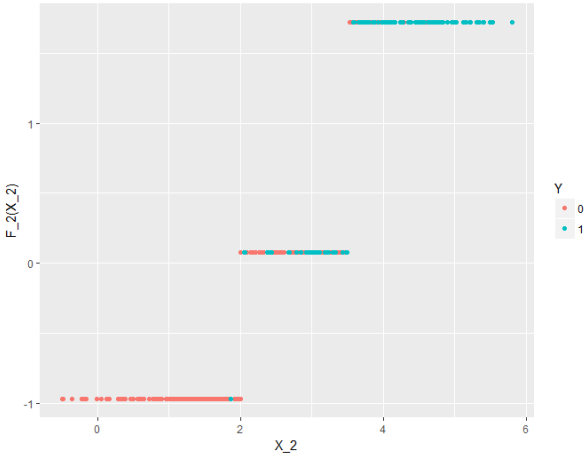
(a) Frontera



(b) Representación 3D de $\hat{f}$



(c) $\hat{f}_1(x_1)$



(d) $\hat{f}_2(x_2)$

Figura 6: Ejemplo 3, con $M = 1$, $J = 3$, $K = 0$, derivando en una función escalonada

Parámetros		Parámetro Sim.
$M = 2$	$N^* = 3$	$N_{\text{sim}} = 1000$
$J = 2$	$d = 2$	$k^* = 500$
$K = 1$	$n = 350$	$k_{\text{thin}} = 2$

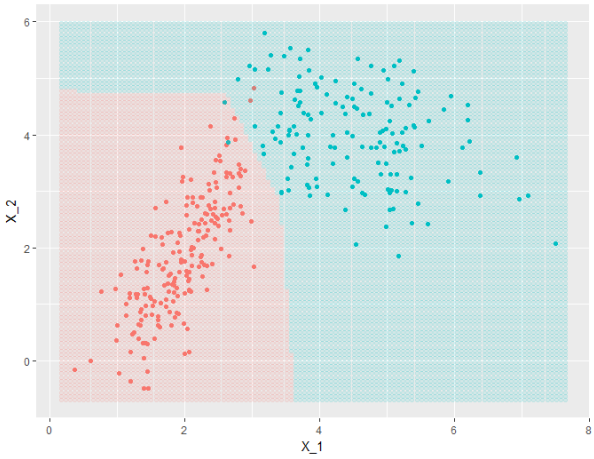
Ejemplo 4

Con resultados:

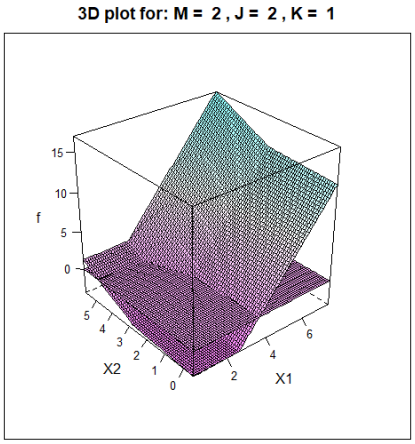
Info. predicción	
Est. Puntual	Media posterior
Precisión	98.85 %
log-loss	0.04109

	$\hat{y}=0$	$\hat{y}=1$	
$y=0$	198	2	200
$y=1$	2	148	150
	200	150	350

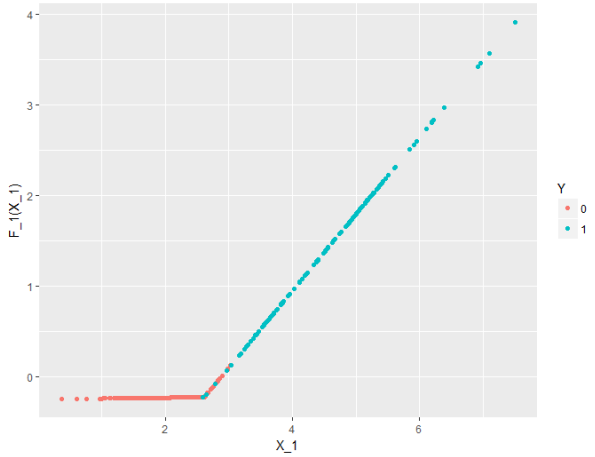
igual de buenos que en el primer ejemplo, y un valor de log-loss marginalmente mayor. Una vez más, se presentan las cuatro imágenes para evaluarlo en la Figura 7. Otra característica contraintuitiva de este modelo en particular, es que se tiene un parámetro menos para cada expansión de bases, ahora $N^* = 3$, pues al añadir la restricción de continuidad por nodo se elimina uno de los dos términos independientes en las expansiones de las ecuaciones (4) y (5). Con estas imágenes, es claro ver que se está buscando



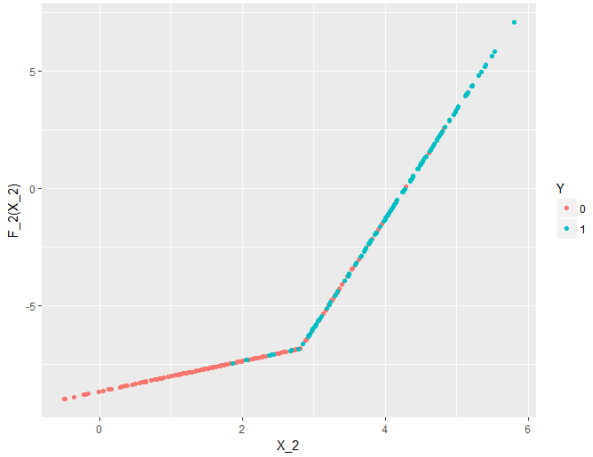
(a) Frontera



(b) Representación 3D de $\hat{f}$



(c) $\hat{f}_1(x_1)$



(d) $\hat{f}_2(x_2)$

Figura 7: Ejemplo 4, con $M = 2$, $J = 2$, $K = 1$, el modelo lineal continuo

mayor suavidad progresivamente. En 7b, se ve que la *sabana* ya no da brincos discontinuos y, aunque

aún no sea suave, si retiene la estructura de predicción. Esta frontera se ve mejor en 7a, donde logra detectar perfectamente bien la región problemática. Finalmente, de 7c y 7d, se pueden apreciar las rectas ya continuas. Analizándolas, se ve claramente que existe muy poca confusión en cuanto a la primera región roja por lo que $\hat{f}_1(x_1)$ es plana al principio y después da un salto grande. Pasa lo mismo con $\hat{f}_2(x_2)$, sin embargo, la escala es un poco diferente, ya que $\hat{\beta}_2$ es cercana a cero, confirmando nuestra creencia de que la región se puede separar de excelente forma casi, únicamente, con la información de x_1. Aunque no se implementó, técnicas de *selección de variables* también podrían ser aplicadas a este tipo de modelos. La precisión mejoró una vez más regresando al modelo óptimo. Asimismo, aumentar el número de nodos no mejora sustancialmente las cosas y solo agrega muchos más parámetros por estimar.

Ejemplo 5 y 6 - polinomios de orden mayor

Para cerrar las pruebas con esta base de datos y empezar a probarlo con regiones más interesantes, se corren dos últimos modelos con polinomios de orden mayor. En particular para el ejemplo 5, se usan parábolas discontinuas con 2 nodos para ver las capacidades del modelo. Finalmente para el ejemplo 6 se usan los famosos splines cúbicos para ver que tan suave puede llegar a ser el modelo. Esto es:

Parámetros		Parámetro Sim.
$M = 3$	$N^* = 9$	$N_{\text{sim}} = 1000$
$J = 3$	$d = 2$	$k^* = 500$
$K = 0$	$n = 350$	$k_{\text{thin}} = 2$

Ejemplo 5

Parámetros		Parámetro Sim.
$M = 4$	$N^* = 6$	$N_{\text{sim}} = 2000$
$J = 3$	$d = 2$	$k^* = 1000$
$K = 3$	$n = 350$	$k_{\text{thin}} = 3$

Ejemplo 6

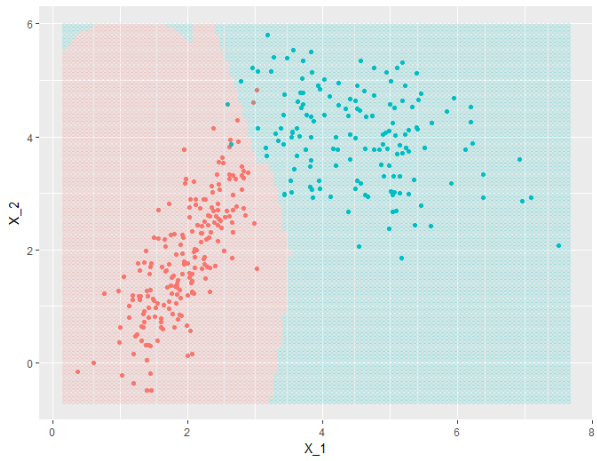
Se hace notar, que para el ejemplo 6 se corre una cadena relativamente más larga y con periodo de *burn-in* k^* también mayor. Esto se debe a el ejemplo 6 es el elegido para analizar su convergencia en la Sección 0.2.2 ya que se dan resultados interesantes.

Para el ejemplo 5, se obtuvo lo siguiente:

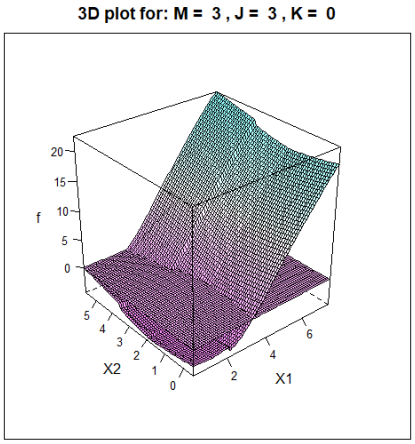
Info. predicción	
Est. Puntual	Media posterior
Precisión	98.85 %
log-loss	0.03189

	$\hat{y} = 0$	$\hat{y} = 1$	
$y = 0$	198	2	200
$y = 1$	2	148	150
	200	150	350

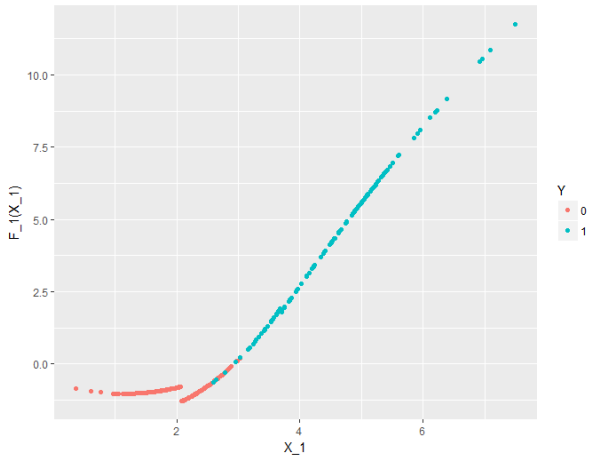
con las imágenes presentadas en la Figura 8. De donde lo único diferente son la forma de las $\hat{f}_j$. La Figura 8c, presenta un comportamiento interesante; se conserva algo de la continuidad, casi como si las



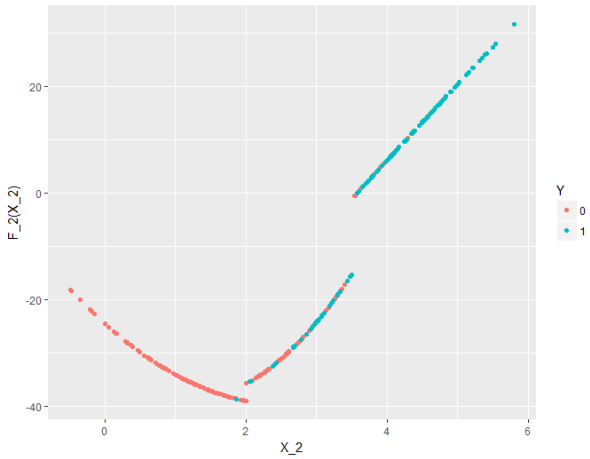
(a) Frontera



(b) Representación 3D de $\hat{f}$



(c) $\hat{f}_1(x_1)$

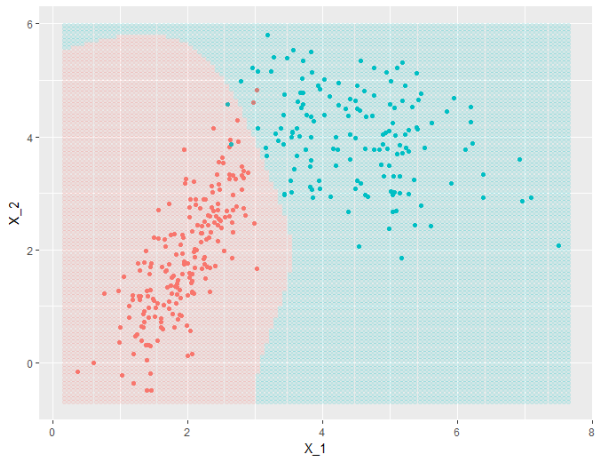


(d) $\hat{f}_2(x_2)$

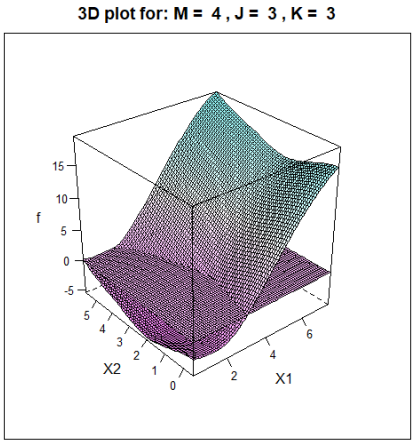
Figura 8: Ejemplo 5, con $M = 3$, $J = 3$, $K = 0$, el modelo lineal continuo

parábolas *quisieran* ser continuas pues esa es la mejor opción para hacer la predicción. Asimismo 8b y 8a presentan claramente los efectos de las discontinuidades derivados de elegir $K = 0$. Este ejemplo, funciona mejor como un referente de las capacidades del modelo preservando la precisión, en la práctica tener 21 parámetros para un ejemplo tan sencillo no es nada útil.

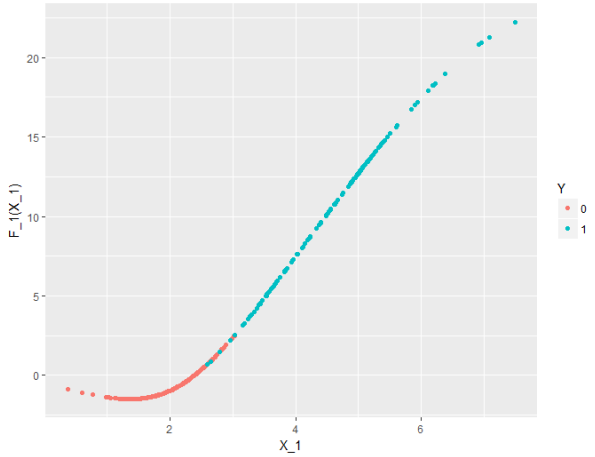
Finalmente, el ejemplo 6, es más de lo mismo, los resultados son prácticamente iguales que los obtenidos con los demás modelos. Sin embargo, se presentan las imágenes en la Figura 6 donde se puede ver claramente la *suavidad* obtenida en la frontera, en la función $\hat{f}$ y las correspondientes $\hat{f}_j$. Al tener una cadena más larga, la escala de las $\hat{f}_j$, empieza a ser relativamente arbitraria, sobre todo para la segunda pues, conforme converge el modelo los parámetros $\mathbf{w}$ compensan la escala de β y viceversa. Se continúa con este ejemplo en la siguiente sección.



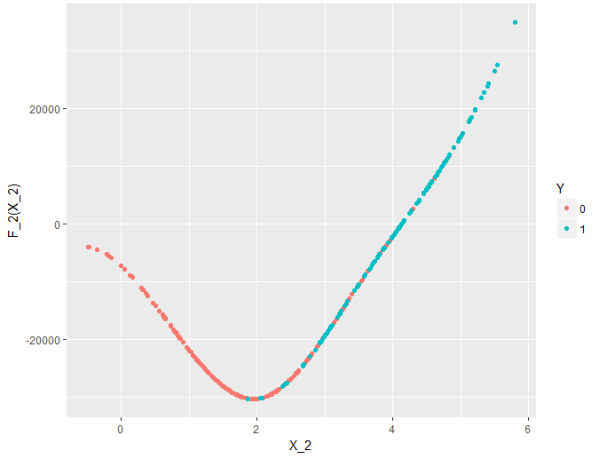
(a) Frontera



(b) Representación 3D de $\hat{f}$



(c) $\hat{f}_1(x_1)$



(d) $\hat{f}_2(x_2)$

Figura 9: Ejemplo 6, con $M = 4$, $J = 3$, $K = 3$, el modelo lineal continuo

En la practica, escoger el *mejor* modelo es subjetivo pues depende del proceso de calibración de los parámetros y los resultados que se busquen obtener. Dado que el algoritmo es realmente rápido, sobre todo para d pequeñas, se puede jugar mucho con el, y explorar una serie de modelos. En casi todos los ejemplos presentados en esta base de datos se obtuvo la precisión del 98.85 % y sería inverosímil tratar de mejorarla pues obligaría al modelo a sobreajustar y hacer regiones pequeñas para los 4 puntos que quedan en regiones de predicción opuestas. Asimismo, el número de nodos, al menos en este ejemplo, parece no importar mucho, aumentarlo, únicamente aumenta la complejidad del modelo y no aporta mucho, se ve claramente que con pocos parámetros y nodos, se tiene excelentes resultados.

Como comentario final, se hace notar que dado el algoritmo es *estocástico* y depende de la simulación de parámetros aleatorios, replicar exactamente las cadenas es una tarea casi imposible; mas, los resultados y las regiones son consistentes. Al menos para este ejemplo, los valores de k^* y k_{thin} también parecieran

no importar mucho pues las cadenas convergen muy rápido por las propiedades del Gibbs Sampler que se usa.

0.2.2. Análisis de Convergencias

El ejemplo 6 es interesante, pues ya se había explorado, sin saberlo, cuando se hablo de mustreo Gibbs en la sección **??**. En las imágenes de la página **??** se aprecian las trazas y los histogramas de los parámetros β derivados de la simulación del modelo. Analizando la imagen **??**, a primera vista, se observan varias cosas; la linea azul, que representa el parámetro $\hat{\beta}_2$ es prácticamente cero, la linea verde del parámetro $\hat{\beta}_1$ es muy consistente, casi sin variar, y la traza de $\hat{\beta}_0$ sube y baja sin aparente patrón. La imagen **??** únicamente confirma esta creencia.

Este es el inicio de un análisis de convergencia que se debe realizar independientemente de los resultados tan positivos que se mostraron anteriormente. Pues, se recuerda, que estos fueron derivados de estimaciones puntuales para los parámetros usando la media posterior, además de tomar en cuenta el periodo de calentamiento y adelgazamiento, dados por los parámetros k^* y k_thin. Asimismo, este proceso se debe realizar para todos los parámetros del modelo, es decir, los vectores $w_jj = 1, 2$. Por lo tanto, en la tabla 2 se presentan métricas interesantes para todos los parámetros del modelo, una vez *corregidas*[8] las cadenas.

Una manera usual de realizar esto es revisando la *media ergódica* que no es más que el promedio móvil de las trazas.

0.3. Otros resultados interesantes

Estos ejemplos, son más expositivos que analíticos, es decir, se analizan más las caracteristicas fundamentales que todos los detalles del modelo como se hizo en la sección anterior. Sin embargo, usando el paquete, se puede hacer el análisis de la misma forma.

0.4. Prueba con datos reales

8. Es decir, descartando las observaciones antes de k^* y adelgazando.

Métrica	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\beta}_2$
Mínimo	-3.34	0.63	0.00
Primer Quartíl	-2.50	0.79	0.00
Media	-1.65	0.82	0.00
Mediana	-1.74	0.81	0.00
Tercer Quartíl	-0.93	0.85	0.00
Máximo	1.01	0.99	0.00
Desviación Estandar	0.93	0.06	0.00

	$\hat{w}_1$					
Métrica	$\hat{w}_{1,1}$	$\hat{w}_{1,2}$	$\hat{w}_{1,3}$	$\hat{w}_{1,4}$	$\hat{w}_{1,5}$	$\hat{w}_{1,6}$
Mínimo	-12.60	-9.18	-23.62	-0.14	-16.06	-1.59
Primer Quartíl	-2.44	-2.87	-2.27	0.45	-3.31	0.28
Media	-1.55	-1.94	-1.64	0.97	-2.27	1.46
Mediana	-1.24	-1.56	-1.20	0.69	-1.53	0.88
Tercer Quartíl	-0.32	-0.72	-0.53	1.30	-0.90	2.29
Máximo	15.22	4.67	4.10	7.74	0.60	11.55
Desviación Estandar	2.10	1.74	1.94	0.78	1.98	1.84

	$\hat{w}_2$					
Métrica	$\hat{w}_{2,1}$	$\hat{w}_{2,2}$	$\hat{w}_{2,3}$	$\hat{w}_{2,4}$	$\hat{w}_{2,5}$	$\hat{w}_{2,6}$
Mínimo	0.00	-61080	-46710	-7448	-25550	$0-00$
Primer Quartíl	0.00	-7509	-5396	77.7	-4444	-43.9
Media	0.00	-4693	-3515	1543	-2917	160.07
Mediana	0.00	-2846	-1887	1227	-2320	355.6
Tercer Quartíl	0.00	-69	-82	2293	-103.2	3185
Máximo	0.00	29420	36720	4110	14410	3092
Desviación Estandar	0.00	7147	5691	1799	3562	452

Tabla 2: Resumenes numéricos para los parámetros del modelo presentado en el ejemplo 6.